HITACHI PROGRAMMABLE CONTROLLER

EH-150 EHV-CPU

PROGRAMMING MANUAL

NJI-482 (X)

○ Warranty period and coverage

The product warranty period will be one year after the product has been delivered to the location designated in the order. If a malfunction occurs within the warranty period even though the product has been used within the range of correct conditions according to the product specifications given in this document, we will exchange or repair the defective part free pf charge.

However, the following conditions are not be covered under this warranty:

(1) Damage due to negligent handling or misuse by the user.

(2) When the cause of the malfunction is due to components other than the product delivered.

(3) When the cause of the error is due to a modification or repair performed by an entity other than the supplier.

(4) When the cause of the error is due to weather or accidents that are out of the supplier's control.

Further, the warranty here refers to that of the product itself, and does not include any damage caused by the malfunction of the product.

○ General repair

Investigations and repairs outside the warranty period (1 year) will be charged. Also, we will repair damaged products caused by any reason not covered by the warranty and investigate the cause of malfunctions for a charge even within the warranty period. Please contact the place or purchase. (Research may not be possible, depending on the area of malfunction.)

○ Ordering parts or asking questions

When contacting us for repair, ordering parts or inquiring about other items, please have the following details ready before contacting the place of purchase.

(1) Model

(2) Manufacturing number (MFG no.)

(3) Details of the malfunction

Warning

(1) This manual may not be reproduced in its entirety or nay portion thereof without prior consent.

(2) This content of this document nay be changed without notice.

(3) This document has been created with utmost care. However, if errors or questionable area are found, please contact us.

Safety Precautions

Read this manual and related documents thoroughly before installing, operating , performing preventive maintenance or performing inspection, and be sure to sue the unit correctly. Use this product after acquiring adequate knowledge of the unit, all safety information, and all cautionary information. Also, make sure this manual enters the possession of the chief person in charge of safety maintenance.
Safety caution items are classified as “Danger” and “Caution” in this document.

◇! DANGER : Cases where if handled incorrectly a dangerous circumstance may be created, resulting in possible death or severe injury.

⚠ CAUTION : Cases where if handled incorrectly a dangerous circumstance may be created, resulting in possible minor to medium injury to the body, or only mechanical damage.

However, depending on the circumstances, items marked with ⚠ CAUTION may result in major accidents.

In any case, they both contain important information, so please follow them closely.

Icons for prohibited items and required items are shown below:

⃠ : Indicates prohibited items (items that may not be performed). For example, when open flames are prohibited, 🚫 is shown.

❗ : Indicates required items (items that must be performed). For example, when grounding must be performed, ⏚ is shown.

1. About installation

⚠ CAUTION

- Use this product in an environment as described in the catalog and this document.
  IF this product is used in an environment subject to high temperature, high humidity, excessive dust, corrosive gases, vibration or shock, it may result in electric shock, fire or malfunction.
- Perform installation according to this manual.
  If installation is not performed adequately, it may result in dropping, malfunction or an operational error in the unit.
- Do not allow foreign objects such as wire chips to enter the unit.
  They may become the cause of fire, malfunction or failure.

2. About wiring

REQUIRED

- Always perform grounding (FE terminal).
  If grounding is not performed, there is a risk of electric shocks and malfunctions.

⚠ CAUTION

- Connect power supply that meets rating.
  If power supply that does not meet rating is connected, fire nay be caused.
- The wiring operation should be performed by qualified personnel.
  If wiring is performed incorrectly, it may result in fire, damage, or electric shock.

3. Precautions when using the unit

DANGER

- Do not touch the terminals while the power is on.
  There is risk of electric shock.
- Structure the emergency stop circuit, interlock circuit, etc. outside the programmable controller (hereinafter referred to as PLC).
  Damage to the equipment or accidents may occur due to failure of the PLC.
  However, do not interlock the unit to external load via relay drive power supply of the relay output module.

⚠ CAUTION

- When performing program change, forced output, RUN, STOP, etc., while the unit is running, be sure to verify safety.
  Damage to the equipment or accidents may occur due to operation error.
- Supply power according to the power –up order.
  Damage to the equipment or accidents may occur due to malfunctions.

4. About preventive maintenance

◇! DANGER

- Do not connect the ⊕, ⊖ of the battery in reverse. Also, do not charge, disassemble, heat, place in fire, or short circuit the battery.
  There is a risk of explosion or fire.

⊘ PROHIBITED

- Do not disassemble or modify the unit.
  These actions may result in fire, failure, or malfunction.

⚠ CAUTION

- Turn off the power supply before removing or attaching module / unit.
  Electric shock, malfunction or failure may result.

Revision History

No.	Description of Revision	Date of revision	Manual Number
1	The first edition	2006.03	―

Table of contents

Chapter 1 Introduction

Thank you for introducing a Hitachi Programming Logic Controller (hereinafter referred to as PLC), EH-150 Series. The contents related to programming of EH-150 series CPU module (hereinafter referred to as EHV-CPU) are described in this manual. Please apply this manual to programming after reading carefully.

Please also refer to the following manuals.

Table1.1 Description material list

Items	Title of material	Manual number
EH-150	EH-150 (EHV) Application manual	NJI-481
Programming software	Control editor start up guide	NJI-486

* Alphabet is appended to the end of manual number according to version-up. (none➔ A ➔ B ➔ …)

1.1 Confirming of purchases

EHV-CPU has been manufactured elaborately. However, please confirm the following respects immediately after purchasing it.

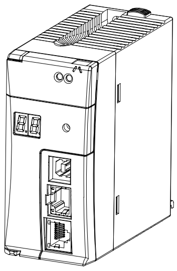
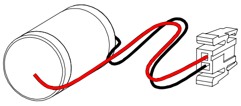
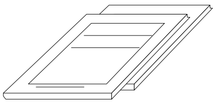
If malfunction are found, please apply to the place of purchase.

(1) Is model and type as ordered?

(2) Has not the product damaged?

(3) Are all products listed in table 1.2 complete?

Table 1.2 Packing list of EHV-CPU

No.	Name	Type	The exterior	Total number	Remarks
1	CPU module	EHV-CPU128		1	The cover is attached to each port of USB, Serial and Ethernet.
2	Battery	LIBAT-H		1	
3	Instruction manual	NJI-442		1	

1.2 Doing after unpacking

(1) Mounting of battery

The battery is not mounted in EHV-CPU when shipped. (The battery is stored in the module. However, the battery is not connected to the battery connector.)

If you want to use a watch function and hold internal data when the power supply is off, please use EHV-CPU after mounting the battery.

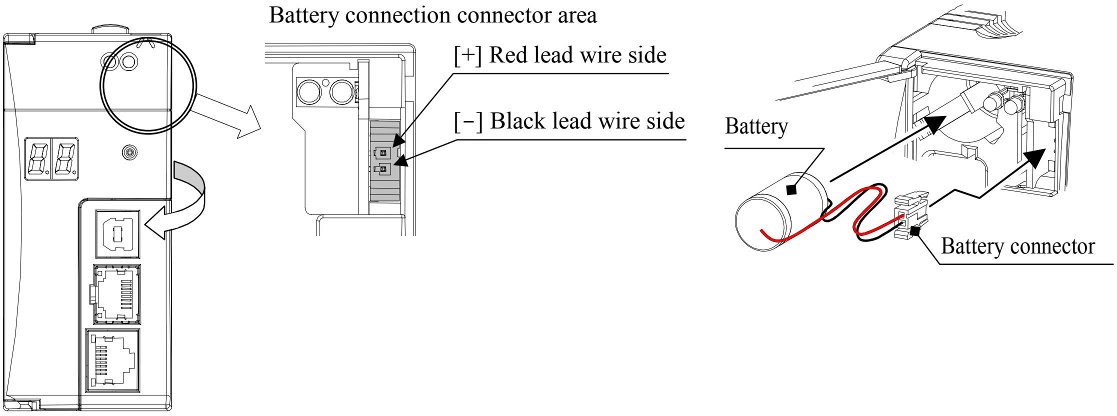


Figure 1.1 Connecting of Battery

◇! Danger: Handling precautions of battery

Be sure to mount an attached battery. Otherwise, there is a risk of exploding.

Do not connect plus and minus of a battery inversely. Do not charge a battery, do not take it apart, do not heat it, do not throw it into the fire, and do not short it.

(2) Initializing of User program

Since a memory in EHV-CPU is changeable at first, there is a case where an error code which stands for memory error is displayed on 7 segment LED. Please initialize the memory in EHV-CPU at the first step after connecting a battery. Please initialize the memory in EHV-CPU at the beginning after connecting a battery.

☞ Selecting “CPU initialize” of “CPU operation” of “On-line” in menu in the programming tool, you can initialize EHV-CPU.

Reference

Users program, data memory, and a part of parameter are initialized by “CPU initialize”. Communication parameter is not initialized.

(3) Setting of Communication parameters

Communication parameter is valid when the power supply is on. Default values are set at the shipment. Therefore, please set necessary parameters first after connecting the programming tool. After that, please reswitch the power supply on

(Set-up parameters are memorized in a back-up memory. Once a setup is completed, it is not necessary to set up later.)

* If connecting to the Serial communication port or the Ethernet port, a communication setup of the programming tool should be set as default values shown in the following table. There is no parameter to need to set in USB port.

Table1.3 Communication parameter (at the shipment)

No.	Parameter			At the shipment
1	IP address	IP address		192. 168. 0. 1
		Subnet mask		255. 255. 255. 0
		Default getaway		0. 0. 0. 0
2	NTP	Valid / Invalid		Invalid
		Time zone		GMT + 09:00
3	Serial communication setting	Particular / General purpose		Particular
		Type of port		RS-232C
		Communication speed		38400 bps
		Communication process		Transmission control process 1 (1:1)
		Modem connection yes or no		No
4	Ethernet communication setting (Task code)	Port 1	Valid / Invalid	Valid
			Port No.	3004
			Protocol	TCP/IP
		Port 2	Valid / Invalid	Valid
			Port No.	3005
			Protocol	TCP/IP
		Port 3	Valid / Invalid	Valid
			Port No.	3006
			Protocol	TCP/IP
		Port 4	Valid / Invalid	Valid
			Port No.	3007
			Protocol	TCP/IP
		Timeout time		30
5	Ethernet communication (ASR) setting	Port 1	Valid / Invalid	Invalid
		Port 2	Valid / Invalid	Invalid
		Port 3	Valid / Invalid	Invalid
		Port 4	Valid / Invalid	Invalid
		Port 5	Valid / Invalid	Invalid
		Port 6	Valid / Invalid	Invalid

☞ Selecting various setting from “CPU setting” of “Tool” in menu in the programming tool, you can change the setting.

(4) Setting of Watch data (when using a watch function)

If the power supply is switched on after opening the package (or after leaving the battery not connected for along time), a watch data updates the time from an initial value. If the watch function is used, the watch data should be set by the programming tool after connecting the battery.

☞ Selecting “CPU Calendar watch setting” of “CPU setting” in menu in the programming tool, you can set the arbitrary time, or the time of the personal computer which is connecting.

Reference

The initial value of the clock is 00:00:00 on Saturday, January 1, 2000.

1.3 Programming manual

This is a manual to create a program for EHV-CPU. Please make use of this manual when creating a program. And manuals related to EHV-CPU are shown in Table 1.4.

Table1.4 Manual related to EHV-CPU

Number	Title	Contents
NJI-481 *	EH-150 Application Manual (for EHV-CPU)	Details of usable modules by combining with EHV-CPU, installing PLC, how to wire, how to use special functions in EHV-CPU, etc.
NJI-486 *	Control Editor Manual	How to operate Control Editor, convenient functions, handling precautions, etc.

* Alphabet is appended to the end of manual number according to version-up. (none ➔ A ➔ B ➔ …)

■ Common terms

Common terms used in all chapters are as follows. As for any other terms, please see each page.

Programming tool … This is a software to create a program running on the personal computer.
In EHV-CPU, the program is created by the software named “Control Editor”.

Peripheral equipment … This means external devices to communicate with PLC. For example , a touch panel and SCADA. The programming tool is also a kind of peripheral equipment.

■ Symbols used in this manual

☞	The reference place of a related explanation, the explanation for operation of the programming tool
Reference	A supplementary explanation and helpful information
Notice	A point to notice when creating program
W Explanation for terms	Explanation for characteristic terms used in sentence

Chapter 2 Basic operations of CPU module

CPU module can run by two kinds of program, one is a system program which controls the CPU module and the other is a user program which users create.

The system program is a program always executed while CPU module is being switched on. The system program monitors abnormalities in CPU module, and a run and a stop of the user program. The user program is a program which users create by a programming tool. The user program is executed if operation conditions are complete, and it stops if not.

2.1 Structure of CPU module

EHV-CPU consists of two processors which are a main processor and an operation processor, a user memory, a backup memory, a data memory, and a system memory.

The internal structure of EHV-CPU is shown in the figure 2.1.

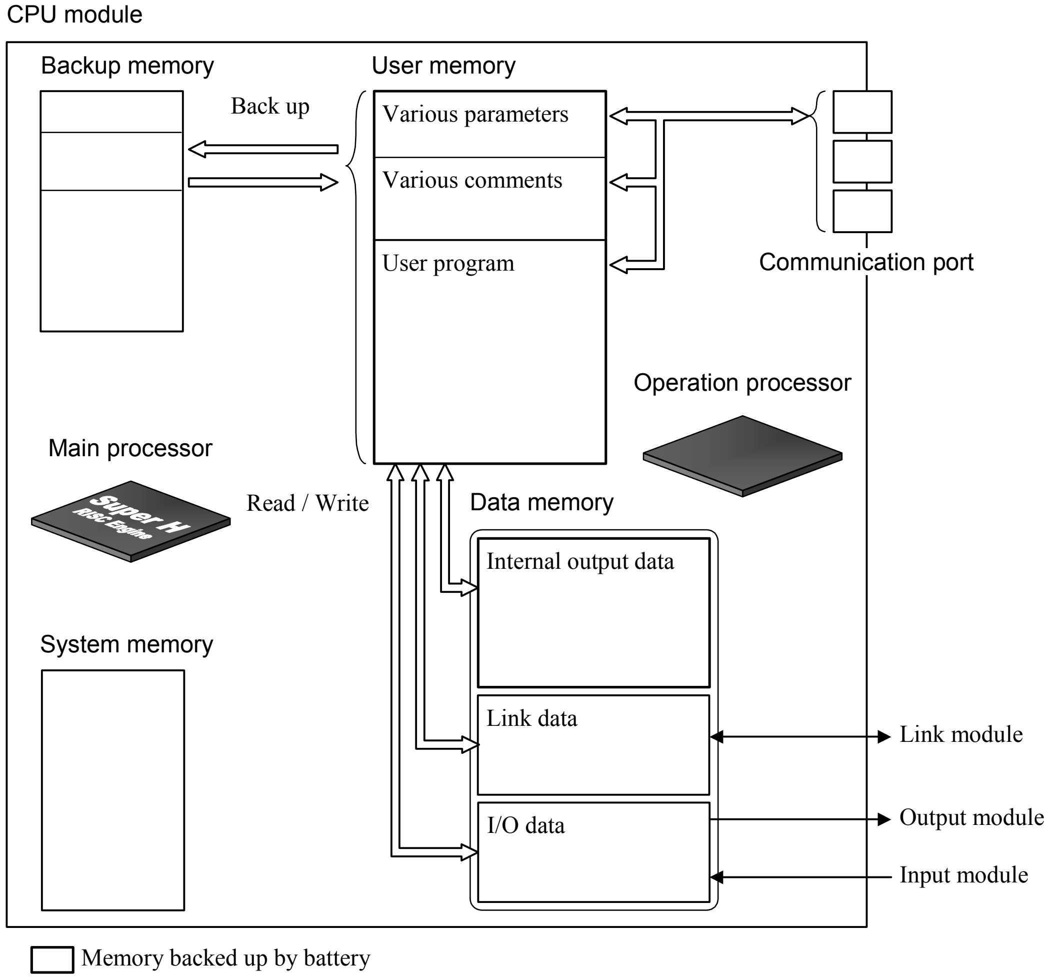


Figure 2.1 Internal structure of EHV-CPU

(1) Main processor

This is a processor to execute a part of the system program and the user program.

(2) Operation processor

This is a processor to execute the user program.

(3) User memory

The user memory is a memory to memorize the user program, various parameters, and various comments.

The user program transferred from a programming tool is written in the user memory.

■ User program

The user program is a program to run operations specified by users on PLC. There are mounting information of I/O and parameters related to the operation of PLC besides the part for writing with commands combining in the user program.

☞ See “Chapter 4 – Procedure to create user programs”

■ Various parameters

There are various kinds of parameters, such as parameters related to a communication setting, an error indication, and execution of the program.

There are two methods to set a parameter, one is a method to set only applicable parameters, the other is a method to set with appending to the user program when transferring the program. (The setting method depends on kinds of parameters.)

☞ See “Chapter 4 – Procedure to create user programs” about an operation parameter, and the application manual about other parameters.

■ Various comments

A comment is the memo written in the program in order to make the user program intelligible. There are the I/O comment, the circuit comment and the box comment in the comment.

In EHV-CPU, it is possible to memorize the comment together with the program inside CPU module.

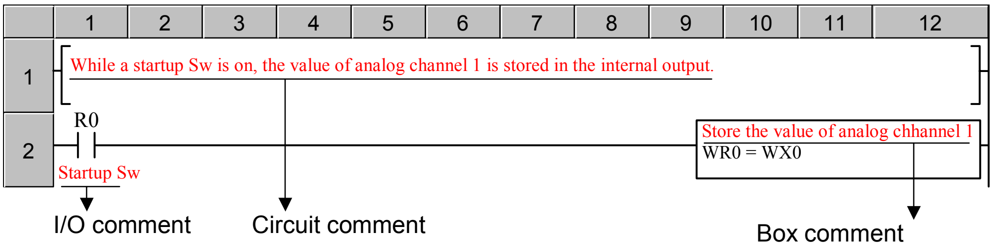

1
2
3
4
5
6
7
8
9
10
11
12
1
While a startup Sw is on, the value of analog channel 1 is stored in the internal output.
2
R0
Startup Sw
Store the value of analog chhannel 1
WR0 = WX0
I/O comment
Circuit comment
Box comment

Figure 2.2 Example of variable comments

(4) Backup memory

A backup memory is the memory to retain data on the user memory. Data is automatically saved in the backup memory when the program is written (including changed during run). And when the power source is on, data is restored from the backup memory if the user memory is not settled.

Reference

The value of the data memory is not memorized into the backup memory. Please set up the power failure memory area after setting the battery in order to retain the value of the data memory when the power source is off.

(5) Data memory

A data memory is a memory to memorize the result calculated on the user program and I/O data.

Since the internal output data can be backed up with the battery particularly, if the data memory is specified as a power failure memory area, that data can be retained even if the power source is off.

■ Internal output area

A register which is used as a work at the operation by the user program, and is used when recording data, is call “Internal output”. There are two areas in the internal output, one consists of only bit data and the other consists of word data.

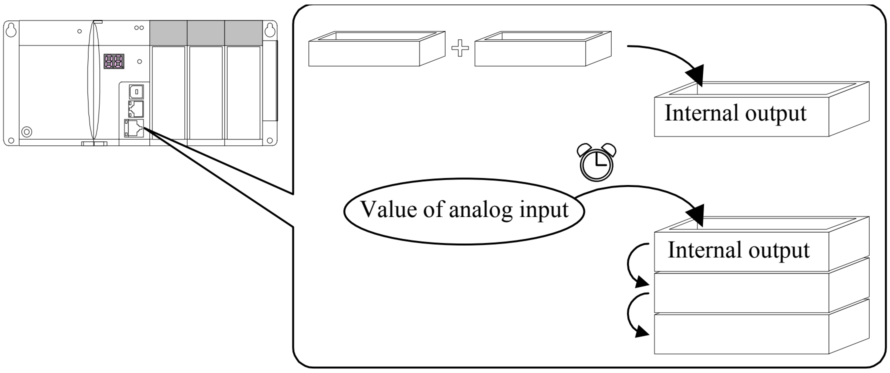


Figure 2.3 Image of internal output

■ Link data area

Data can be shared with another PLC by using the link module. This function is called a data link or a link simply.

The link data area consists of two areas, one (local station) is an area to store data to transfer to another PLC, and the other (other station) an area to store received data. And the link module is updated automatically.

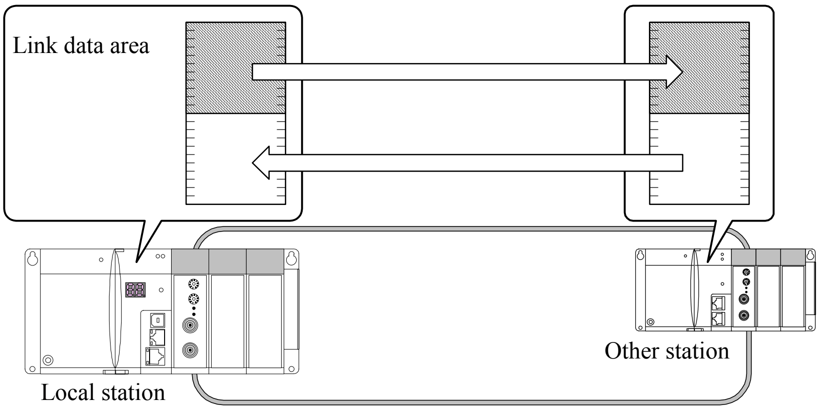


Figure 2.4 Outline of link data area

In addition, if the link module is not used, the link data area can be all used as the internal output.

■ I/O data area

I/O data area is an area to store input information taken in from the input module and to store output information specified by the user program. CPU module takes in the input information and reflects the output information in the external output automatically. This process is called I/O refresh.

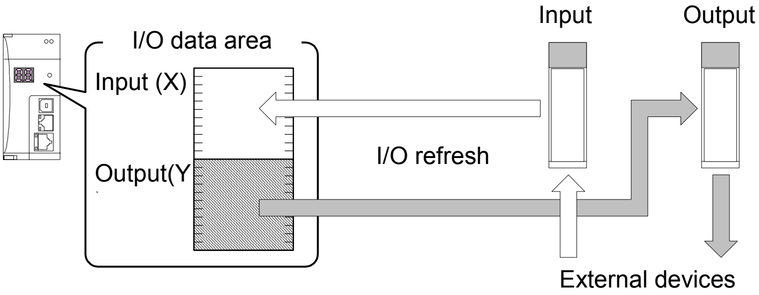


Figure 2.5 Outline of I/O data area

(6) System memory

System memory is a memory to store the system program to control CPU module. The system program cannot be rewritten.

2.2 Operating and Stopping

The state where CPU module is executing the user program is called “RUN”, and the state where it is not is called “STOP”.

(1) Stop ➔ Run

CPU module has no abnormality, and the user program can be run by means of starting operation of CPU module in which the normal user program is stored.

Reference

Internal information which has not been specified to the power failure memory is cleared when the operation is started.

(2) Run ➔ Stop

While CPU module is operating, the running user program can be stopped by means of stopping the operation.

And even if abnormality is detected under operation, the user program stops.

In addition, if serious abnormality is detected, not only the user program but also the system program stops.

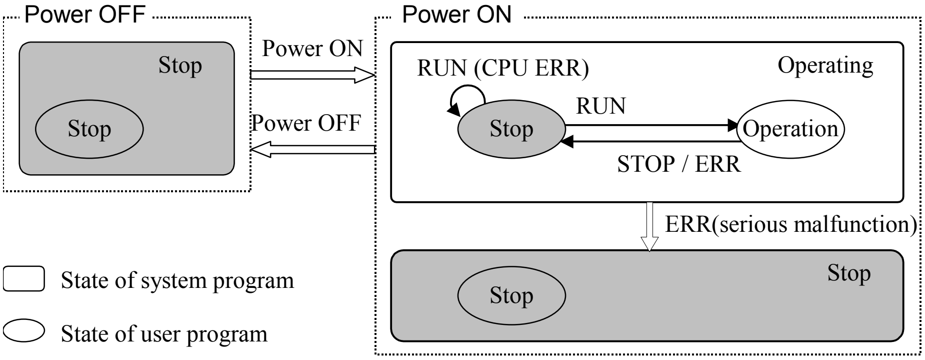


Figure 2.6 State transition diagram of operation and stop

Reference

While the operating is stopping, the internal information remains unchanged and the external output is cut off (OFF).

(3) Abnormality which CPU module detects

The abnormality which CPU module detects is classified into the following levels, serious malfunction, medium malfunction, slight malfunction, and warning. A situation of operation when the abnormalities of each level occur is shown the following table.

Table 2.1 Abnormalities which CPU module detects

Level	Details	Operation / Stop
Serious malfunction	Power fail, Micon error, System ROM error, System RAM error, System bus error, etc. This means that there are serious abnormalities which cannot return.	Stop
Medium malfunction	Data memory abnormality, System program abnormality, User memory abnormality, etc. This means that it may result in an operational error if the operation is continued.	Stop
Slight malfunction	I/O information checking error, Remote abnormality, Delay error, I/O assignment points over, etc. This is an operational continuable abnormality by setting the operation parameter.	Stop (Continuation by specification)
Warning	Transmission error, etc. This is quite a slight and an operational continuable abnormality.	Continuation of operation

☞ See “Appendix 1 – Self diagnosis” for details of error code.

(4) Operation of Start and Stop

The following diagram is showing the operation of start and stop, and the state of CPU module.

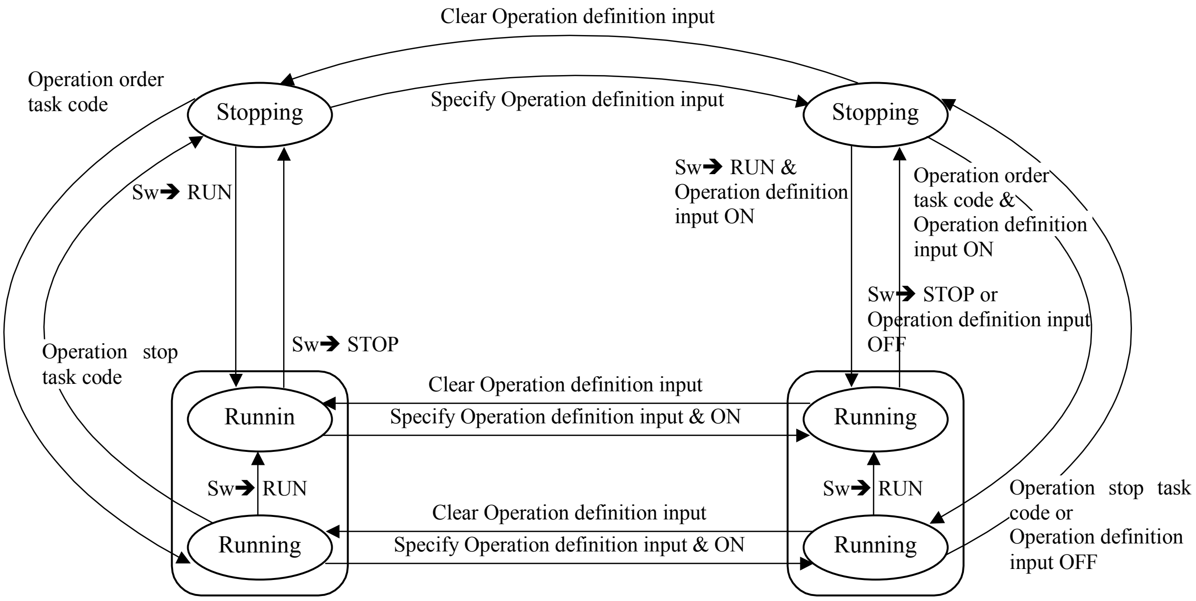


Figure 2.7 State transition diagram of start and stop

One of the hardware switch (RUN / STOP Sw), the input of operation definition, and the task code communication is used for start and stop. However, if there are several conditions, the operation is not started if both do not come into effect (as the switch of RUN/STOP is RUN and the operation definition input is ON). And the operation is stop even if only one condition does not come into effect.

Explanation for terms

Task code communication

EHV-CPU usually communicates using the exclusive communication protocol. This protocol is called “High protocol”. Because a command in a communication format decided by the high protocol is called a task code, the communication by the exclusive protocol is called also “Task code communication”.

In addition, because the control editor has a function to call RUN / STOP, RUN and STOP of operation is possible by the communication on the control editor even if there is no knowledge for the communication format of the task code.

2.2.1 Working under operation

The following chart is showing an outline of the working while CPU module is operating.

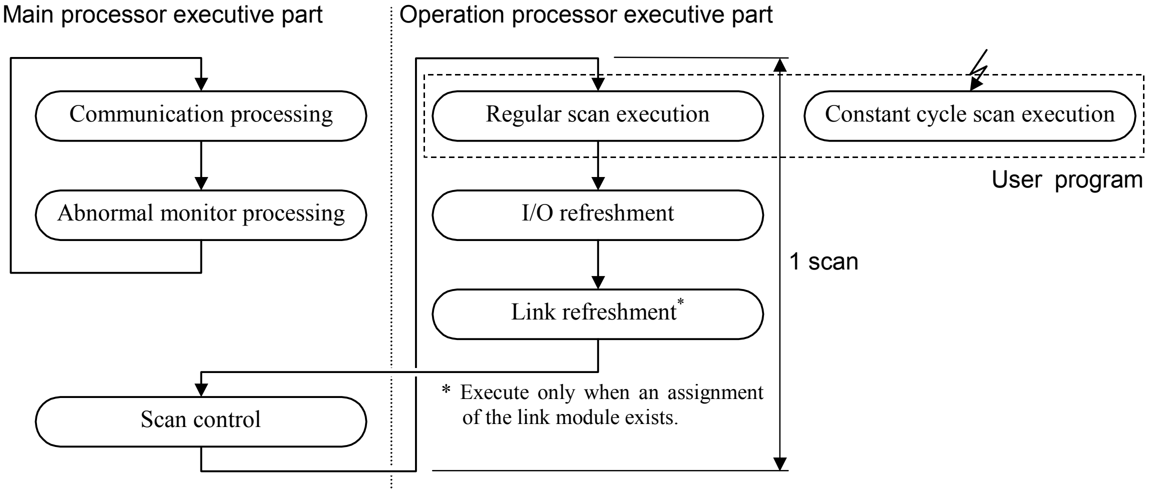

Main processor executive part
Operation processor executive part
Communication processing
Abnormal monitor processing
Scan control
Regular scan execution
Constant cycle scan execution
User program
I/O refreshment
Link refreshment*
1 scan
* Execute only when an assignment of the link module exists.

Figure 2.8 Outline of Working under running

(1) A part of Main processor

An important part of the main processor is a communication processing and an abnormal monitoring processing. However, these processings are called “System processing” overall, the system processing is executed in spite of the running or stopping of CPU module

And a scan control processing is executed only when CPU module is running. The scan control is a processing to monitor abnormality related to the scan and control an operation processor. The scan control processing is executed once per one regular scan.

(2) A part of Operation processor

The operation processor is a processor to run the user program. And this refreshes the I/O data and the link data which are a part of the user program.

Reference

The user program is roughly classified into a regular scan and a constant cycle scan. A regular scan is a program to be executed cyclically while CPU module is operating. “Scanning time” means the time from a start of the regular scan until a completion of the system processing. And a constant cycle scan is a program to execute only once whenever a predecided cycle comes. In addition, since the constant cycle scan has the high priority to execute, for example, the constant cycle scan is executed interrupting even if I/O of the regular scan is refreshing.

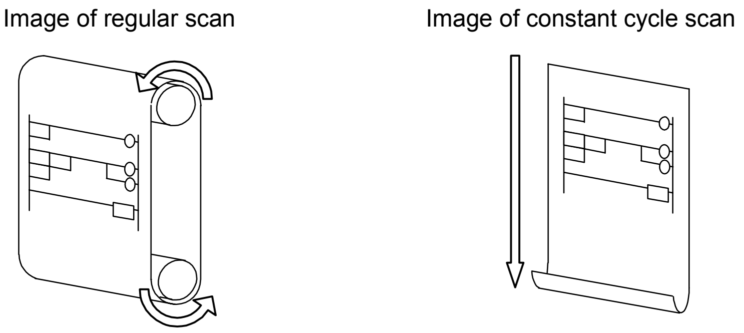


Figure 2.9 Image of Regular scan and Constant cycle scan

2.2.2 Working under stop

The following chart is showing an outline of the working while CPU module has stopped.

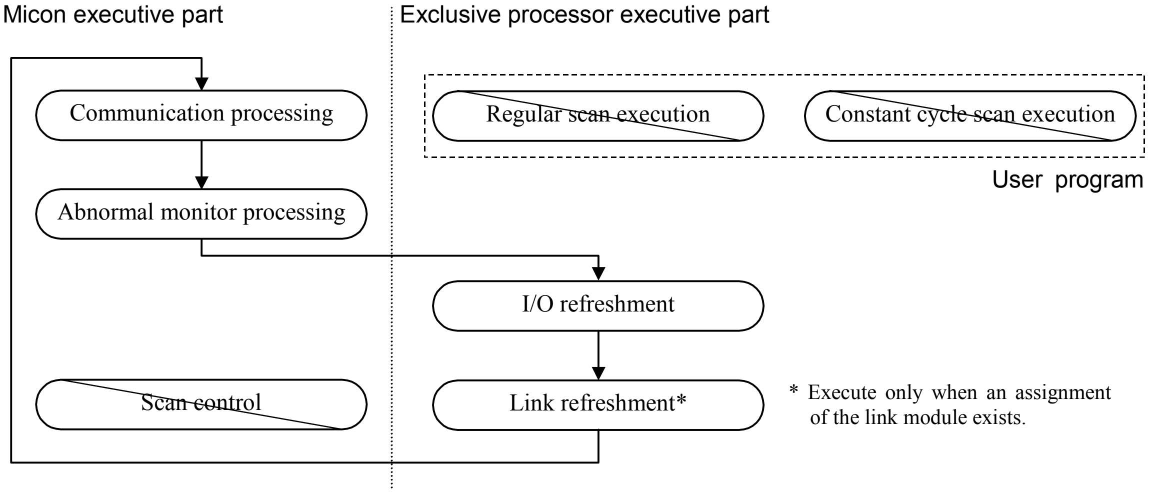

Micon executive part
Exclusive processor executive part
Communication processing
Abnormal monitor processing
~~Scan control~~
~~Regular scan execution~~
~~Constant cycle scan execution~~
User program
I/O refreshment
Link refreshment*
* Execute only when an assignment of the link module exists.

Figure 2.10 Outline of Working under stop

Although the user program is not executed under stop, the I/O refreshment and the refreshment of the link data are executed.

Reference

Since the refreshment processing works even under stop, it is possible to monitor a state of the input and to turn the output on by a programming tool.

In addition, CPU module clears I/O data and the internal output not to specify to a power failure memory till 0 at the starting of the operation. Therefore, an output turned on during the stop is tuned off at the same time CPU module is started. And after that, the output is turned on and off in accordance with the user program.

2.2.3 Updating of each data

PLC deals with the following data. One is an external I/O, a link data, and a data (internal output) used by the user program.

(1) External I/O

CPU module updates an input data on the data memory to the state of an external input and reflects an output data on the data memory in the output module.

Whether CPU module runs or stops, the CPU module can update I/O data. The CPU module updates the data at regular cycle during a stop and updates the data in a lump at the end of the user program (Scan END) during a run. (The way to update I/O data in a lump during a run is called “Refresh method”.)

Ladder program refers to data on the data memory. If a value to output was changed in the middle of a scan, for example, the operation would be performed using the changed value in scans on and after that. (The value at the time of Scan END is output finally.)

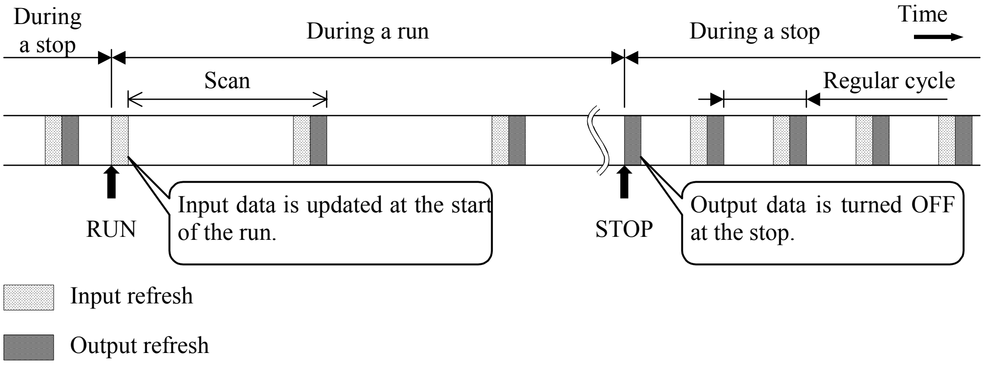


Figure 2. 11 Conception of External I/O refresh

Reference

Methods of reading the state of external input and reflecting it in external output at the time the command is executed are called “Direct method”. The refresh method is used in EHV-CPU but external I/O data can be refreshed in the middle of a scan by using the command to refresh I/O

(2) Link data

Link data similarly to external I/O is updated at regular cycle during a stop and updated in a lump at Scan END during a run.

Link data is 1k words per loop. An area of own station’s link data (data that is sent to a CPU module of other stations via a link module) is allocated to the area of this 1k-word. Areas except own station’s link data area become other station’s link data area, in which data that link module received from other stations is stored. Since a CPU module and a link module are usually in motion asynchronously , the data is updated to other station’s link data at a point in time when the CPU module refreshed the link data.

(3) Internal output data

Whether a CPU module runs or stops, an internal output data reflects the value at a point in time of set.

Further, the value of the internal output is cleared to return to '0' at the start of operation except a power failure storage area. And the just previous value that the operation stops is retained at the operation stop.

Reference

The internal output can be set up so that the value can be retained (a power failure storage setting) even if the switch of PLC is off. A battery is required for the power failure storage.

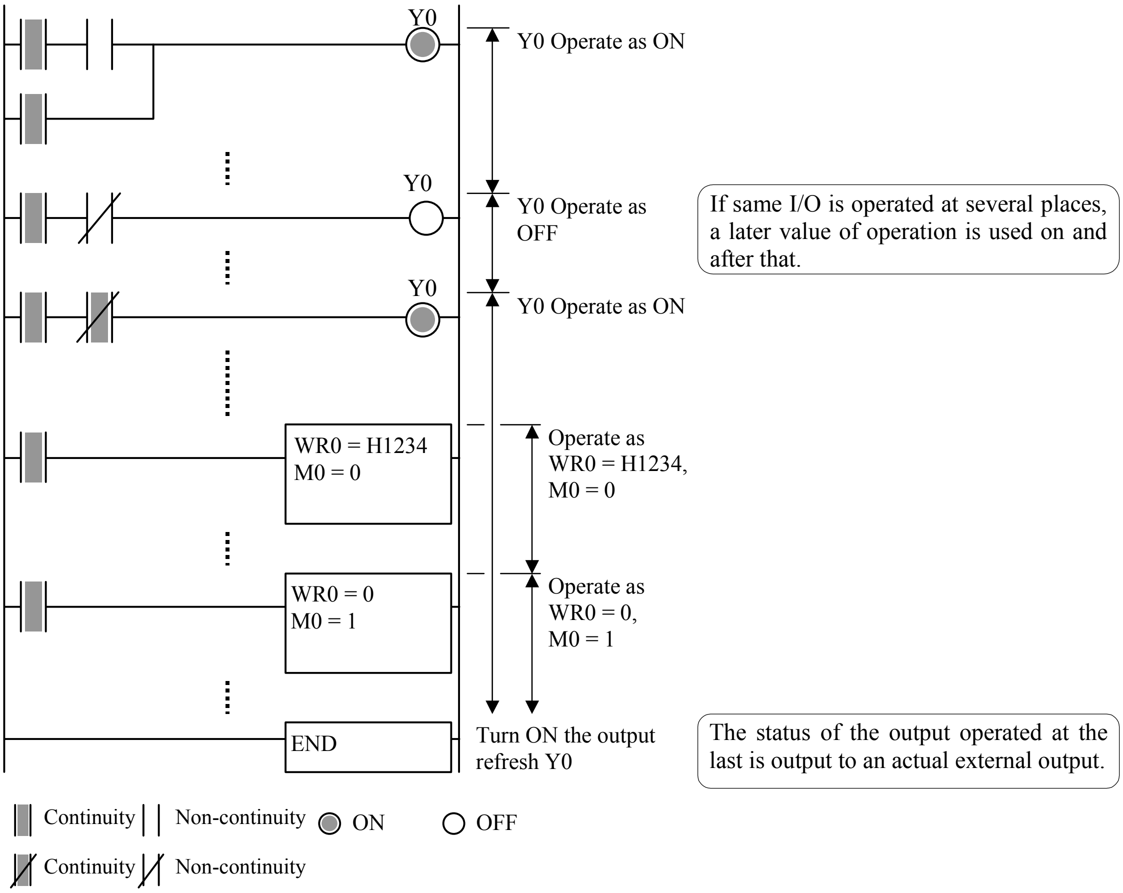

Y0
Y0 Operate as ON
Y0
Y0 Operate as OFF
If same I/O is operated at several places, a later value of operation is used on and after that.
Y0
Y0 Operate as ON
WR0 = H1234
M0 = 0
Operate as WR0 = H1234, M0 = 0
WR0 = 0
M0 = 1
Operate as WR0 = 0, M0 = 1
END
Turn ON the output refresh Y0
The status of the output operated at the last is output to an actual external output.
Continuity
Non-continuity
ON
OFF
Continuity
Non-continuity

Figure 2.12 Example for Data refresh

2.2.4 Processing of System

A processing to control motions of CPU module after a main processor is executed is called a system processing.

The system processing includes processing as shown below.

(1) Communication processing

This is the processing to communicate with peripheral devices connected via high-functional modules such as a communication port of CPU module and Ethernet module.

(2) Abnormal monitoring (Self checkup)

This is a processing to monitor whether there are any abnormalities in the CPU module.

Abnormal causes are classified into 4 types, as a serious malfunction, a medium malfunction, a slight malfunction and warning and the later operation of detection will change in accordance with the level for generated abnormalities. The operation of the CPU module stops at the generation of the serious malfunction. The run stops at the generation of the medium malfunction. Only the run stops or an error is displayed at the generation of the slight malfunction and warning.

(3) Scan control

The operation processor mainly executes and controls a user program but a main processor detects abnormalities for the operation processor and changed the user program. These processors are called a scan control processor and executed one per one scan.

The scan control is a processing of top priority in the system processor.

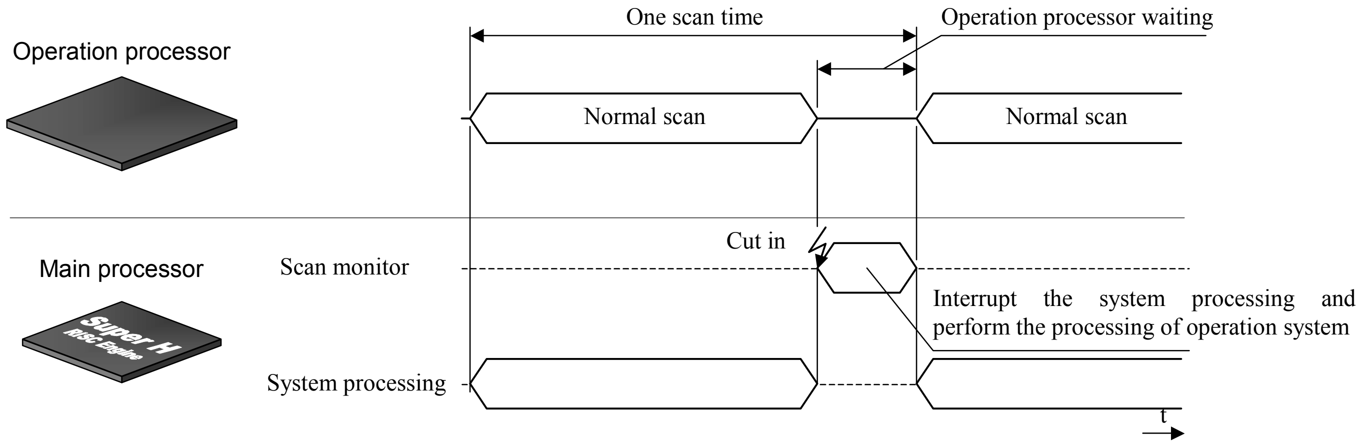


Figure 2.13 Operation of System

Notice

If a scan time is short*, it is getting difficult to spend enough time for the system processing because the main processor increases in frequency to execute the scan monitoring processing. Therefore, the performance to respond for communication deteriorates.

* A standard is less than 1ms.

Please set up the system processing time by a programming tool if the performance to respond for communication is bad since it is possible to set up time for the system processing inside the scan monitoring processing in EHV-CPU.

☞ See "Chapter 4 – Procedure to create user programs" for a setting for the system processing time.

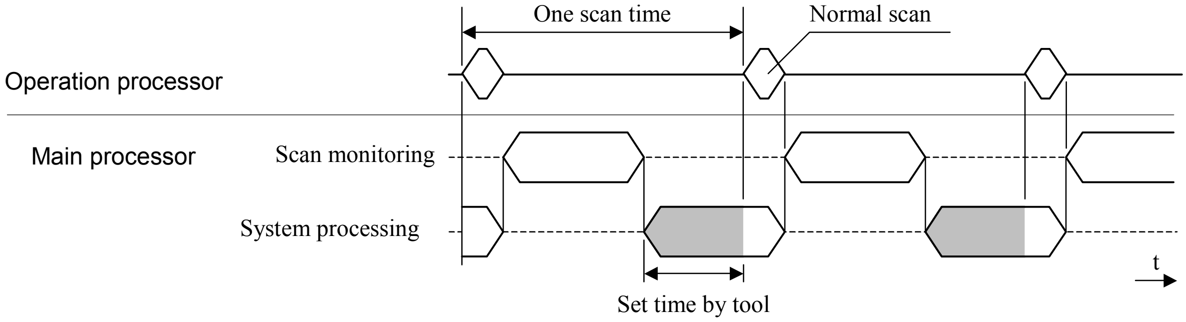


Figure 2.14 Operation of System (at setting for a system processing time)

Chapter 3 User Program

3.1 Structure of user program

Configuration of user program is shown in fig. 3.1.

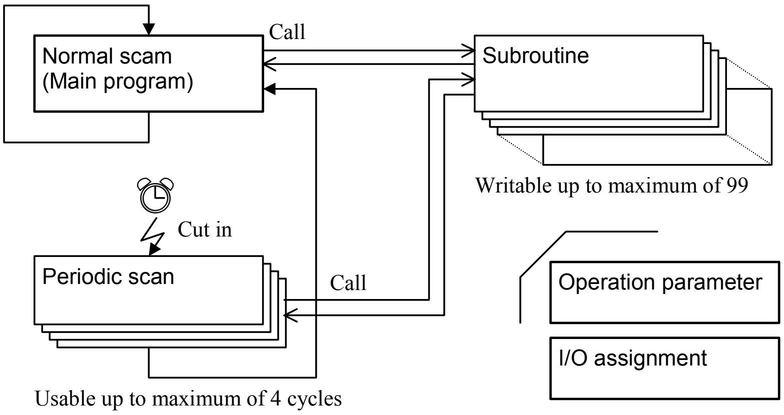


Figure 3.1 Configuration of user program

User program is configured with a program section described combining commands, an operation section and an I/O assignment table. The user program usually means a program section but the program is transmitted as an operation parameter and I/O assignment table are added if the program is transmitted from the programming tool. Both are important components in order to execute the program.

(1) Program section

The program section can be classified into three types, “a normal scan”, “a periodic scan” and “ a subroutine”.

A normal scan is always required because it is a main program of the program particularly. A periodic scan and a subroutine should be used if necessary.

(2) Operation parameter

Parameters related to an operation and an error display of CPU module are set up. An operation parameter is indispensable to the user program but you can write into CPU module without setting the parameter since a default value is set up when a new program is created. Parameter setting should be changed depending on uses.

(3) I/O assignment table

I/O assignment is information for mounting modules. CPU module updates external I/O data and transfers data with a high-functional module based on this information.

When the program is input, it is necessary to set an I/O assignment before inputting the program because an error occurs if I/O number of external input and output is written without the I/O assignment.

3.2 Normal scan

(1) Definition and motion of normal scan

A normal scan means an operation of a main program and an execution of END command (Scanning END processing).

A main program (a normal scan) will be executed from the beginning if CPU module is run, and the main program will be executed from the beginning again if executed till END command which indicates the end.

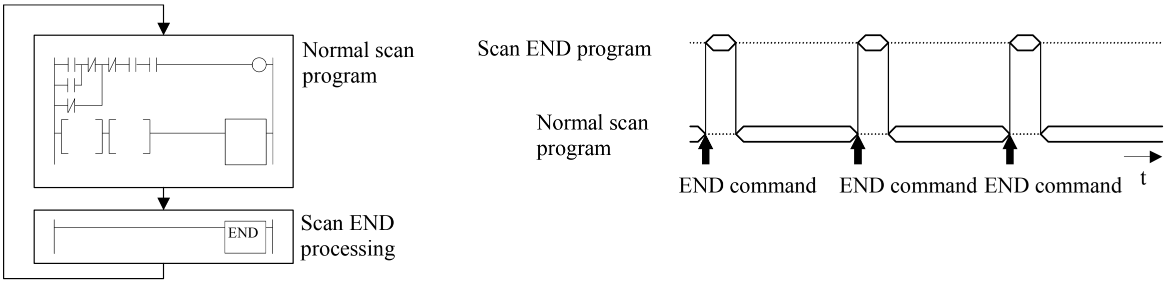


Figure 3.2 Normal scan outline

Reference

If a periodic scan and a subroutine are not combined, END command can be omitted.

(2) The cause that an overload error occurred

There are the following two kinds in the cause that the normal scan causes the overload error.

A) Scan time exceeded an overload check time because one scanning time was too long.

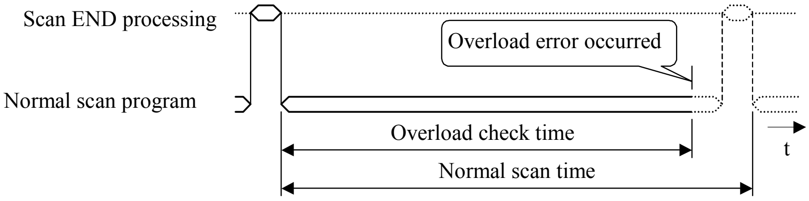


Figure 3.3 Overload error of normal scan (1)

B) An overload check time was exceeded since the situation that a normal scan was stagnant with a periodic scan.

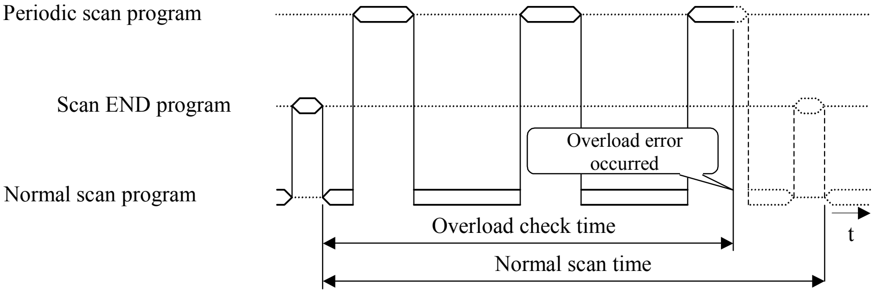


Figure 3.4 Overload error of normal scan (2)

The normal scan stops the processing if the periodic scan starts up, but an overload error monitor does not stop. Therefore, there are cases where it becomes the overload error by adding the periodic scan.

(3) Continuation of operation after overload error

If the motion is specified to “Continuation of operation” by the operation parameter when the overload error of the normal scan has occurred, the overload error will not be detected. Therefore, the normal scan executes the scan irrespective of the overload monitoring time.

Notice

If a program to loop infinitely inside the normal scan is created, the scan does not stop.

In this case, I/O is not refreshed since the scan END processing is not executed. And a change during RUN is also impossible since a change during RUN is performed by the scan END.

(4) Convenient functions of programming tools

Programming tool can describe a program of the normal scan with dividing into each sheet.

A program that is easy to see can be created by using this function.

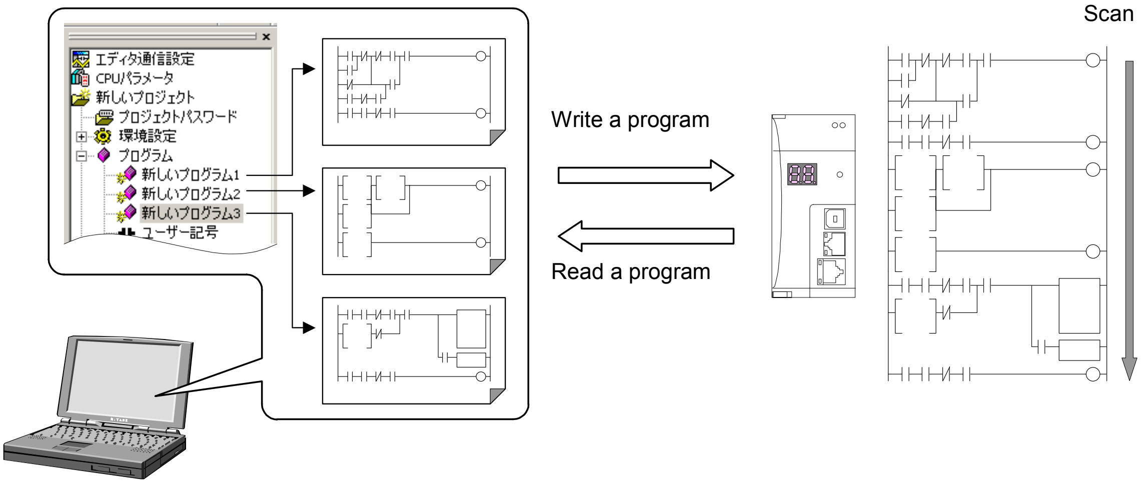


Figure 3.5 Sheet division of normal scan

Notice

CPU module executes a sheet registered in the project tree in order from the top. If you create a program to process in sequence, take care on arranging order of sheets.

And if a subroutine and a periodic scan are used as another sheets, the END command should be described on the end of sheets for the normal scan.

3.3 Periodic scan

(1) Definition and motion of periodic scan

A periodic scan executes a program surrounded by INT and RTI in a fixed cycle. After CPU module is run, a program will execute in sequence from INT to RTI if the setup cycle is reached. A periodic scan can create a program with four cycles at the maximum. (More than two periodic scans with same cycle cannot be created.)

Since a periodic scan has priority over a normal scan, when it is time to execute the periodic scan, the normal scan is interrupted and the periodic scan starts. When the program of the periodic scan finishes executing, the normal scan restarts from the interrupted processing.

And since the shorter a cycle of the periodic scan is, the higher the order of priority is, even if the periodic scan is in the middle of executing, when the periodic scan with higher priority is executed, the periodic scan under executing is interrupted and the program is executed. And the shorter a cycle of the periodic scan is, the higher the order of priority. Therefore, even if the periodic scan is in the middle of executing, when the periodic scan with higher priority is executed, the periodic scan under execution is interrupted and the program is executed.

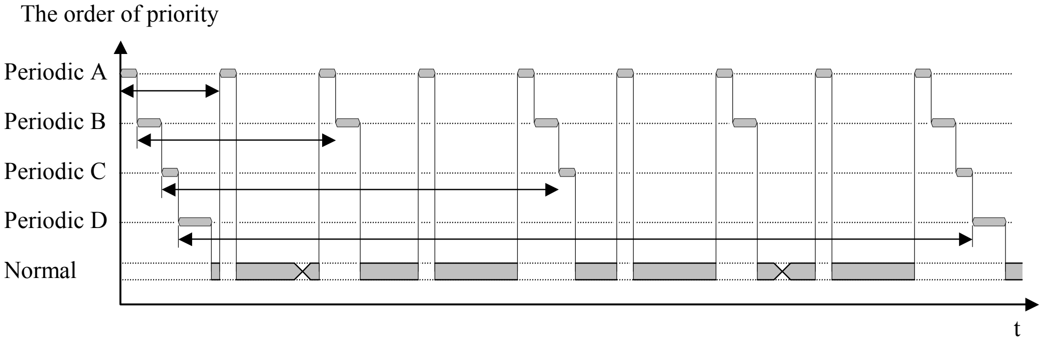


Figure 3.6 Execution timing of scan

(2) Occurrence causes of overload error for periodical scan

If a periodic scan with the same cycle starts under execution of the periodic scan, overload error will occur.

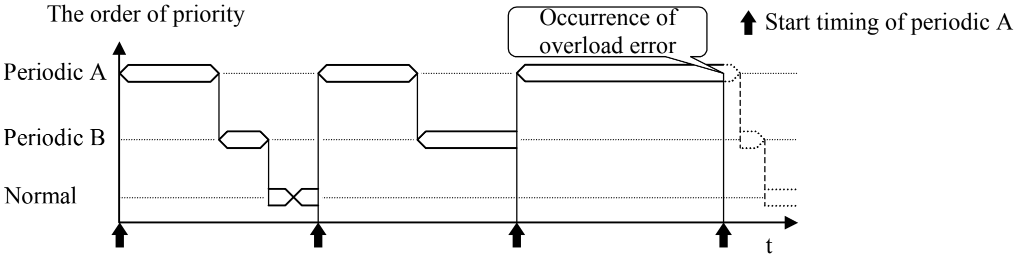


Figure 3.7 Overload error of periodic scan

(3) Continuation of operation after overload error

If the motion is specified to "Continuation of operation" by the operation parameter when the overload error of the periodic scan has occurred, the periodic scan executed until then will be interrupted at the timing of start of the periodic scan and it will be executed from the beginning again. In this case, processing after interrupting in the program is not executed. And only periodic scan which has priority over the periodic scan in which overload error has occurred is executed.

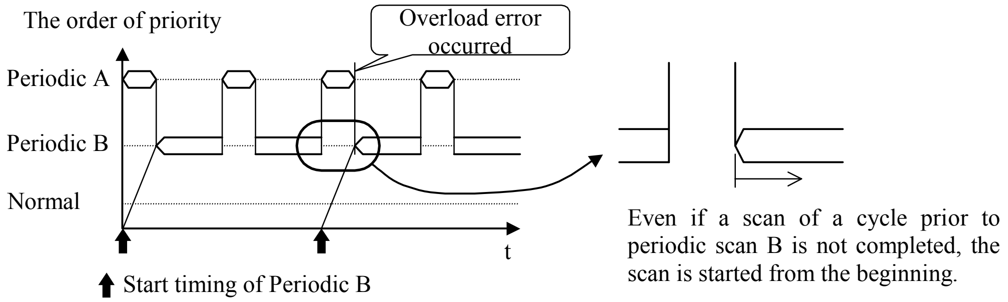

The order of priority
Overload error occurred
Periodic A
Periodic B
Normal
t
Start timing of Periodic B
Even if a scan of a cycle prior to periodic scan B is not completed, the scan is started from the beginning.

Figure 3.8 Continuation of operation at overload error in periodic scan

Notice

The normal scan is not executed if the periodic scan is set to the continuation of operation at overload error. Overload error will occur in the normal scan if only the periodic scan is set to the continuation of operation. And since scan END processing is not executed, an external input and output is not refreshed.

If you want to continue an operation of the periodic scan, also the motion when overload error has occurred in the normal scan must be set to the continuation of operation. If the external input and output is used, a command to refresh the external input and output should be described in the periodic scan. Additionally, care must be taken because a change during RUN is not executed when the periodic scan is running as the continuation of operation.

(4) How to fix cycle of periodic scan

If two or more periodic scans are used, there are cases where the periodic scan whose priority is low (cycle is long) is not started at fixed intervals according to specifying the cycle.

When two or more periodic scans are used, the timing of start will be stabilized if a cycle is fixed by the multiple of the shortest cycle.

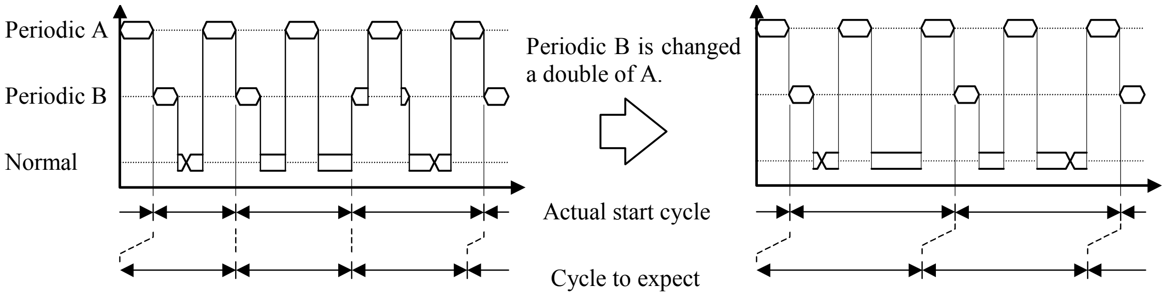


Figure 3.9 Selection of cycle for periodic scan

3.4 Subroutine

(1) Definition and motion of subroutine

Subroutine means the processing group in which processing of the purpose is packed.

Subroutine is a program surrounded by SB n command (n is subroutine number) and RTS command. When a subroutine is called, commands are executed from SB to RTS in sequence, and then it returns to the called point. Subroutine can call either a normal scan or a periodic scan, first. (Subroutine with the same number can also be called.)

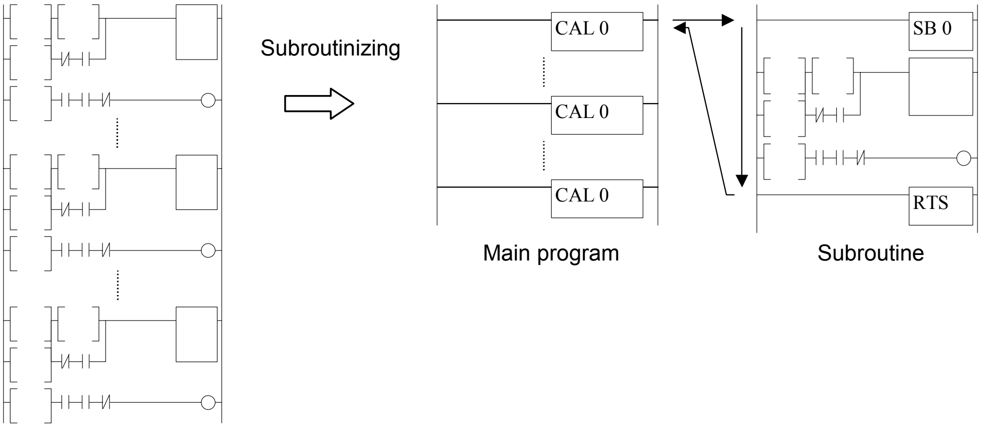


Figure 3.10 Subroutinizing of main program

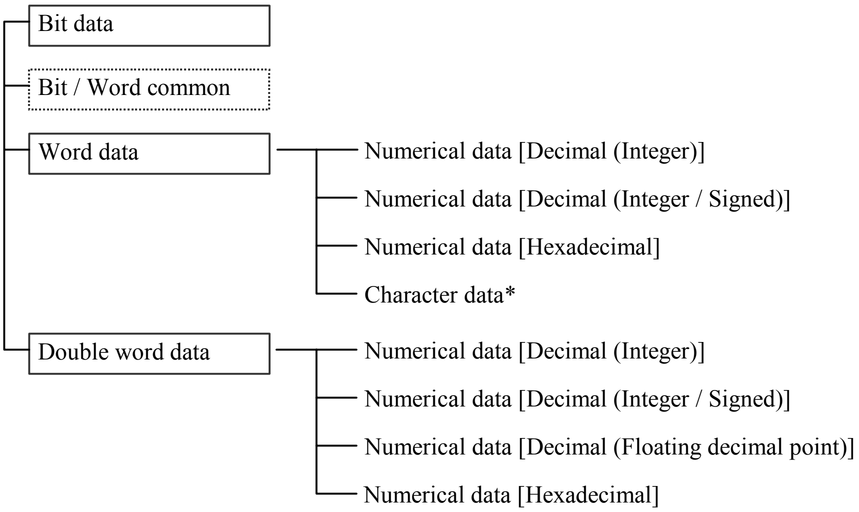
(2) Merit of subroutine

Subroutinizing has some merits as follows.

- ❖ User program which is easy to see.
- ❖ Standardization is possible as considering one function one package. (Diversion to other system is easy.)
- ❖ Modification is easy.
- ❖ Division of a word to crate user program is possible.
- ❖ It is possible to reduce the volume of user program.

3.5 How to specify data

Data used in user program of EHV-CPU is shown below.


Bit data
Bit / Word common
Word data
Numerical data [Decimal (Integer)]
Numerical data [Decimal (Integer / Signed)]
Numerical data [Hexadecimal]
Character data*
Double word data
Numerical data [Decimal (Integer)]
Numerical data [Decimal (Integer / Signed)]
Numerical data [Decimal (Floating decimal point)]
Numerical data [Hexadecimal]

* Treating is different from other word data. See the explanation of character data for details.

Figure 3.11 Type of data

(1) Bit data

Bit data is a data to process ON and OFF information with one bit unit. Bit data has three types as follows. Bit data has three types as follows.

i) Bit internal output

Data (Internal output) to use an area for bit only.

ii) Common internal output for Bit and Word (External output)

Data (internal output, external output) with which the data of bit and word uses the same area. Word data is configured with information of a bit data 16 point.

iii) Bit specification of Word internal output

Word internal output is a word access basically in order to use an are for word only but any one bit in word internal output can be handled if specified as follows.

Bit specification of Word data : Word I/O No. . Bit No. (Bit No. “0” - ”F”)

ex.) : Specify the 10th bit in WR100. WR100.A

Reference

When bit in word data is handled, the word data is processed once inside CPU module. Therefore, compared with the internal output of bit or the common internal output of bit and word, a speed to access is slow.

(2) Word data

Word data is configured with 16 bits and handled in the word units.

Word data can process data to store as a signed integer and a character string by adding extension behind an address of word data.

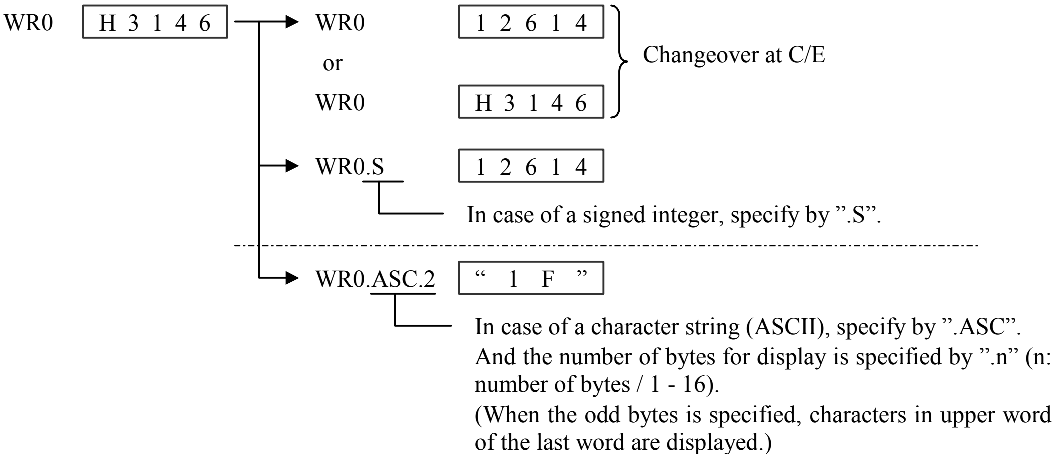

WR0
H 3 1 4 6
WR0
1 2 6 1 4
or
WR0
H 3 1 4 6
Changeover at C/E
WR0.S
1 2 6 1 4
In case of a signed integer, specify by ”.S”.
WR0.ASC.2
“ 1 F ”
In case of a character string (ASCII), specify by ”.ASC”.
And the number of bytes for display is specified by ”.n” (n: number of bytes / 1 - 16).
(When the odd bytes is specified, characters in upper word of the last word are displayed.)

Figure 3.12 Display of word data

Notice Specification of character string

Word I/O No. ASC. n is used when character string data is stored in several words. But when 3 bytes or more data are specified, data for read and write is as follows.

WR0.ASC.9 = “Hello□EHV”

⇩

WR0	“H”	“e”		6 8	6 5
WR1	“l”	“l”		6 C	6 C
WR2	“o”	“ ”	=	6 F	2 0
WR3	“E”	“H”		4 5	4 8
WR4	“V”			5 6	0 0

Data is stored in order from the specified word I/O No. in order of upper bytes and lower bytes. If the number of bytes is an odd number, Null (H00) will be stored in lower byte of the last word I/O.

(3) Double word data

Double word data is configured with 2 words / 32 bits.

Double word data can process data to store as a signed integer and a floating point (single accuracy) by adding extension behind an address of double word data.

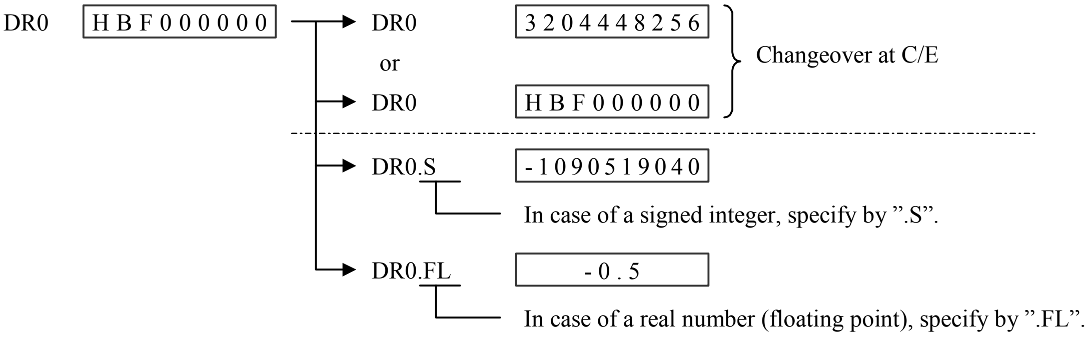


Figure 3.13 Display of Double word data

Reference Floating point format

Data of the floating point command uses a floating point of single accuracy based on IEEE 754.

An internal expression method for a floating point of single accuracy based on IEEE 754 is described below.

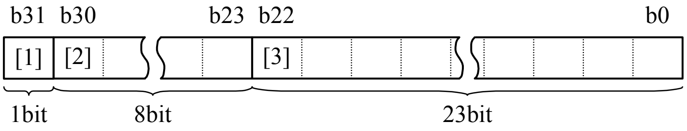


Figure 3.14 Floating point format

[1] Sign section 0…Positive, 1…Negative

[2] Exponential section

Exponent (E)	Numerical value to be 2 to a power (E')
FF	Show an overflowed value
FE	127
↓	↓
80	1
7F	0
7E	-1
↓	↓
01	-126
00	Handle as 0

[3] Mantissa section

Mantissa (M)	Numerical value to be mantissa (M')
7FFFFF	$(1.11\cdots11)_2$
7FFFFE	$(1.11\cdots10)_2$
↓	↓
1	$(1.00\cdots01)_2$
0	$(1.00\cdots00)_2$

■ Expression of numerical formulas

Floating point (F) can be expressed the following numerical formulas using signs mentioned above (S), exponent section(E) and mantissa section (M).

$(F) = (-1)^S \times (1 + M \times 2^{-23}) \times 2^{E-7FH} = (-1)^S \times M' \times 2^{E'}$

■ Range to be able to express by floating point

Hexadecimal expression		Floating point expression	Remarks
Upper word	Lower word		
H7F7F	HFFFF	$+3.402823\cdots\times10^{38}$	Maximum value
H0080	H0000	$+1.175494\cdots\times10^{-38}$	Value whose absolute value is minimum in positive numbers.
↓		↓	Handled as 0 during this.
H8080	H0000	$-1.175494\cdots\times10^{-38}$	Value whose absolute value is minimum in negative numbers.
HFF7F	HFFFF	$-3.402823\cdots\times10^{38}$	Minimum value

3.5.1 External input and output

External input represents it the sign X and external output represents it the sing Y.

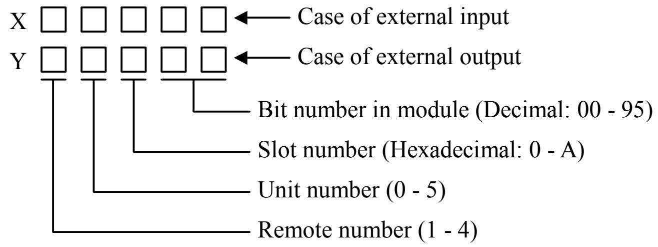
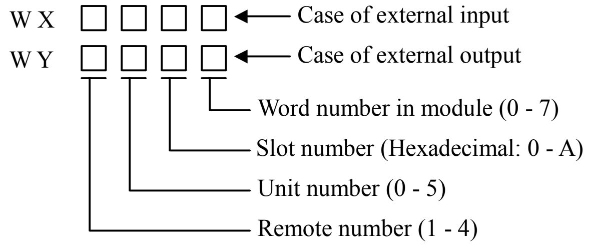
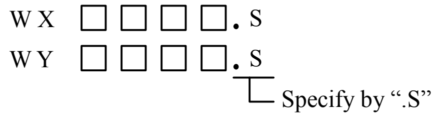
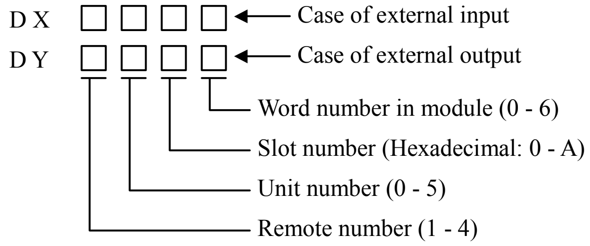
Both the bit I/O and the word I/O can fix the only peerless number depending on the position where the module is mounted.

Table 3.1 List of classification and data types for external input and output

I/O classification	Input and output classification	Data types	Remarks
X	External input	Bit types	Support to signals on each terminal stand Notice: decimal (X0,1,2,..,9,10,..,15,16,17,..,95)
WX		Word types (16 points)	Batch processing of data 0 to 15 Guarantee of simultaneity of 16 points
DX		Double word types (32 points)	Batch expression of two word data Non guarantee of simultaneity of 32 points
Y	External output	Bit types	Support to signals on each terminal stand. Notice: decimal (Y0,1,2,..,9,10,..,15,16,17,..,95)
WY		Word types (16 points)	Batch processing of data 0 to 15 Guarantee of simultaneity of 16 points
DY		Double word types (32 points)	Batch expression of two word data Non guarantee of simultaneity of 32 points

I/O number of the external input output is represented according to a rule as follows.

Table 3.2 List of I/O number rule for external input and output

Data type	Rule of number
Bit types (Basic / Expansion)	 X □ □ □ □ □ ← Case of external input Y □ □ □ □ □ ← Case of external output Bit number in module (Decimal: 00 - 95) Slot number (Hexadecimal: 0 - A) Unit number (0 - 5) Remote number (1 - 4)
Word types	 WX □ □ □ □ ← Case of external input WY □ □ □ □ ← Case of external output Word number in module (0 - 7) Slot number (Hexadecimal: 0 - A) Unit number (0 - 5) Remote number (1 - 4) 
Double word types	 DX □ □ □ □ ← Case of external input DY □ □ □ □ ← Case of external output Word number in module (0 - 6) Slot number (Hexadecimal: 0 - A) Unit number (0 - 5) Remote number (1 - 4)

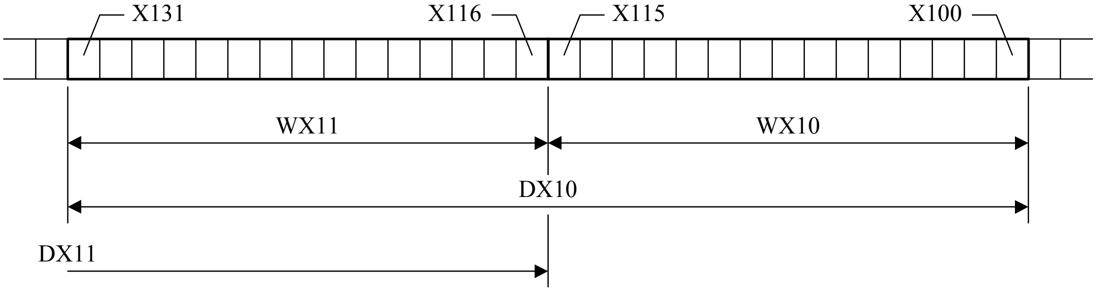
* EHV-CPU can consist of a basic unit and five expansion units. And since the maximum number of slots for the base unit is 11, the range of the input (X) is 0 to 5A95 and the range of the output (Y) is 0 to 5A95.

Reference

The remote number is represented with r, the unit number is represented with u and the slot number is represented with s in this document.

Word type of the external input and output is data which collected 16 applicable bit types and double word type is data which collected 32 applicable bit types.

(Example) Relation among DX10, WX10 and X100 to X115



3.5.2 Extension external input and output (Extension XY)

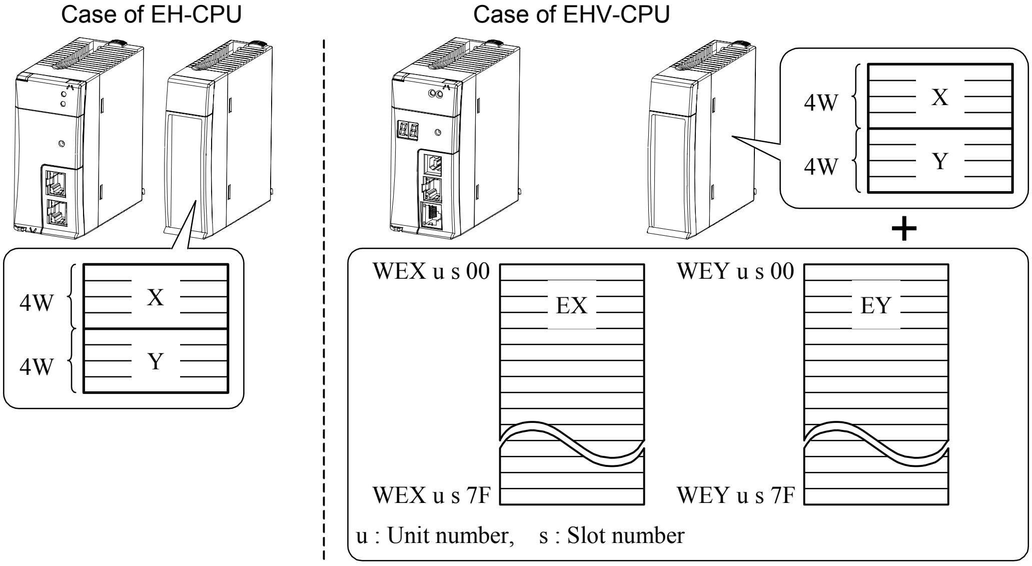
EHV-CPU has a special internal output which consists of 256 words in every slot.

This internal output is represented with the sign EX and EY. These are called Extension external input and output because there is X or Y in the sign. Although these are handled as same as the internal output, these can be used as an area to store information the external input and output and an area for module the same as usual X and Y depending on the I/O allocation.

An area of the extension external output is divided into an extension input area which consists of 128 words and an extension output area which consists of 128 words. I/O number can be fixed depending on the slot position. Furthermore, you need care because I/O number of the extension X and Y is allotted with hexadecimal, unlike normal X and Y allotted with decimal.

Table 3.3 List of classification and data types for extension external input and output

I/O classification	Input and output classification	Data types	Remarks
EX	External input	Bit types	Support to signal on each terminal stand, etc. Notice: Hexadecimal (EX0,1,2,..,9,A,..,F,10,11,..,7FF)
WEX		Word types (16 points)	Batch processing of data 0 to 15 Guarantee of simultaneity of 16 points
DEX		Double word types (32 points)	Batch expression of two word data Non guarantee of simultaneity of 32 points
EY	External output	Bit types	Support to signal on each terminal stand, etc. Notice: Hexadecimal (EY0,1,2,..,9,A,..,F,10,11,..,7FF)
WEY		Word types (16 points)	Batch processing of data 0 to 15 Guarantee of simultaneity of 16 points
DEY		Double word types (32 points)	Batch expression of two word data Non guarantee of simultaneity of 32 points



There is an area for modules of basic unit and expansion unit.

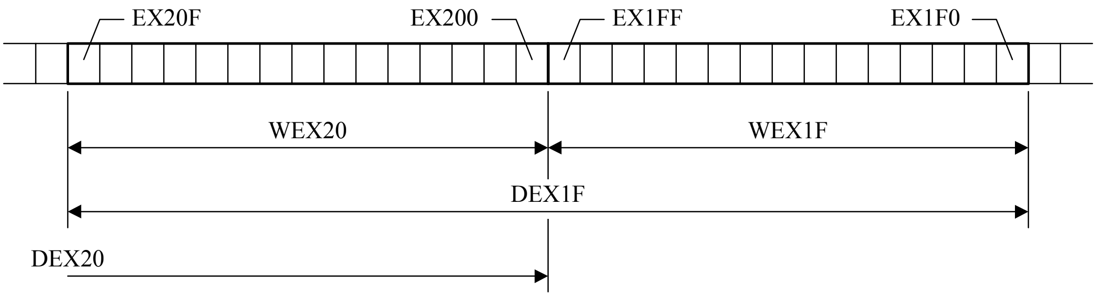
I/O number of the extension external input and output is represented according to a rule as follows.

Table 3.4 List of I/O number rule for extension external input and output

Data types	Rule of number
Bit types (Basic / Expansion)	EX □ □ □ □ □ ← Case of extension external input EY □ □ □ □ □ ← Case of extension external output Bit number in module (Hexadecimal: 000 - 7FF) Slot number (Hexadecimal: 0 - A) Unit number (0 - 5)
Word types	WEX □ □ □ □ ← Case of extension external input WEY □ □ □ □ ← Case of extension external output Word number in module (0 - 7F) Slot number (Hexadecimal: 0 - A) Unit number (0 - 5)
	WEX □ □ □ □. S WEY □ □ □ □. S Specify by ".S"
Double word types	DEX □ □ □ □ ← Case of extension external input DEY □ □ □ □ ← Case of extension external output Word number in module (0 - 7E) Slot number (Hexadecimal: 0 - A) Unit number (0 - 5)

Word type of the extension external input and output is data which collected 16 applicable bit types and double word type is data which collected 32 applicable bit types.

(Example) Relation among DEX1F, WEX1F and EX1F0 to EX1FF



■ Module to use the extension external input and output as the special use

The module to use the extension external input and output as an area for module and its use are shown in the following table.

Table 3.5 Usable module of extension external input and output, and its uses

No.	Type	Specification	Uses	
1	EH-ETH	Ethernet communication module	EX, WEX	[Command] Status area [Command] Display of module setting parameter
			EY, WEY	[Command] Control area [Command] Module setting parameter
2	EH-POS4	4 axes positioning module	EX, WEX	[CPU] Status area (14 words) [Command] Read data at command execution
			EY, WEY	[Command] Write data at command execution
3	EH-DBW	Any wire DB Interface	EX, WEX	[CPU] I/O data or status area* [Command] Sizing / Error address information display
			EY, WEY	[CPU] I/O data or control area*
4	EH-AXH8M, EH-TC8	Analog input module, Thermocouple input module	EX, WEX	[CPU] Overflow flag [Command] Module setting parameter
			EY, WEY	[Command] Module setting parameter
5	EH-AYH8M	Analog output module	EX, WEX	[CPU] Overflow flag [Command] Module setting information display
			EY, WEY	[Command] Module setting parameter

[Command] : Refreshed at command execution. [CPU] : Refresh CPU module automatically.

* The uses change according to an allocation to the module in EH-DBW.

Notice

EHV-CPU processes a module whose I/O allocation is the same in the same way (some area of the extension X and Y is refreshed automatically on the system program in EHV-VPU) as the above module. Although the area of the extension X and Y is handled as the internal output, care must be taken because the area will be overwritten on the refresh processing if the area to be refreshed is used on the system program.

Example : Case of using EH-AX8V

WEXus00 is refreshed automatically at every scan like ENDEH-AXH8M. If computations is stored in WEXus00 and a program to be referred on other processing is created, it may not be an expected operation since an indefinite value will be overwritten at the scan END.

3.5.3 Internal output

Internal output is a register which can be used on the user program.

Internal output has an area (R) for bit, an area (WR, WN) for word and a common area (M.WM) for bit and word.

And Internal output has an area (L/WL) to exchange data for other CPU using a link module.

Internal output has an area where user can access any time to and an area which is called a special internal output to user for a special purpose. The special internal output sets up the system and is used to display the state.

☞ See “Appendix 2 – Special internal output list” for more details.

Table 3.6 List of Internal output

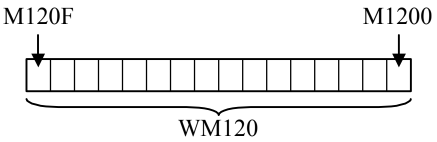
I/O classification		Points
Bit		1,984 points (R0 - R7BF)
Word (WR)		61,440 words (WR0 - WREFFF)
Word (WN)		131,072 words (WN0 - WN1FFFF)
Common Bit / Word (WM)		524,288 points 32,768 words (M0 – M7FFFF, WM0 - WM7FFF)
Special Internal Output	Bit	2,112 points (R7C0 - RFFF)
	Word	4,096 words (WRF000 - WRFFFF)
CPU link		16,384 points 1,024 words × 8 loops Link system 1 : L0 - L3FFF / WL0 - WL3FF Link system 2 : L10000 - L13FFF / WL1000 - WL13FF Link system 3 : L20000 - L23FFF / WL2000 - WL23FF Link system 4 : L30000 - L33FFF / WL3000 - WL33FF Link system 5 : L40000 - L43FFF / WL4000 - WL43FF Link system 6 : L50000 - L53FFF / WL5000 - WL53FF Link system 7 : L60000 - L63FFF / WL6000 - WL63FF Link system 8 : L70000 - L73FFF / WL7000 - WL73FF

I/O number of the internal output is represented according to a rule as follows.

Table 3.7 List of I/O number for Internal output (1 / 2)

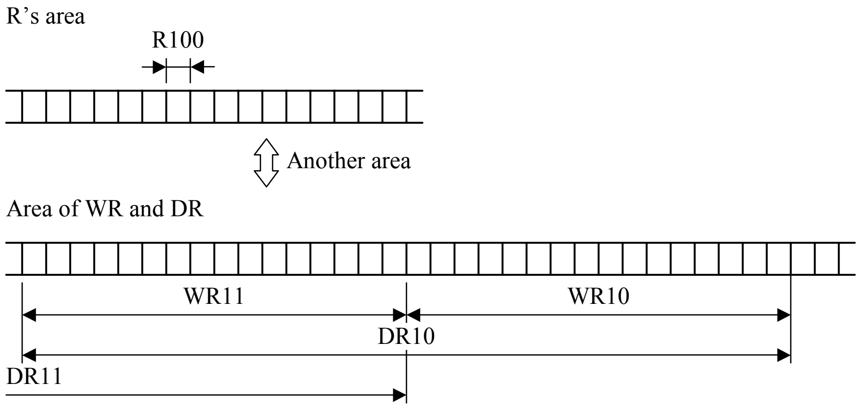
Data type	Number's rule	
Type for Bit		R □□□ Normal area H000 - H7BF Special area H7C0 - H7FF Represented both with hexadecimal
Type for Word	<Case of Word>	W R □□□□ Normal area H0000 - Special area HF000 - Represent both with hexadecimal W N □□□□□ Normal area H00000 - Represent with hexadecimal
	[Specify Bit]	W R □□□□.□ W N □□□□□.□ Specify by ".n" (n : Bit No. , 0 - F)
	[Signed integer]	W R □□□□. S W N □□□□□. S Specify by ".S"
	[Specify Character string]	W R □□□□. A S C . n W N □□□□□. A S C . n Specify by ".n" (n: Number of bytes 1- 32 [Decimal]) Specify by ".ASC"
	<Case of Double word>	D R □□□□ Normal area H0000 - Special area HF000 - Represent 2-word continuous WR Represent both with hexadecimal D N □□□□□ Normal area H0000 - Represent with hexadecimal
	[Signed integer]	D R □□□□. S D N □□□□□. S Specify by ".S"
	[Real number (Floating point)]	D R □□□□. FL D N □□□□□. FL Specify by ". FL"

Table 3.8 List of I/O number for Internal output (2 / 2)

Data type	Number's rule	
Common type –Bit and Word	<Case of Bit>	M □□□□□ L □□□□□ H00000 - / H0000 - Represent with hexadecimal
	<Case of Word>	W M □□□□ W L □□□□ H0000 - Represent with hexadecimal  * There is no bit specification in common type of bit and word.
	[Signed integer]	W M □□□□. S W L □□□□. S “.S”にて指定
	[Specification of Character string]	W M □□□□. A S C . n W L □□□□. A S C . n Specify by “.n” (n: Number of bytes1 – 32 [Decimal]) Specify by “.ASC”
	< Case o double >	D M □□□□ D L □□□□ H0000 - Represent both with hexadecimal, Represent 2-word continuous WR
	[Signed integer]	D M □□□□. S D L □□□□. S Specify by “.S”
	[Real number (Floating point)]	D M □□□□. FL D L □□□□. FL Specify by “. FL”

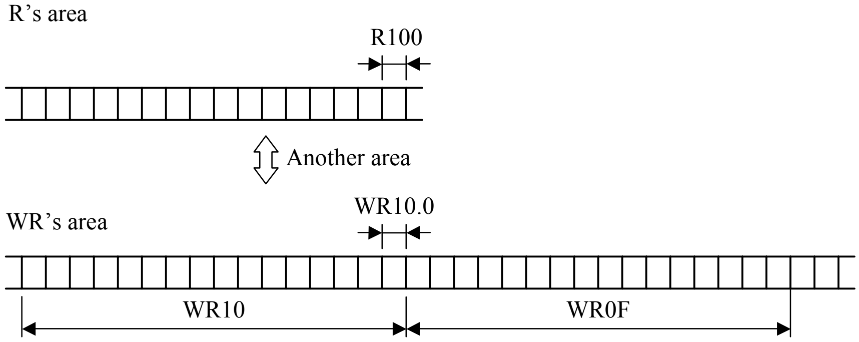
Internal output R is a different area from WR and DR.

(Example) Relation between R100 and WR10 and DR10.



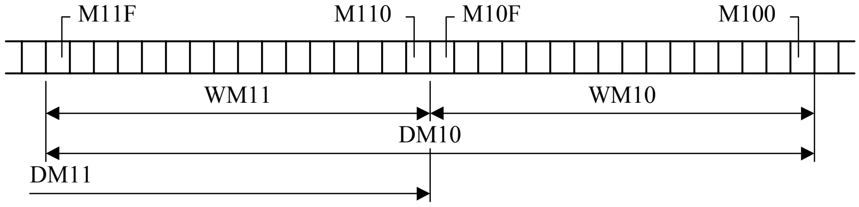
Notice

EHV-CPU can access by selecting any one bit from Word data.



Internal output M, WM and DM use the same area. (It is possible to handle a bit unit by word I/O.)

(Example) Relation between M100 and WM10 and DM10.



3.6 Program volume

Program volume is computed in a unit of “Step”.

Although user program is created by combining commands for various uses, the number of steps to be used for each command is different. (See the chapter for Command specification for the number of steps of each command.)

The sum total of the number of steps of all programs, such as normal scan, periodic scan and subroutine, is the number of steps written into a CPU module. Since the maximum number of steps of CPU module is fixed, a program which does not exceed it should be created

Table 3.9 Program volume

Items	EHV-CPU128	[Ref.] EH-CPU548
Program volume	128k steps	48k steps
Command size	40 bits / 1 step	32 bits / 1 step
Comment volume	1M bytes	(No function of commend storage)

Reference

The number of steps does not include all kinds of comment described on user program.

Although EHV-CPU memorizes not only a user program but also comments in a backup memory in CPU module, comments are controlled in “Comment volume”. When creating a user program, be careful not to exceed the maximum of comment volume about the comment also.

Chapter 4 Procedure to create User program

4.1 A flow to create User program

A procedure to create user programs is shown in the following diagram.

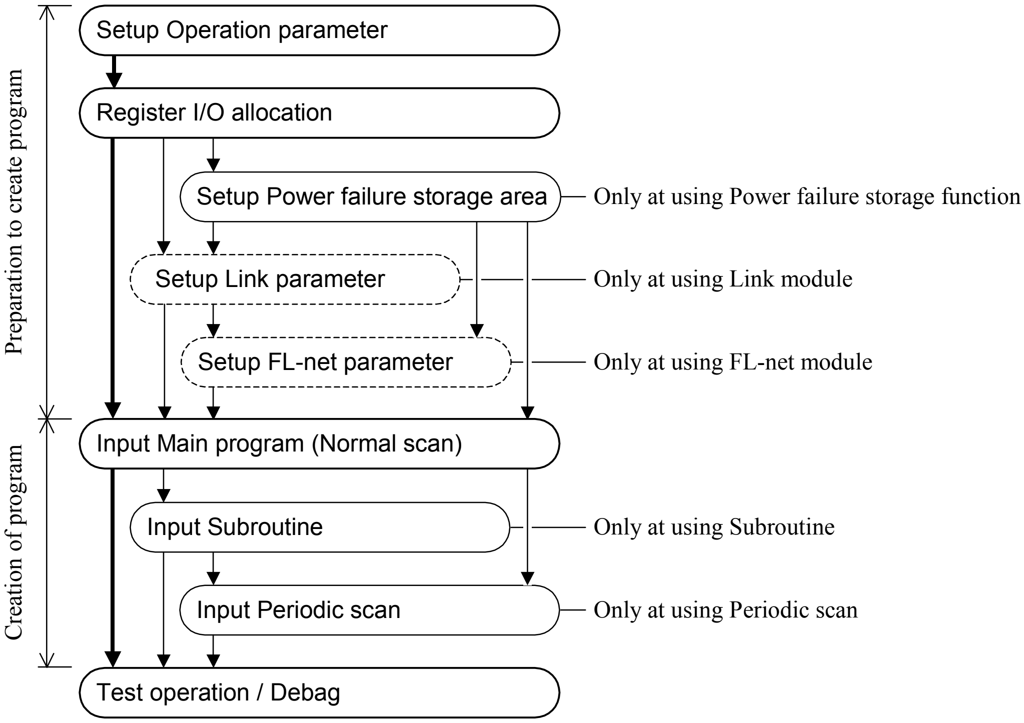

Preparation to create program
Creation of program
Setup Operation parameter
Register I/O allocation
Setup Power failure storage area
Only at using Power failure storage function
Setup Link parameter
Only at using Link module
Setup FL-net parameter
Only at using FL-net module
Input Main program (Normal scan)
Input Subroutine
Only at using Subroutine
Input Periodic scan
Only at using Periodic scan
Test operation / Debag

Figure 4.1 A flow to create user program

A basic user program is created in the following procedure.

Setup Operation parameter ➔ Register I/O allocation ➔ Input Normal scan ➔ Test Operation

Especially I/O allocation and normal scan input are indispensable. (The program consisting of only subroutine and periodic scan cannot be created.) Since a default value is set to the operation parameter when creating a new program, the input is not required when not changing the operation parameter from the default value.

4.2 Preparation to create program

The matter to set before creating program parts is shown below.

(1) Operation parameter

Parameters to set as an operation parameter are shown in the following list.

Table 4.1 List of Operation parameters

Item	Description		Default
Operation definition input	Bit to use as the operation definition input is specified from the external output (x) or the bit internal output (R, L, M). ☞ See "2.2 – Operation and Stop" for the function of operation definition input.		None
System processing time	Time to secure for a system program processing is set up. ☞ See "2.2.4 – Processing of system" for details of the system processing time.		1 ms
Overload checking time	Time to occur overload error in normal scan is set up. ☞ See "3.2 – Normal scan" for details of overload error		10 ms
Operation mode of CPU at error	At a disagreement of I/O allocation	Permission or Ban for Run at a disagreement of I/O allocation is specified. ☞ See the following page for details.	Impossible Run
	At a occurrence of remote error	Permission or Ban for Run is specified while error has occurred in remote module.	Impossible Run
	At a occurrence of overload error in normal scan	Continuation or Stop for Run is specified when overload error has occurred in normal scan. ☞ See "3.2 – Normal scan" for details.	Impossible Run
	At a occurrence of overload error in periodic scan	Continuation or Stop for Run is specified when overload error has occurred in periodic scan. ☞ See "3.3 – Periodic scan" for details.	Impossible Run
Remote transmission mode	At a disagreement of I/O allocation	Remote error or not is specified at a disagreement of I/O allocation in remote module.	Impossible transmission
	The slave station error of remote	Remote error or not is specified while error has occurred in the slave station of remote.	Impossible transmission
Error display	Level of display	Level of error display is set up. ☞ See "Appendix 1 – Self check" for details.	Display all
	At a occurrence of caution error	At a occurrence of errors for battery, power failure and back-up memory, it is selected where to display these errors on the 7-segment and ERR LED. ☞ See "Appendix 1 – Self check" for details.	Display

Reference Error display

EHV-CPU is always performing error detection. If error display is not set, factors of detected error are displayed on the 7-segment regardless of error level.

A setup of the level of error display is useful to display the error factor on the 7-segment only when serious errors, such as errors that a system of CPU module and a user program stopped, have occurred.

(2) II/O allocation

There are two methods for inputting I/O allocation as follows.

i) One slot of prearranged module for mounting is specified at a time on a setting screen for I/O allocation.

ii) I/O allocation list is created with “Function for read mounting I/O”.

EHV-CPU has a function for reading mounted module type from CPU module in EHV-CPU.

When all modules to use are arranged, if the mounting I/O is read with connecting a programming tool with the CPU module which modules have been mounted on the setting screen for I/O allocation, the I/O allocation’s list is created automatically as allocation information of mounted modules is read.

Notice

I/O allocation information is controlled by symbols, such as “X16” and “Y32”. When the function for reading the mounting I/O is used, the corresponded symbol to the module is read. (The module type is not read.)

And although I/O points (or the number of words) and the numbed of I/O allocation symbol are matched basically, there are exceptions, such as input and output of 8 points (8 points input ➔ “X16” and 8 points output ➔ “Y16”).

See the application manual for I/O allocation symbols for each module.

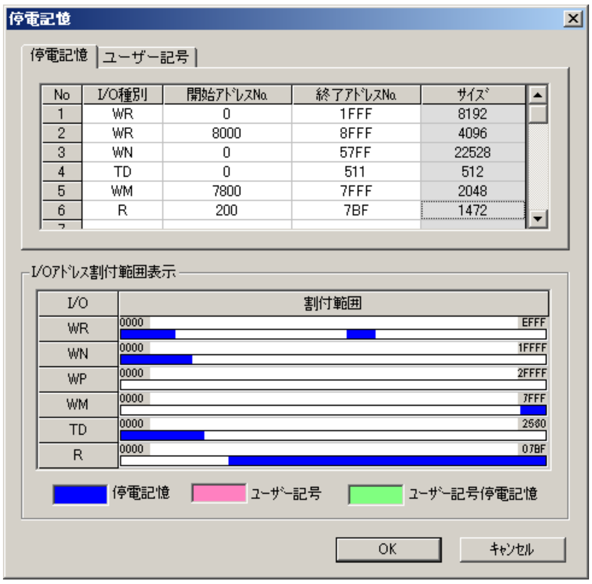
Reference

When checking operation of the program while all the systems are not arranged, please switch operation of an operation parameter under disagreement of I/O allocation to “RUN”. Even if I/O information written in from the programming tool and the mounted I/O are not matched, stop by error can be prevented by this setting.

Further, the system is actually worked, please return operation of the operation parameter under disagreement of I/O allocation to “Operation impossible”.

(3) Setting for Power failure storage area

In order to retain data even if the power supply of PLC is turned off, I/O area to retain is specified to the power failure storage area. The power failure storage area can specify 16 areas at the maximum in any range for the internal output and the timer. Also several areas for the same type of internal output can be specified.



No	I/O種別	開始ｱﾄﾞﾚｽ№	終了ｱﾄﾞﾚｽ№	ｻｲｽﾞ
1	WR	0	1FFF	8192
2	WR	8000	8FFF	4096
3	WN	0	57FF	22528
4	TD	0	511	512
5	WM	7800	7FFF	2048
6	R	200	7BF	1472

Figure 4.2 Setting for power failure storage area

(4) Setting for Link parameter

Since link parameter is a parameter to operate a link module, link parameter needs a setting for link module each.

Table 4.2 List of Link parameter

Items	Description	Default
Sending area Yes or No	When its own station's sending are is used, please check the mark When all link areas are used as a receiving area (other station's data area) , please remove the check mark.	None
Leading allocation No. End allocation No.	Setting the leading address and the end address of a sending area of its own station. (Input is impossible if the check box of the sending area is not checked.)	—
Clear for Operation Start / Stop	Data in link area is not usually cleared by a start and a stop of CPU module. When CPU module starts or stops an operation, please check the mark to clear data in link area.	No cleared

Reference

Although FL-net module is a kind of link module, a setting of link parameter is unnecessary since FL-net module is set on the setting screen for FL-net.

(5) Setting for FL-net parameter

Necessary parameters to operate FL-net module are set. See the manual for FL-net module for details of parameter set here.

4.3 Description of basic program

Programs are combined commands with particular functions and created. The command can be divided into six classes according to the processing to be executed.

Table 4.3 Classification of commands

No.	Classification	Description
1	Basic command	Operate bit data and word data by using a symbol peculiar to the ladder program. The basic command includes a timer and a counter.
2	Arithmetic command (Describe in a processing box)	Compare the substitution, the four basic operations, the logic operation and data.
3	Application command (Describe in a processing box)	Unite the processing which can not be realized unless combining some basic commands and arithmetic commands.
4	Control command (Describe in a processing box)	Define the end of program and change an order to execute programs, such as jump, repeat and subroutine.
5	CPU serial communication command (Describe in a processing box)	Control the serial communication port of CPU. Used when communicating data with external devices. Used when communicating data with external devices.
6	High-functional module command (Describe in a processing box)	Deliver data with high-functional module. Used when communicating data with external devices through module, and when setting parameter to module, and when reading the status.

(1) Basic circuit structure

The maximum unit of user program is "command". The command is described making lines on both sides, which are called BUS. This is called "circuit". The circuit has a rule which describes conditions in the left side and describes output (coil) or processing (processing box) in the right end.

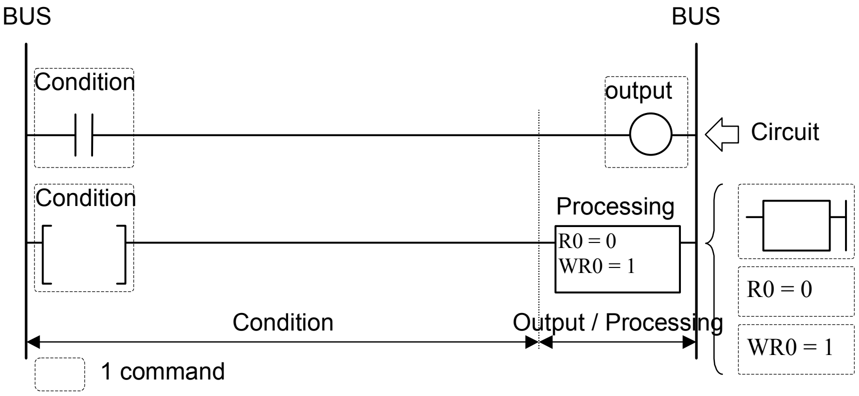


Figure 4.3 Basic circuit of User program

When all conditions at the left side in the circuit have been complete, the coil turns ON and commands in the processing box are executed. When conditions have not been complete, the coil turns OFF and commands in the processing box are not executed.

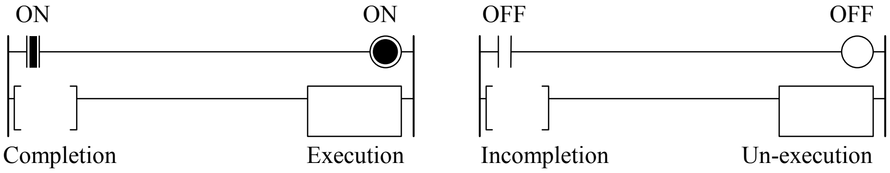


Figure 4.4 Operation of Basic circuit

Reference

A coil when conditions are not complete executes processing to turn the coil off. Since the processing box is not performed when conditions are not complete, the processing is different from the coil.

(2) Order to execute commands

Programs are executed in order from the left to the right and from the top to the bottom. When all conditions are complete in one circuit, the output is turned ON, and when not complete, the output is turned OFF. Similarly, the processing is executed when all conditions are complete, and it is not executed when not complete.

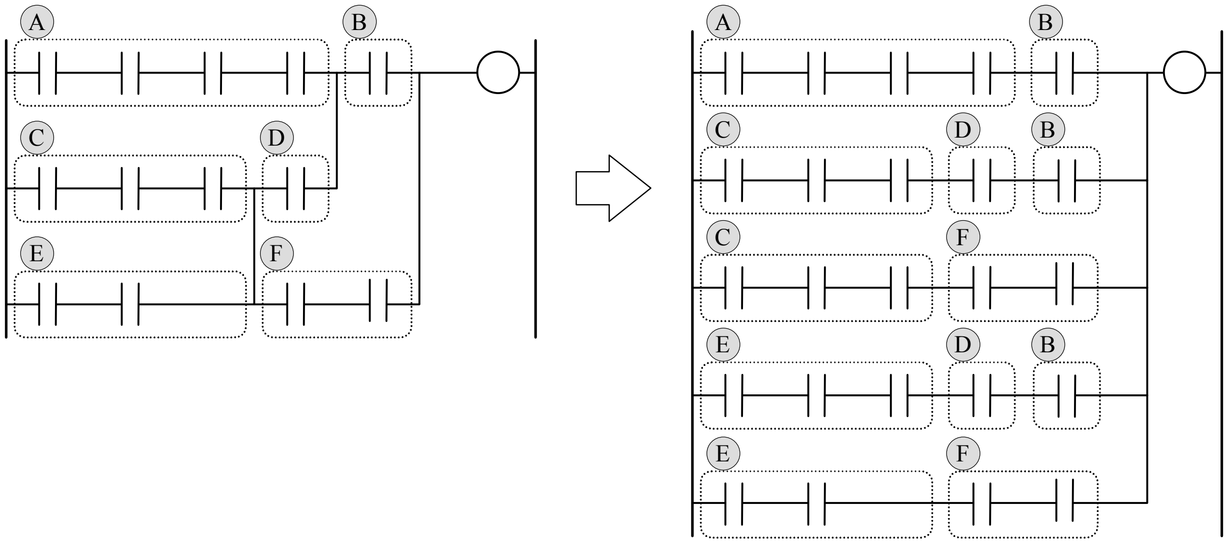


Figure 4.5 The view of OR connecting circuit

In fig.4.3, if the circuit on the left decomposed into each block follows the rule that a program is executed in order from the left to the right and from the top to the bottom, the circuit on the right will be made.

Reference

It is impossible to describe a contact point or a comparison box on the right side than the coil or the processing box.

And a route in fig.4.3, which is A ➔ D ➔ F ➔ a coil, is called the sneak circuit but this circuit is invalid since this is against the rule that a program is executed in order from the left to the right.

(3) Describable range in one circuit

It is possible to describe 11 contact points and 32 coils in one circuit as shown below. And using the wrap marker, it is possible to describe a circuit with 321 contact points and one coil within 32 lines.

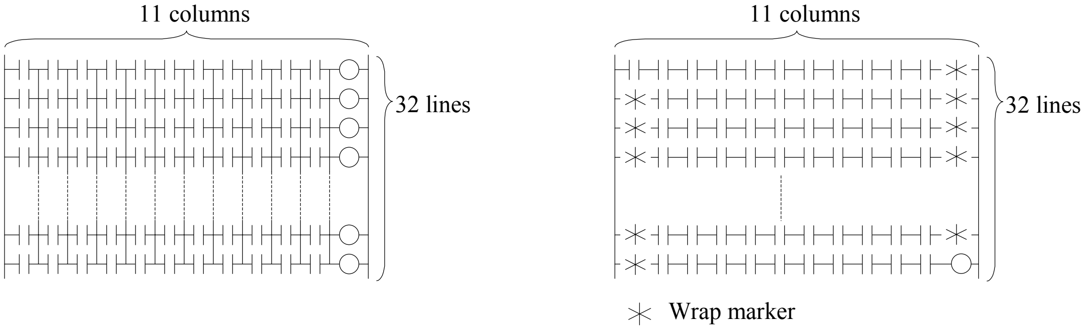


Figure 4.6 Describable range in one circuit

Using width in three contact points, it is possible to describe one comparison box. The comparison box can be considered as the a-contact point which turns ON when conditions in the box are complete.

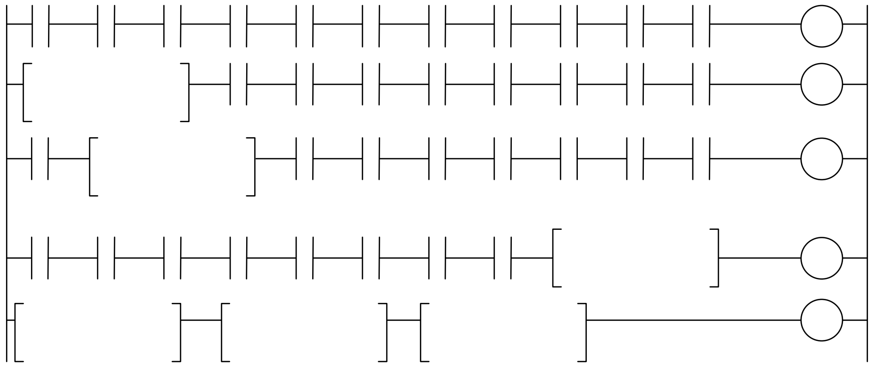


Figure 4.7 Description of Comparison box

The processing box uses width in 2 contact points and one coil. Commands except the basic command are described in the processing box. It is possible to describe 32 commands at the maximum in one processing box.

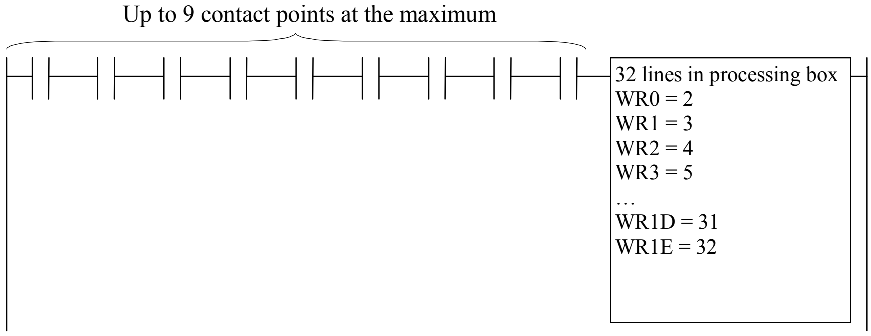

Up to 9 contact points at the maximum
32 lines in processing box
WR0 = 2
WR1 = 3
WR2 = 4
WR3 = 5
…
WR1D = 31
WR1E = 32

Figure 4.8 Description of Processing box

The processing box and the coil can be connected by OR in EHV-CPU.

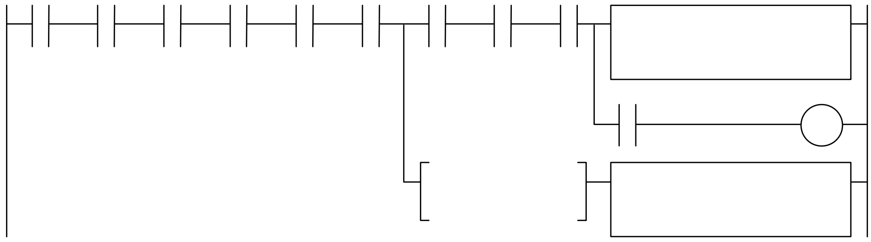
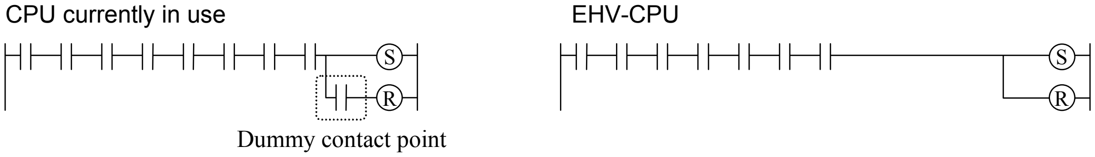


Figure 4.9 OR connection in Processing box and Coil

Reference

When connecting a set coil or a reset coil with OR, setting of a dummy contact point is required for CPU currently in use but it is not required for EHV-CPU.



4.4 Instructions on creating user program

(1) Timer

■ Updating a progress value

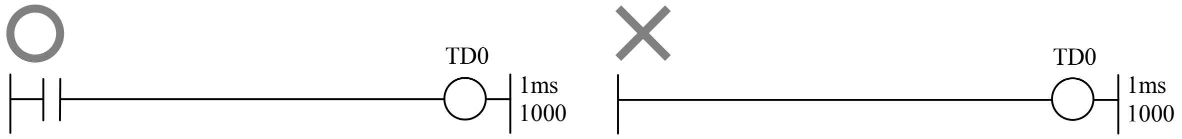
A progress value of a timer is updated when a timer command is executed. Therefore, there are cases where the timer does not turn ON correctly if the timer command is not scanned by a program which uses JMP command and the master control (MCS).

(If the time not to scan the timer command exceeds the value (= the time base × 65535), the timer does not turn ON correctly.)

And the timer progress value remains unchanged until the timer command is executed.

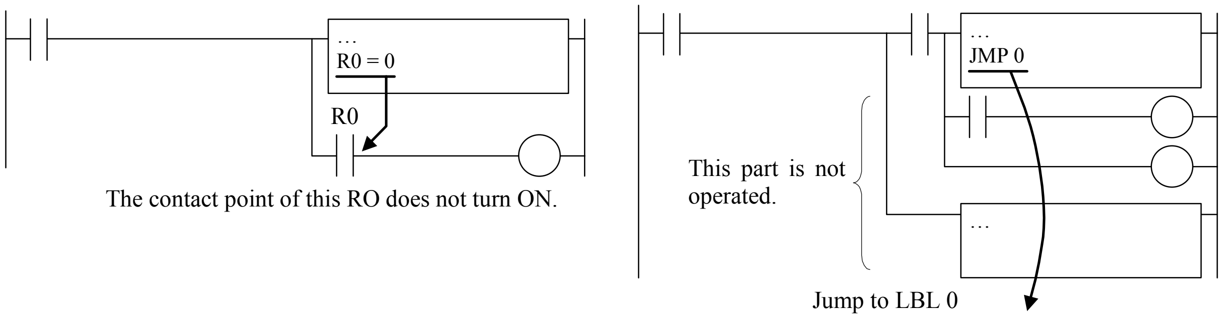
■ Startup condition for Timer

The timer command cannot connect from the bus directly. A condition is always needed in front of the timer command.



(2) OR connection for Coil and Processing box

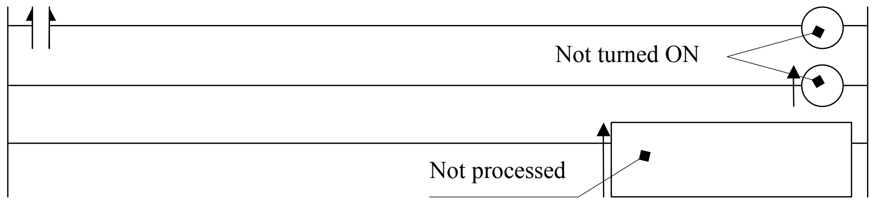
When the coil and the processing box are connected by OR, since CPU module is scanned in order from top, there are cases where the operation in the lower part of OR connection is not executed by the operation in the upper part of OR connection.



…
R0 = 0
R0
The contact point of this RO does not turn ON.
…
JMP 0
This part is not operated.
…
Jump to LBL 0

(3) Edge, Coil with edge and Processing box with edge

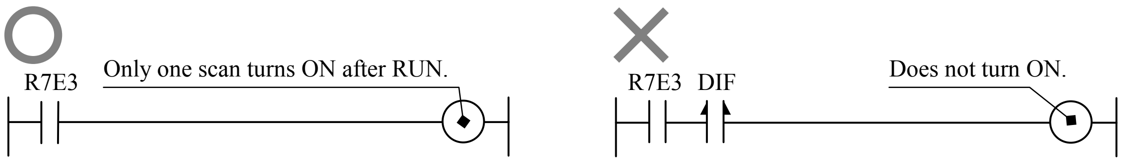
An edge, a coil with edge and a processing box with edge commands (rising and falling) detect a change in the status of the condition on the left side than the edge command. Therefore, the condition is needed on the left side than the edge command.



Notice

The special internal output (R7E3), which one scan turns ON after RUN, starts with ON at the start of RUN. (Only changes from ON to OFF.) Therefore, R7E3 cannot be used as a condition for the rising edge.

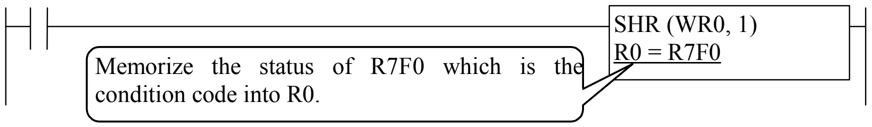
Further, since only one scan turns ON after RUN in R7E3, the edge command is not required.



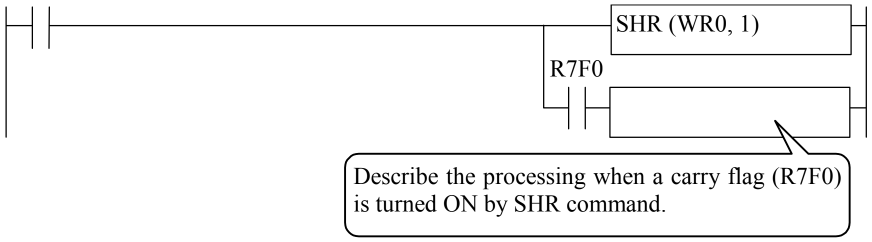
(4) Condition code

Many of commands use a bit internal output which is called the condition code. Since the condition code is used in each command in common, there are cases where the status changes after executing the command. Therefore, the following programs should be created when the condition code is referred.

❖ Be stored in another internal output just after the command is executed.


SHR (WR0, 1)
R0 = R7F0
Memorize the status of R7F0 which is the condition code into R0.

❖ The output is branched off under the same condition and the contact point of the condition code is put in front of a processing after branching off.


SHR (WR0, 1)
R7F0
Describe the processing when a carry flag (R7F0) is turned ON by SHR command.

(5) Floating point

The number of digit valid is finite in floating point. Therefore, an error arises between a computation result and a true value.

If the program (especially comparing in [= = (agreement)] or [< > (disagreement)]) compares a constant with a computation result using floating point in data type, the result expected according to an error might not be obtained.

When comparing the computation result in floating point, we recommend a decision by not agreement or disagreement but the range.

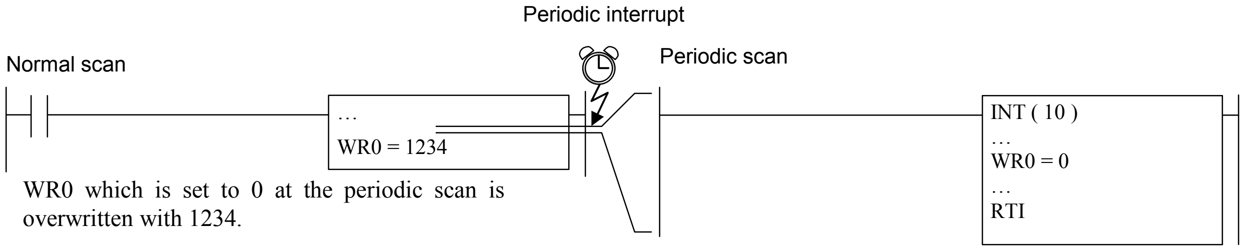
Reference

There are three kinds of errors in floating point as shown below.

Name	Description
Rounding error	Arise by deleting the lower digits by rounding down, rounding up and rounding off, to represent the operational resulting in the number of digits valid. Example) If 0.1 in decimal is converted to the binary number, it will become a recurring decimal. It is nearly 0.1 but it is not 0.1 within the finite number of digits valid.
Information omission	Arise since the small number is not reflected in the computation result if addition and subtraction of a very large number and a small number of the absolute value. Example) When 1234 and 0.0056 are added, the expected result is 1234.0056 but a numerical value with mantissa part with smaller exponential part is rounded down since a numerical value with large exponential part is a standard against which the operation is performed.
Digit omission	Arise since the number of digits valid reduces when finding the remainder between 2 absolute values which are almost equal. Example) When subtracting 1.23789 from 1.23456, the number of digits valid is 6 digits before the calculation, but the result is –0.00333 of which the number of digits is 3 digits.

(6) Periodic scan

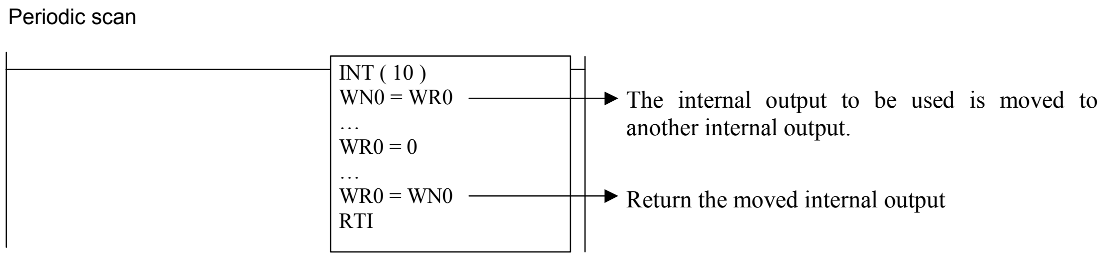
On the program which uses a periodic scan and a normal scan, or several periodic scans, if the same I/O is operated in the scan of which the order of priority is different, a value which is set at the scan of high priority might be disappeared.



WR0 which is set to 0 at the periodic scan is overwritten with 1234.

Some programming methods to keep data from disappearing are shown below.

- ❖ The same I/O is not used at the scan of which the order of priority is different
- ❖ I/O which is set at the scan of high priority is only referred at scans other than it.
- ❖ I/O used at the leading in the scan of high priority is stored in another I/O once, and it is returned when the scan is completed.



The internal output to be used is moved to another internal output.

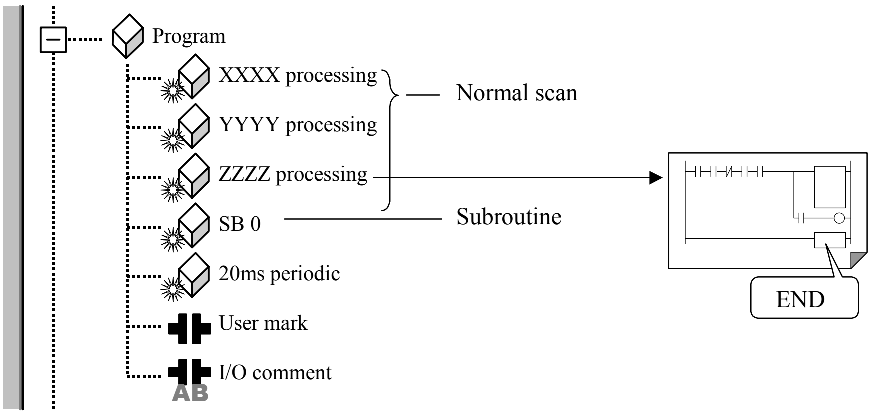
Return the moved internal output

And the external input and output are refreshed at the end of the normal scan. The latest input information should be referred and the I/O command should be used when the output is operated.

(7) Sheet division

When a subroutine and a periodic scan are used, the END command is required at the end of a normal scan.

Although it is possible to write on each sheet the subroutine and the periodic scan separately, the END command should be written at the end of the sheet for the normal scan on a project tree since the CPU module executes a sheet registered to the project tree in order from top.



Chapter 5 Command specification details

5.1 Command classification

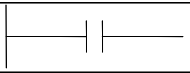
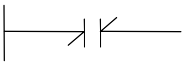
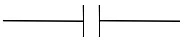
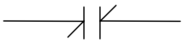
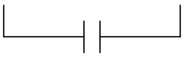
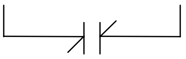
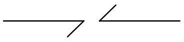
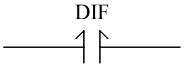
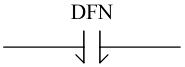
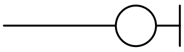
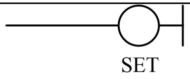
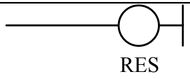
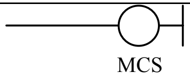
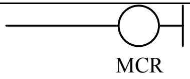
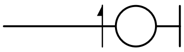
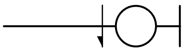
Usable commands in EHV-CPU can be classified as follows.

Table 5.1 Command classification table

No.	Command classification	Description	Types
1	Basic command	Sequence	25
		Timer / Counter	11
		Comparison box	18
2	Arithmetic command	Substitution expression (Array variable)	3
		Four arithmetical operations	16
		Logical operations	3
		Comparison expression	18
		Type conversion, Code	8
		Square root, Exponentiation	3
		Trigonometric function	12
		Exponent, Logarithm	3
3	Application command	Command support 命令補助	1
		Bit operation	4
		Shift / Rotate	14
		Character conversion	14
		Data operation	9
		Data search	3
		Exchange	2
		Transfer	4
		Decode, Encode	3
		I/O refresh	3
		PID control	3
		FIFO	3
		Communication support 通信補助	2
		Others	2
4	Control command	END, JMP, CAL, FOR, NEXT, RTS, RTI, LBL, SB, INT, CEND, CJMP	12
5	CPU serial port command	TRNS0, RECV0	2
6	High-functional module transfer command	Transmission command for EH-ID	2
		Transmission command for EH-SIO	1
		Error / Sizing information read for EH-DBW	1
		Control command for EH-POS4	4
		User message communication command for EH-FLN2	2
		Explicit message communication command for EH-RMD/IOCD	2
		General-purpose transfer command	2

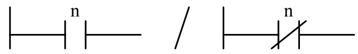
5.2 Command list

(1) Basic command (Sequence command)

No.	Ladder symbol	Command name	Processing	Page
1		Start of logical operations	Indicates the beginning of a-contact operation.	5-24
2		Start of NOT operation	Indicates the beginning of b-contact operation.	5-24
3		AND operation	Indicates a-contact series connection.	5-25
4		NAND operation	Indicates b-contact series connection.	5-25
5		OR operation	Indicates a-contact parallel connection.	5-26
6		NOR	Indicates b-contact parallel connection.	5-26
7		NOT operation	Reverses the operation result up to that time.	5-27
8		Rising edge detection	Indicates the rising detection of input.	5-28
9		Falling edge detection	Indicates the falling detection of input.	5-29
10		I/O output	Indicates the output coil.	5-30
11		I/O set	Indicates the set coil.	5-31
12		I/O reset	Indicates the reset coil.	5-31
13		Start of Master control	Indicates the set operation of master control.	5-32
14		Master uncontrol	Indicates the reset operation of master control.	5-32
15		Coil with edge (Rising)	Detects the rising of condition, and output is turned ON for only one scan.	5-33
16		Coil with edge (Falling)	Detects the falling of condition, and output is turned ON for only one scan.	5-34

[1] Basic commands

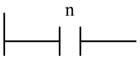
Name	Logical operation start (LD, LDI)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	Bit I/O	1	●	●	●	●	●
	Bit in Word	2					
	Extension XY	2					

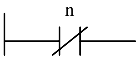
Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Bit I/O	0.02	–	–	
Bit in Word (.m)	0.04	–	–	
Extension XY (EX, EY)				

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	I/O No.	✓	✓	✓	✓	✓	✓										

Function



A start of the a-contact logical operation is represented. It is in a continuity state when an input is on. If the word I/O.m (m; bit No.) is specified, it is in a continuity state when an applicable bit is on.

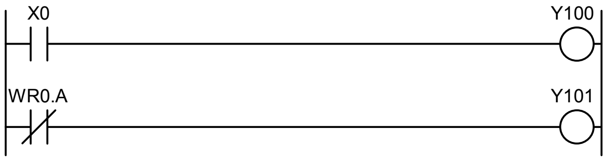


A start of the b-contact logical operation is represented. It is in a contituity state when an input is off. If the woer I/O.m is specified, it is in a contituity state when an applicable bit is off.

Cautionary notes

‘m’ specified by Bit in Word is valid from 0 through F.

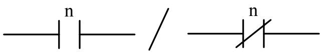
Program example



[Program description]

- When an input X0 is turn on, an output Y100 is turned on. When X0 is turned off, Y100 is turned off.
- When the Ath bit of an internal output WR0 is turned off, an output Y101 is turned on.
  When the Ath bit is turned on, Y101 is turned off.

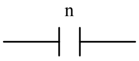
Name	Contact series connection (AND, ANI)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	Bit I/O	1	●	●	●	●	●
	Bit in Word	2					
	Extension XY	2					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Bit I/O	0.02	—	—	
Bit in Word (.m)	0.04	—	—	
Extension XY (EX, EY)				

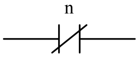
Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	I/O number	✓	✓	✓	✓	✓	✓										

Function



‘AND’ of the previous operation result and the a-contact operation is taken.

When the word I/O.m (m: bit No.) is specified, ‘AND’ of thr precious operation result and an applicable bit (a-contact) in Word is taken.



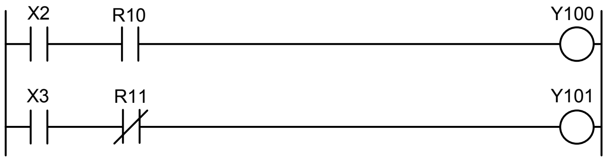
‘AND’ of the previous operation result and the b-contact operation is taken.

When the word I/O.m is specified, ‘AND’ of the previous operation result and an applicable bit (b-contact) in Word is taken.

Cautionary notes

‘m’ specified by Bit in Word is valid from 0 through F.

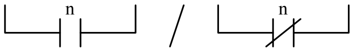
Program example



[Program description]

- When both an input X2 and R10 are on, an output Y100 is turned on. All other cases are turned off.
- When an input X3 is on and R11 is off, an output Y101 is turned on. All other cases are turned off.

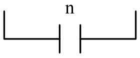
Name	Contact parallel connection

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	Bit I/O	2	●	●	●	●	●
	Bit in Word	3					
	Extension XY	3					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Bit I/O	0.04	–	–	
Bit in Word (.m) Extension XY (EX, EY)	0.06	–	–	

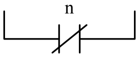
Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	I/O number	✓	✓	✓	✓	✓	✓										

機 能



‘OR’ of the previous operation result and the a-contact result is taken.

When the word I/O.m (m: bit No.) is specified, ‘OR’ of the previous operation result and an applicable bit (b-contact) in Word is taken.



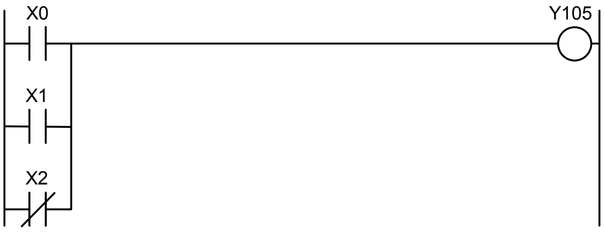
‘OR’ of the previous operation result and the b-contact operation is taken.

When the word I/O.m is specified, ‘OR’ of the previous operation result and an applicable bit (b-contact) in Word is taken.

Cautionary notes

‘m’ specified by Bit in Word is valid from 0 through F.

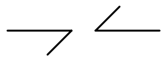
Program example



[Program description]

When X0 or X1 is on, or X2 is off, Y105 is turned on.

Name	Negation

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	—	2	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	0.04	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
—	No argument																

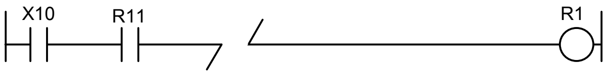
Function

The operation results obtained by then are reversed.

Cautionary notes

It is impossible to write the negation command into the top of the circuit.

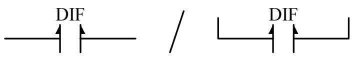
Program example



[Program description]

When both an input X10 and X11 are on, the operation is’0’ but it becomes ‘0’ because of the negation command. As a result, R1 is turned off. All other cases, R1 is turned on.

Name	Rising edge detection

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	AND DIF	1	●	●	●	●	●
	OR DIF	2					

Command processing time (µs)				Remarks
Average		Maximum		Useable to 512 maximum.
Condition	Time	Condition	Time	
AND DIF	0.04	—	—	
OR DIF	0.06	—	—	

Usable I/O		Bit: X	Bit: Y	Bit: EX, EY	Bit: R, L, M	Bit: TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU	Bit: WR, WN (.m)	Word: WX	Word: WY	Word: WEX, WEY	Word: WR, WL, WM, WN	Word: TC	Double word: DX	Double word: DY	Double word: DEX, DEY	Double word: DR, DL, DM	Constant
—	No argument																

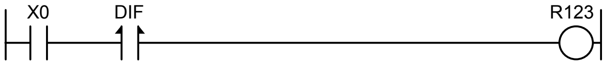
Function

The rising of an input signal is detected and the operation result for one scan only is retained.

Cautionary notes

- A programming tool assigns DIF number automatically.
- DIF cannot be used singly.
- DIF is a command to detect a change (0→1) of the operation result obtained by then.

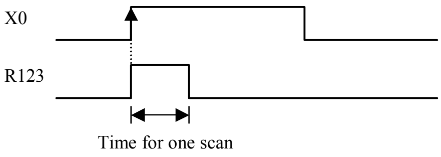
Program example



[Program description]

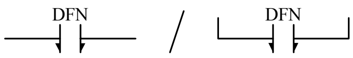
- R123 is turned on during one scan at the rising of X0.

Time chart



- When X is the b-contact, the program is the same meaning as a-contact DFN operation of X0.

Name	Falling edge detection

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	AND DFN	1	●	●	●	●	●
	OR DFN	2					

Command processing time (μs)				Remarks
Average		Maximum		Useable to 512 maximum.
Condition	Time	Condition	Time	
AND DFN	0.04	—	—	
OR DFN	0.06	—	—	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
—	No argument																	

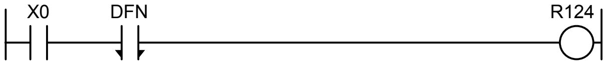
Function

The falling of an input signal is detected and the operation result for one scan only is retained.

Cautionary notes

- A programming tool assigns DIF number automatically.
- DFN cannot be used singly.
- DFN is a command to detect a change (1→0) of the operation result obtained by then.

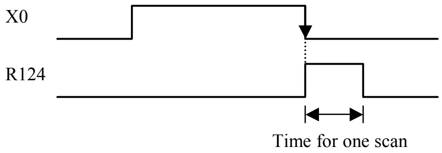
Program example



[Program description]

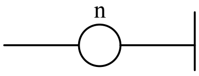
- R124 is turned on during on scan at the falling of X0.

Time chart



- When X0 is the b-contact, the program is the same meaning as a-contact DIF operation of X0.

Name	Output to Coil

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
 n	Bit I/O	1	●	●	●	●	●
	Bit in Word	2					
	Extension XY	2					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Bit I/O	0.04	—	—	
Bit in Word (.m) Extension XY (EX, EY)	0.06	—	—	

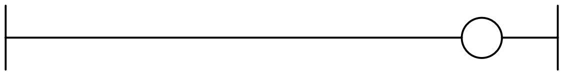
Usable I/O		Bit						Word					Double word				Constant
		X	Y	EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	I/O number		✓	✓	✓	✓	✓										

Function

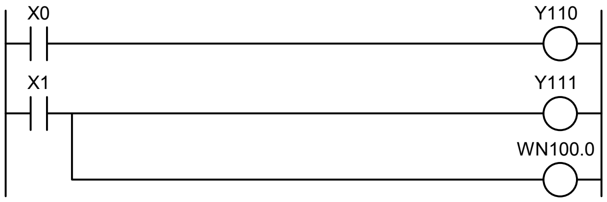
- When the operation result obtained by then is ‘1’, the coil is switched on.
- When the operation result obtained by then is ‘0’, the coil is switched off.

Cautionary notes

- ‘m’ specified by Bit in Word is valid from 0 through F.
- In case of the circuit such as a chart below, ‘1’ is added to number of steps shown in the above table.



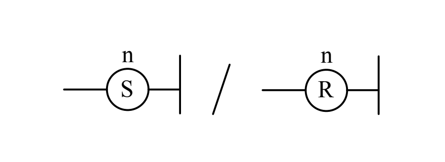
Program example



[Program description]

- When an input X0 is on, the operation is set to ‘1’ and Y110 is turned on.
- When an input X1 is on, the operation is set to ‘1’ and Y111 is turned on. Also the 0th bit of WN100 is set to ‘1’.

Name	Output Set / Reset to Coil

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	Bit I/O	1	●	●	●	●	●
	Bit in Word	2					
	Extension XY	2					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Bit I/O	0.04	–	–	
Bit in Word (.m) Extension XY (EX, EY)	0.06	–	–	

Usable I/O		Bit							Word					Double word				
		X	Y	EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	Constant
n	I/O number		✓	✓	✓			✓										

Function

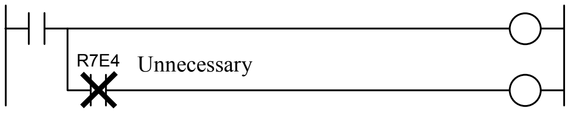
SET　n　When the operation result obtained by then is ‘1’, the device is switched on.

The device switched on is not turned off even if the operation is set to ‘0’.

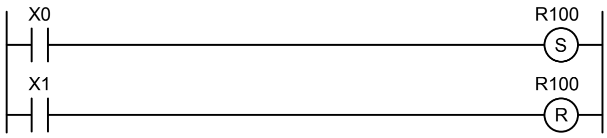
RES　n　When the operation result obtained by then is ‘1’, the device is switched off.

Cautionary notes

- ‘m’ specified by Bit in Word is valid from 0 through F.
- A dummy contact is necessary in front of the Set/Reset coil which made OR connection in models currently in use, but it is unnecessary in EHV-CPU.



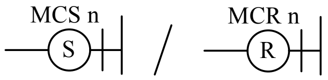
Program example



[Program description]

- When an input X0 is turned on, R100 is turned on. Even if X0 is turned off, R100 remains on.
- When an input X0 is turned on, R100 is turned off.
- If both inputs X0 and X1 were turned on, the command later on performed on the program has priority.

Name	Set / Reset of Master control

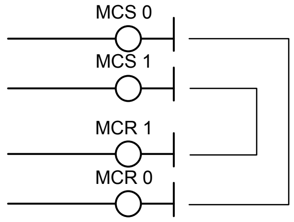
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
	MCS n	2	●	●	●	●	●
	MCR n	1					

Command processing time (μs)				Remarks
Average		Maximum		Useable No.0 to No.49 (decimal)
Condition	Time	Condition	Time	
MCS n	0.06	—	—	
MCR n	0.04			

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	Number																✓

Function

- An input in the circuit surrounded by Set (MCS n) and Reset (MCR n) of Master control is controlled. (‘AND’ operation is performed with each input and MCS.)
- The master control can be used up to eight layers.

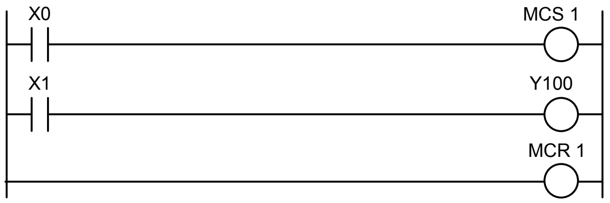


Up to eight layers are possible

Cautionary notes

MCS and MCR of the master control should be always used in pairs.

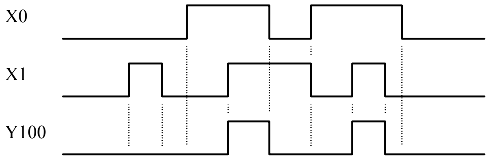
Program example



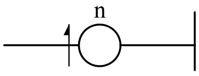
[Program description]

- Y100 is turned on or off in accordance with the state of X1 when X1 is on.
- Y100 is turned off independently of the state of X1 when X1 is off.

Time chart



Name	Coil with rising edge

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	Bit I/O	2	●	●	●	●	●
	Bit in Word	3					
	Extension XY	3					

Command processing time (μs)				Remarks
Average		Maximum		Useable to 1,024 maximum.
Condition	Time	Condition	Time	
Bit I/O	0.08	–	–	
Bit in Word (.m)	0.10	–	–	
Extension XY (EX, EY)				

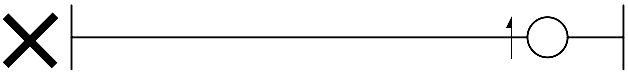
Usable I/O		Bit							Word					Double word				Constant
		X	Y	EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	I/O number		✓	✓	✓			✓										

Function

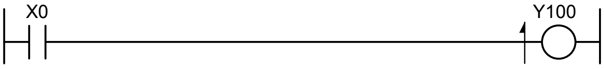
When the operation result obtained by then changes ‘0’ to ‘1’, the device is switched on during one scan.

Cautionary notes

- ‘m’ specified by Bit in Word is valid from 0 through F.
- It is impossible to write the circuit without any condition (the circuit of only a coil with edge).



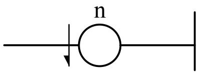
Program example



[Program description]

- Y100 is turned on during one scan at the rising of time X0 is turned on.
- If X0 is the b-contact, it is the same operation as the coil with a falling edge.

Name	Coil with falling edge

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
	Bit I/O	2	●	●	●	●	●
	Bit in Word	3					
	Extension XY	3					

Command processing time (μs)				Remarks
Average		Maximum		Useable to 1,024 maximum.
Condition	Time	Condition	Time	
Bit I/O	0.08	—	—	
Bit in Word (.m)	0.10	—	—	
Extension XY (EX, EY)				

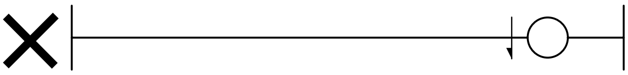
Usable I/O		Bit						Word					Double word				Constant
		X	Y	EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	I/O number		✓	✓	✓		✓										

Function

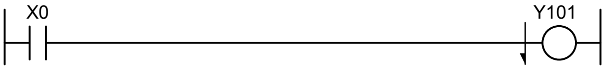
When the operation result obtained by then changes ‘1’ to ‘0’, the device is switched on during one scan.

Cautionary notes

- ‘m’ specified by Bit in Word is valid from 0 through F.
- It is impossible to write the circuit without any condition (the circuit of only a coil with edge).



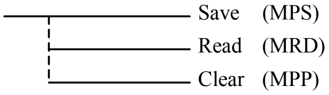
Program example



[Program description]

- Y101 is turned on during one scan at the falling of time X0 is turned off.
- If X0 is the b-contact, it is the same operation as the coil with a rising edge.

Name	Save / Read / Clear of Operation result (Branching of ladder)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	—	0	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	—	—	—	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
—	No argument																	

Function

- MPS saves the last operation result. (Push)
- MRD reads the result saved at MPS and continues the operation.
- MPP reads the last result saved at MPS and continues the operation. Then it clears the result after the operation. (Pull)

Name	Logical block series connection

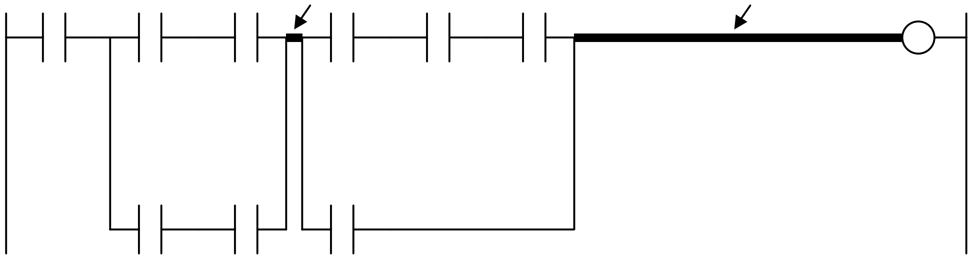
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
(See function column)	—	0	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	—	—	—	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
—	No argument																	

Function

This command is used when the logical operation block is made AND connection.



Name	Logical block parallel connection

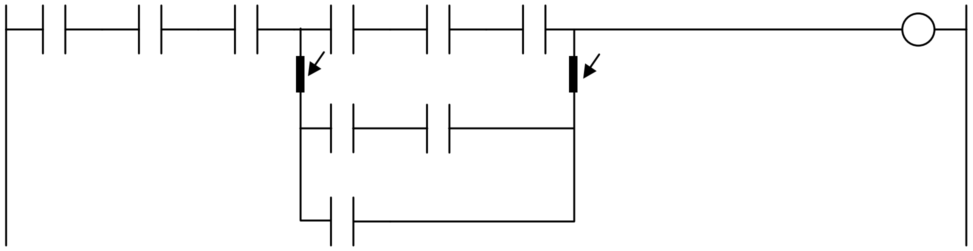
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
(See function column)	—	1	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	0.02	—	—	

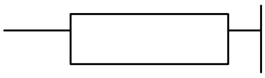
Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
—	No argument																

Function

This command is used when the logical operation block is made OR connection.



Name	Start and end of Processing box

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	—	3	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	0.06	—	—	

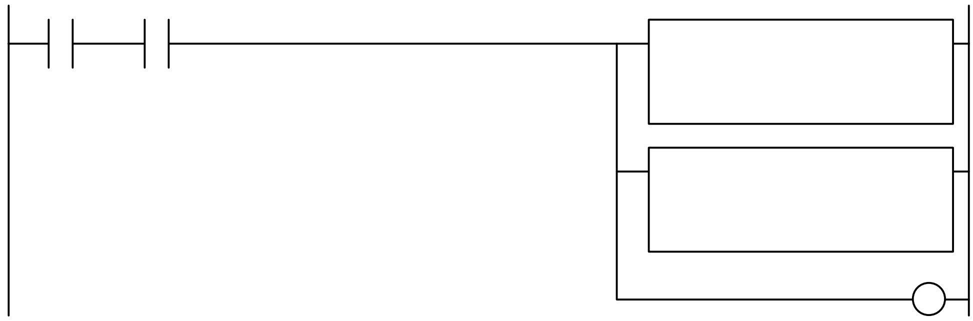
Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
—	No argument																	

Function

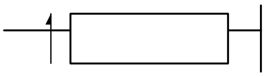
A start and an end of the processing box are represented.

参　考

- It is possible to write the operation details up to 32 digits inside the processing box.
- A processing box and a coil can be connected parallel.



Name	Start and end of processing box with rising edge

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	—	2	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		Useable to 1,024 maximum.
Condition	Time	Condition	Time	
—	0.06	—	—	

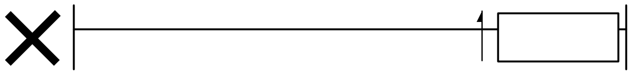
Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT　TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
—	No argument																

Function

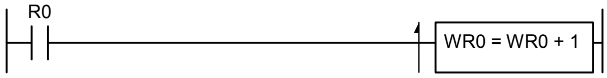
- A start and an end of the processing box with the rising edge are represented.
- When the operation result obtained by then changes ‘0’ to ‘1’, the operation in the processing box is performed.
- A processing box and a coil can be connected parallel.
- It is possible to write the operation result up to 32 digits in the processing box.

Cautionary notes

It is impossible to write the circuit without any condition (the circuit of only a processing box with the rising edge).

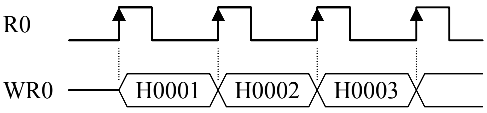


Program example

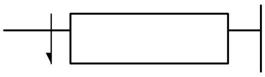


[Program description]

The operation in the processing box is performed only once at the rising of R0.



Name	Start and end of processing box with falling edge

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
	—	2	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		Useable to 1,024 maximum.
Condition	Time	Condition	Time	
—	0.06	—	—	

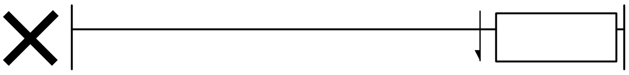
Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
—	No argument																

Function

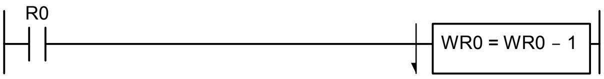
- A start and an end of the processing with the falling edge are represented.
- When the operation result obtained by then changes ‘1’ to ’0’, the operation in the processing box is performed.
- A processing box and a coil can be connected parallel.
- It is possible to write the operation details up to 32 digits in the processing box.

Cautionary notes

It is impossible to write the circuit without any condition (the circuit of only the processing box with the falling edge).

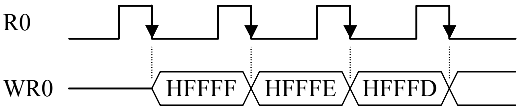


Program example



[Program description]

The operation in the processing box is performed only once at the falling of R0.



Name	Start and end of relational box

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
	—	0	●	●	●	●	●

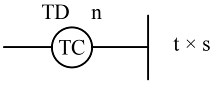
Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	—	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
—	No argument																

Function

A start and an end of the relational box are represented.

Name	On delay timer

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	—	6	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		• Timer points are 2,560 points. (0 – 2,559 / decimal) • Time base is selectable from 1, 10, 100, and 1000 [ms]. • Set value is 0 through 65,535.
Condition	Time	Condition	Time	
—	2.38	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	Time number																✓
t	Time base																
s	Set value							✓	✓	✓	✓						✓

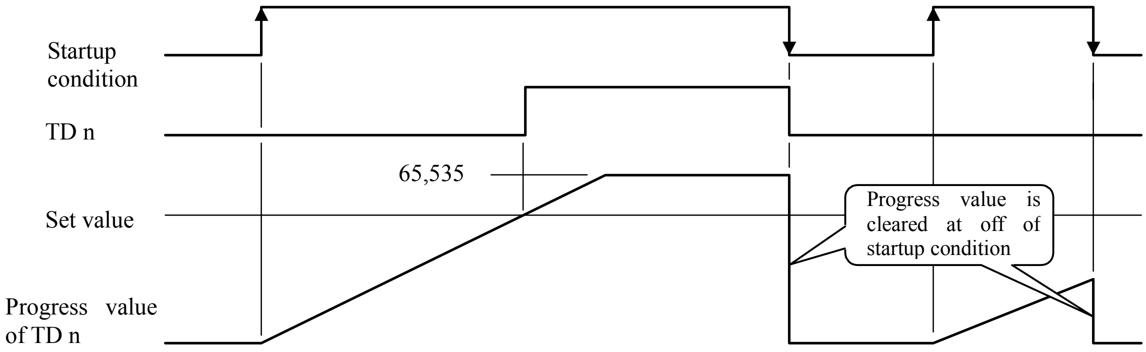
Function

- If Progress value ≥ Set value as a result of updating the progress value while a startup condition is on, a coil is turned on.
- If a startup condition is turned off, a coil is turned off because a progress value is cleared.
- A progress value gets into TC n. The progress value does not exceed 65,535 (decimal).
- If a progress value is updated while a system is running, the system operates in accordance with a new progress value at that time.
- If I/O is specified to a set value, the set value can be changed during operation by changing I/O value because of taking in the set value at each scan.

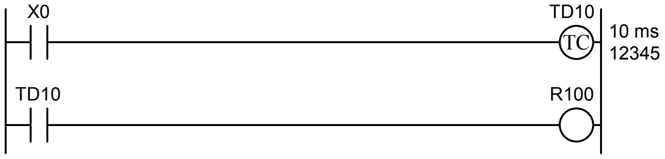
Cautionary notes

- A timer can be used to 2560 points maximum including TD, TDN, SS, MS, TMR, and WDT.
  However, the same area as a counter is used. A timer number and a counter number cannot be used repeatedly.
- A timer updates a progress value at the time the command is executed. Therefore, if a program that a portion of the timer command execution is not scanned after the timer has started up is created using JMP command and the master control (MCS), there are cases where the timer is not turned on correctly. (If time when a portion of the timer command execution is not scanned exceeds hours (='timer base' x 65,535), the timer is not turned on.)
  Also the progress value remains unchanged until the timer command is executed.

Time chart



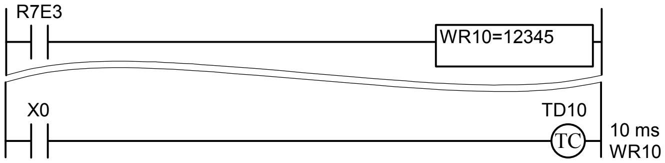
Program example



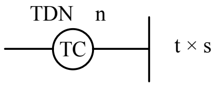
[Program description]

- If X0 is turned on, the progress value of TD10 is updated.
- If X0 is turned off, the progress value of TD10 is cleared.
- If the progress value ≥ the set value, TD10 is turned on.
- The progress value is increase while X0 is on, but it does not exceed 65,535.

  If X0 is turned off when TD10 is on, TD10 is turned off.
- The set value of the timer can be specified by Word I/O.



Name	Off delay timer

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	–	6	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		• Timer points are 2,560 points. (0 - 2559 / decimal) • Time base is selectable from 1, 10, 100, and 1000 [ms]. • Set value is 0 through 65,535.
Condition	Time	Condition	Time	
–	2.38	–	–	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)		WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	Timer number.																	✓
t	Time base																	
s	Set value								✓	✓	✓	✓						✓

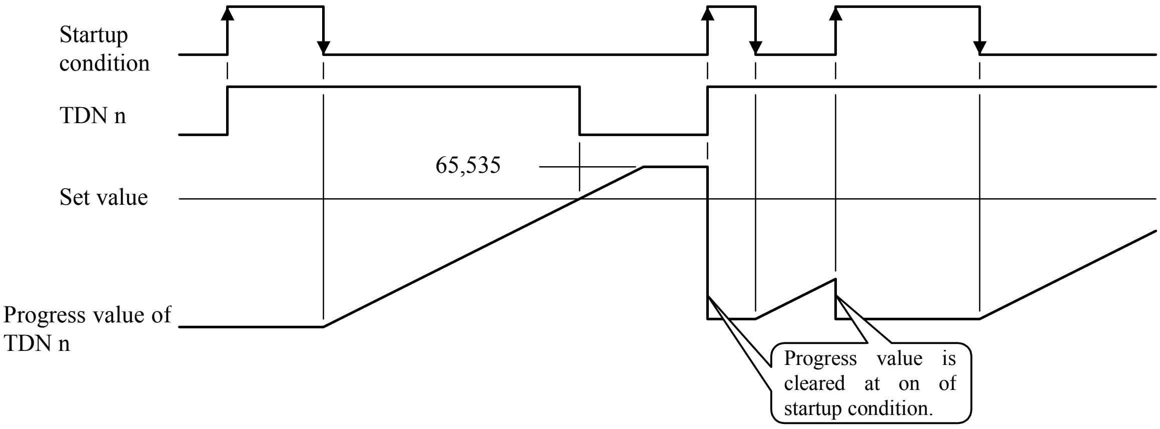
Function

- A rising edge of a startup condition is detected and a coil is switched on.
- If a startup condition is switched off, a progress value is updated. As a result, if the progress value ≥ the set value, a coil is turned off.
- If a startup condition is turned on, a progress value is cleared.
- A progress value gets into TC n. The progress value does not exceed 65,535 (decimal).
- If a progress value is updated while a system is running, the system operates in accordance with a new progress value at that time.
- If I/O is specified to a set value, the set value can be changed during operation by changing I/O value because of taking in the set value at each scan.

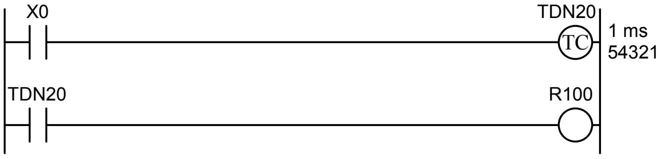
Cautionary notes

- A timer can be used to 2560 points maximum including TD, TDN, SS, MS, TMR, and WDT.
  However, the same area as a counter is used. A timer number and a counter number cannot be used repeatedly.
- A timer updates a progress value at the time the command is executed. Therefore, if a program that a portion of the timer command execution is not scanned after the timer has started up is created using JMP command and the master control (MCS), there are cases where the timer is not turned on correctly. (If time when a portion of the timer command execution is not scanned exceeds hours (= ‘time base’ x 65,535), the timer is not turned on correctly.)
  Also the progress value remains unchanged until the timer command is executed.

Time chart

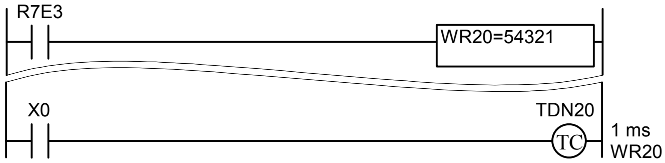


Program example

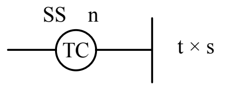


[Program description]

- If X0 is turned on, TDN is turned on. After that, if X0 is turned off, TDN20 starts updating of the progress value with on.
- If progress value ≥ set value, TDN20 is turned off.
- When X0 changed from on to off, a progress value of TDN20 does not exceed 65,535 although it increases while X0 is off.
- If Xo is switched on while a progress value of TDN20 is updated (X0 remains off), the progress value is cleared. (TDN20 remains on even if the progress value is cleared.)
- A set value is specified by Word I/O like TD.



Name	Single shot

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
	—	6	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		• Timer points are 2,560 points. (0 - 2559 /decimal) • Time base is selectable from 1, 10, 100, and 1000 [ms]. • Set value is 0 through 65,535.
Condition	Time	Condition	Time	
—	1.58	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	Timer number																✓
t	Time base																
s	Set value							✓	✓	✓	✓						✓

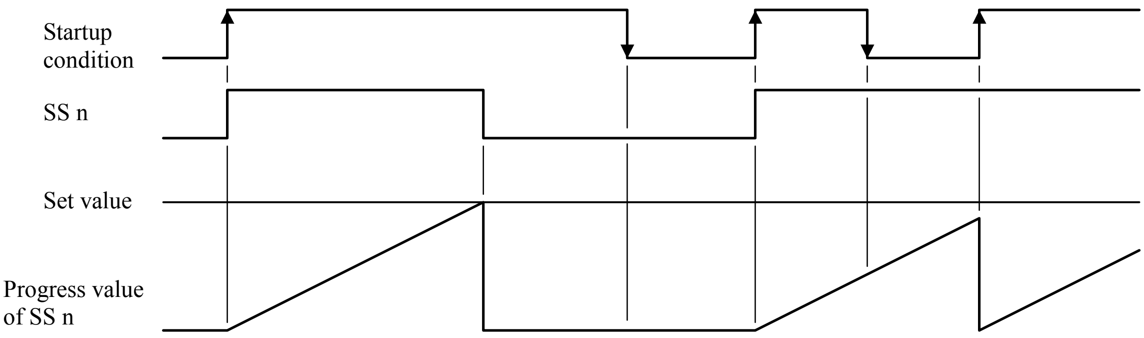
Function

- A rising edge of a startup condition is detected and a updating of a progress value is started. And a coil is switched on.
- If progress value ≥ set value, a coil is turned off. If a rising edge of a startup condition is detected furthermore while progress value < set value, it is counted from the beginning again with considering the progress value ‘0’
- A progress value gets into TC n. The progress value does not exceed a set value.
- If a progress value is updated while a system is running, the system operates in accordance with a new progress value at that time.
- If I/O is specified to a set value, the set value can be changed during operation by changing I/O value because of taking in the set value at each scan.

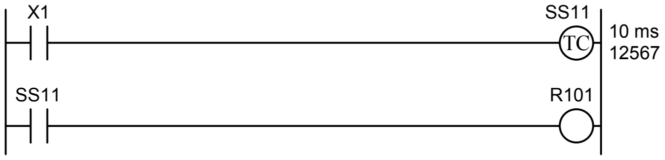
Cautionary notes

- Since a startup condition of Single shot is the edge detection, it is impossible to detect under the condition of one scan after the system started to run.
- A timer can be used to 2560 points maximum including TD, TDN, SS, MS, TMR, and WDT.
  However, the same area as a counter is used. A timer number and a counter number cannot be used repeatedly.
- A timer updates a progress value at the time the command is executed. Therefore, if a program that a portion of the timer command execution is not scanned after the timer has started up is created using JMP command and the master control (MCS), there are cases where the timer is not turned on correctly. (If time when a portion of the timer command execution is not scanned exceeds hours (= ‘time base (s)’ x 65,536), the timer is not turned on correctly.)
  Also the progress value remains unchanged until the timer command is executed.

Time chart



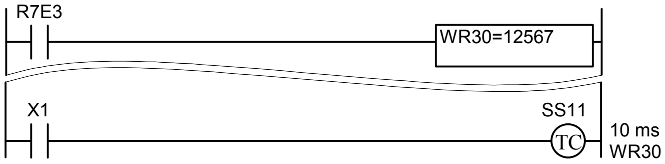
Program example



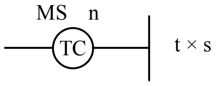
[Program description]

- SS11 is turned on as a result of updating a progress value at the rising edge of X1.
- If set value ≥ progress value, SS11 is turned off.

  A startup condition of Single shot is ignored because of an edge trigger although X1 is being turned on in this case.
- If a rising edge of X1 is detected before a progress value reaches a set value, the progress value starts to increase after returning to ‘0’ because a single shot timer is triggered again.
- A set value can be specified by Word I/O like TD.



Name	Mono stable timer

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
	—	6	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	1.68	—	—	• Timer points are 2,560 points. (0 - 2559 / decimal) • Time base is selectable from 1, 10, 100, and 1000 [ms]. • Set value is 0 through 65,535.

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	Timer number																✓
t	Time base																
s	Set value							✓	✓	✓	✓						✓

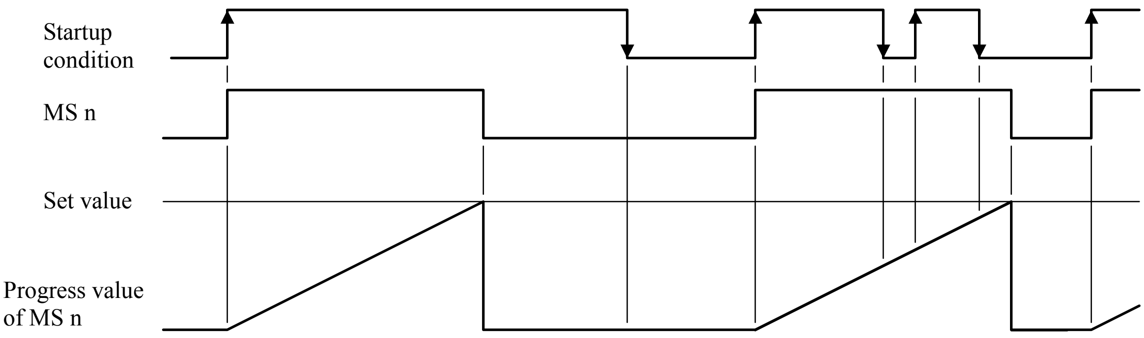
Function

- A rising edge of a startup condition is detected and a updating of a progress value is started. And a coil is switched on.
- If progress value ≥ set value, a coil is turned off. A rising of a startup condition while MS is on is ignored.
- A progress value gets into TC n. The progress value does not exceed a set value.
- If a progress value is updated while a system is running, the system operates in accordance with a new progress value at that time.
- If I/O is specified to a set value, the set value can be changed during operation by changing I/O value because of taking in the set value at each scan.

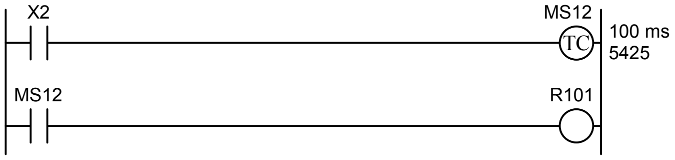
Cautionary notes

- Since a startup condition of Mono stable timer is an edge detection, it is impossible to detect under the condition of one scan after the system started to run.
- A timer can be used to 2560 points maximum including TD, TDN, SS, MS, TMR, and WDT.
  However, the same area as a counter is used. A timer number and a counter number cannot be used repeatedly.
- A timer updates a progress value at the time the command is executed. Therefore, if a program that a portion of the timer command execution is not scanned after the timer has started up is created using JMP command and the master control (MCS), there are cases where the timer is not turned on correctly. (If time when the timer command execution is not scanned exceeds hours (= 'time base(s)' x 65,536), the timer is not turned on correctly.)
  Also the progress value remains unchanged until the timer command is executed.

Time chart

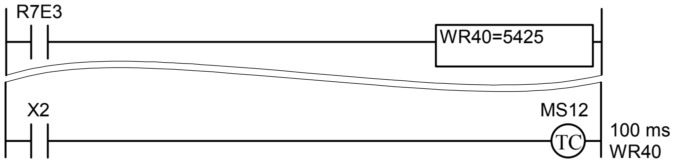


Program example

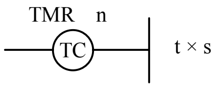


[Program description]

- MS12 is turned on as a result of updating a progress value at a rising of X2.
- If set value ≥ progress value, MS12 is turned off.
  A startup condition of Mono stable timer is ignored because of an edge trigger although X2 is being turned on in this case.
- Even if a rising edge of X2 is detected before a progress value reaches a set value, Mono stable timer ignores this rising.
- A set value can be specified by Word I/O like TD.



Name	Integral timer

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	—	6	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		• Timer points are 2,560 points. (0 - 2559 / decimal) • Time base is selectable from 1, 10, 100, and 1000 [ms]. • Set value is 0 through 65,535.
Condition	Time	Condition	Time	
—	2.52	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	Timer number																✓
t	Time base																
s	Set value							✓	✓	✓	✓						✓

Function

- A progress value is updated while a startup condition is on. The progress value restarts updating after the startup condition is turned on again without being cleared even if the condition is turned off.
- If progress value ≥ set value, a coil is turned on, and the coil is not turned off until a clear input CLn is turned on.
- A progress value gets into TC n. The progress value does not exceed 65,535 (decimal).
- If a progress value is updated while a system is running, the system operates in accordance with a new progress value at that time.
- If I/O is specified to a set value, the set value can be changed during operation by changing I/O value because of taking in the set value at each scan.

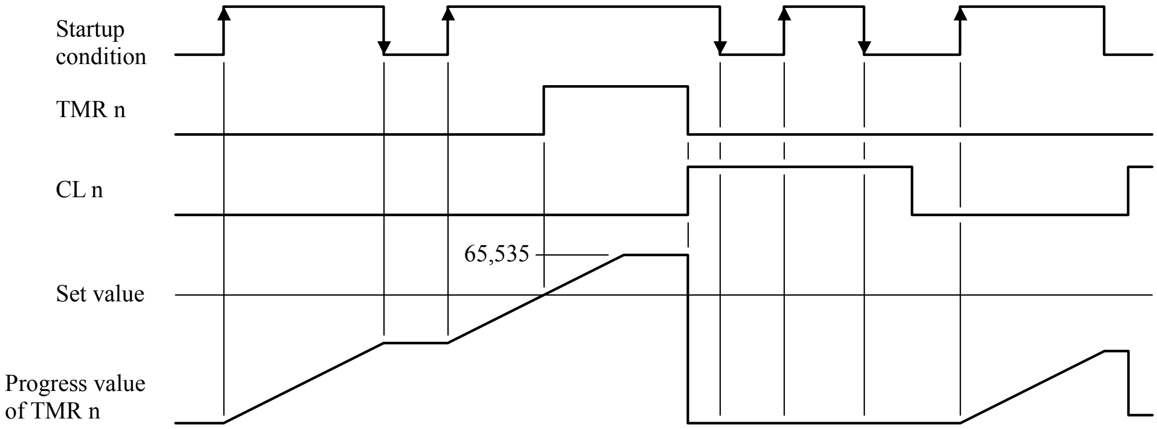
Cautionary notes

- 'on' of a startup condition is ignored while a clear input CLn is on.
- A timer can be used to 2560 points maximum including TD, TDN, SS, MS, TMR, and WDT.

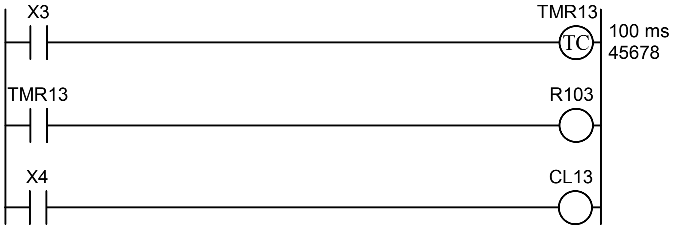
  However, the same area as a counter is used. A timer number and a counter number cannot be used repeatedly.
- A timer updates a progress value at the time the command is executed. Therefore, if a program that a portion of the timer command execution is not scanned after the timer has started up is created using JMP command and the master control (MCS), there are cases where the timer is not turned on correctly. (If time when the portion of the timer command execution is not scanned exceeds hours (= 'time base(s)' x 65,536), the timer is not turned on correctly.)

  Also the progress value remains unchanged until the timer command is executed.

Time chart

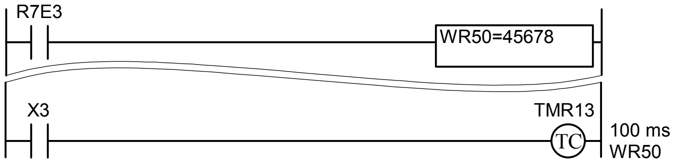


Program example

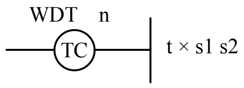


[Program description]

- The progress value is updated while X3 is on.
- If X3 is turned off, the updating of the progress value is retained after stopping.
- If X3 is turned on again, the updating of the progress value is restarted.
- If progress value ≥ set value, TMR13 is turned on. TMR13 is retained until a timer clear is turned on.
- If a timer clear (Cl13) is turned on, both a timer coil and the progress value are cleared.
- The startup condition is ignored while a timer clear (CL13) is on.
- The set value can be specified by Word I/O like TD.



Name	Watchdog timer

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	—	8	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		• Timer points are 2,560 points. (0 - 2559 / decimal) • Time base is selectable from 1, 10, 100, and 1000 [ms]. • 1st / 2nd set value is 0 - 65,535.
Condition	Time	Condition	Time	
—	2.26	—	—	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	Timer number																	✓
t	Time base																	
s1	The 1st set value								✓	✓	✓	✓						✓
s2	The 2nd set value								✓	✓	✓	✓						✓

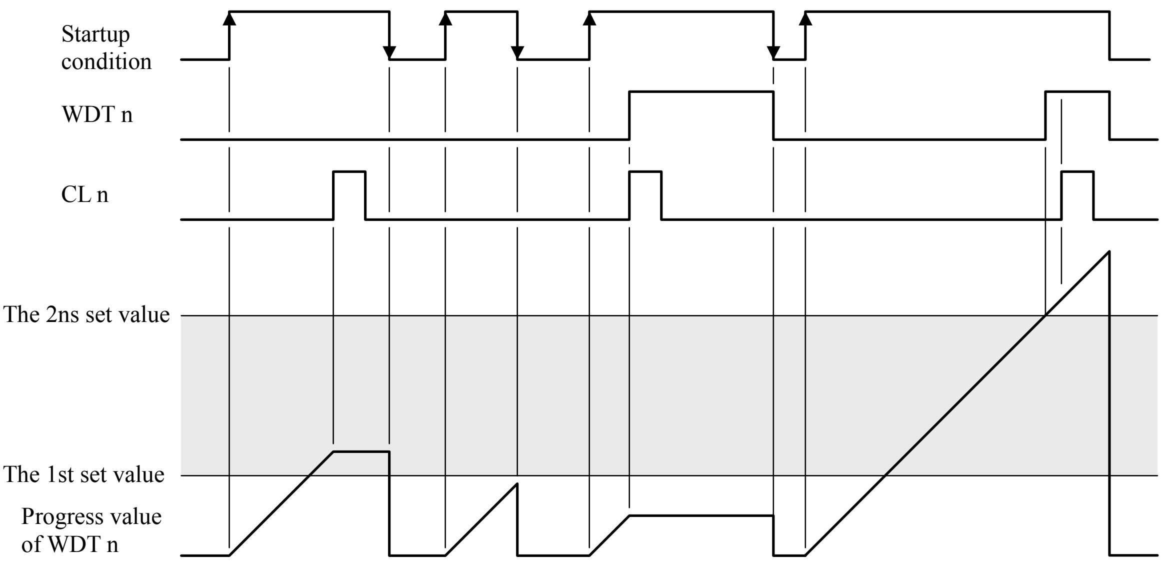
Function

- A progress value is updated while a startup condition is on.
  While the 1st set value ≤ progress value < the 2nd set value, if a clear input CLn is accessed, a coil is turned on. If the clear input CLn is accessed while progress value < the 1st set value and if the 2nd set value ≤ progress value, the coil is turned on. If the startup condition is turned off, all is cleared.
- A progress value gets into TC n. The progress value does not exceed 65,535(decimal).
- If a progress value is updated while a system is running, the system operates in accordance with a new progress value at that time
- If I/O is specified to a set value, the set value can be changed during operation by changing I/O value because of taking in the set value at each scan.

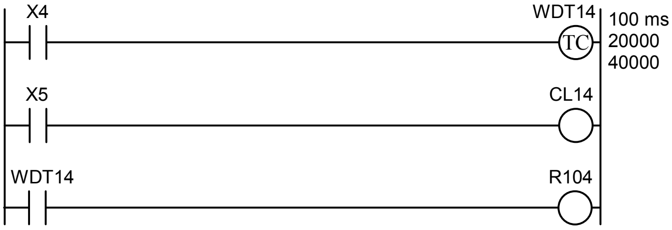
Cautionary notes

- A set value should surely fulfill the following condition, s1 < s2. Otherwise, just when a progress value reached s2, a coil is turned on.
- A timer can be used to 2560 points maximum including TD, TDN, SS, MS, TMR, and WDT.
  However, the same area as counter is used. A timer number and a counter number cannot be used repeatedly.
- A timer updates a progress value at the time the command is executed. Therefore, if a program that a portion of the timer command execution is not scanned after the timer has started up is created using JMP command and the master control (MCS), there are cases where the timer is not turned on correctly. (If time when the portion of the timer command execution is not scanned exceeds hours (= 'time base(s)' x 65,536), the timer is not turned on correctly.)
  Also the progress value remains unchanged until the timer command is executed.

Time chart



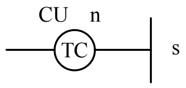
Program example



[Program description]

- A clear operates on condition just before WDT coil command is executed.
- A progress value is updated while X4 is on.
- If a watchdog clear (CL14) is switched on before a progress value exceeds the 2nd set value after exceeding the 1st set value, WDT14 (R104) is not turned on.
- If X4 is turned off, a progress value and an output of WDT coil are cleared.
- If a startup condition is turned off before a progress value exceeds the 1st set value, the progress value returns ‘0’ (zero clear) without turning WDT coil on.
- If a watchdog clear (CL14) is switched on before a progress value exceeds the 1st set value, WDT14 (R104) is turned on. The then progress value is retained.
- If a watchdog clear (SL14) is not switched on even if a progress value exceeds the 2nd set value, WDT14 is turned on. The progress value is updating without changing.
- Even if a watchdog clear (CL14) is switched on after a progress value exceeds the 2nd set value and then WDT coil has been turned on, it is ignored.

Name	Counter

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	—	6	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		•Timer points are 512 points. (0 - 511 / decimal) • Set value is 0 - 65,535.
Condition	Time	Condition	Time	
—	0.70	—	—	

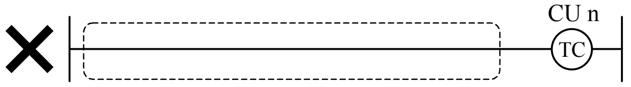
Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	Timer number																✓
s	Set value							✓	✓	✓	✓						✓

Function

- ‘1’ is added to a progress value at each rising edge detection of a startup condition, and a coil is turned on at progress value ≥ set value.
- If a counter clear CL n is turned on, the coil turned on is turned off and the progress value returns to ‘0’ (zero clear).
- A progress value gets into TC n. The progress value does not exceed 65,535 (decimal).
- If a progress value is updated while a system is running, the system operates in accordance with a new progress value at that time.
- If I/O is specified to a set value, the set value can be changed during operation by changing I/O value because of taking in the set value at each scan.

Cautionary notes

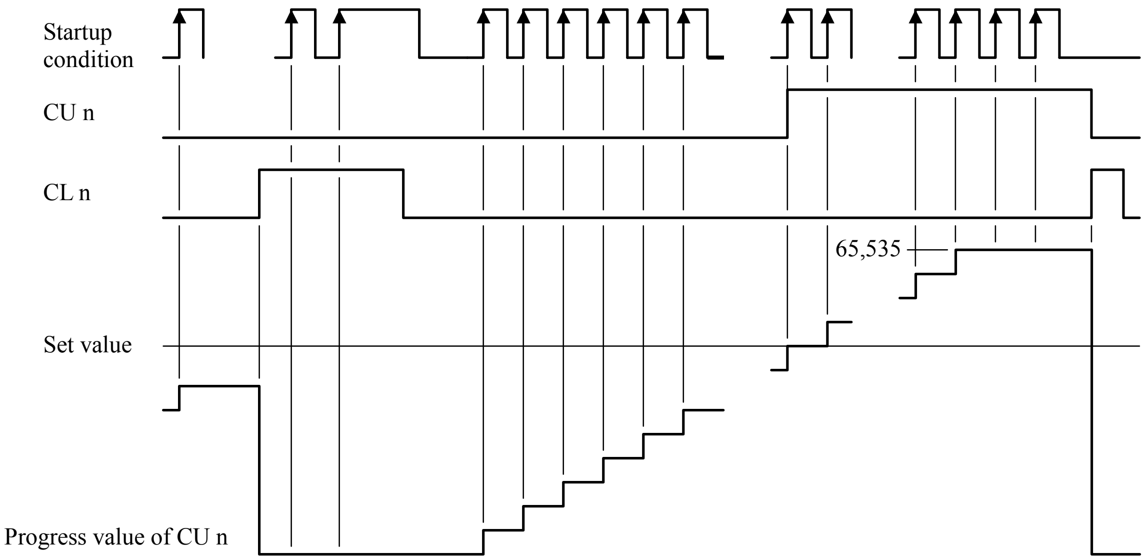
- A counter can be used 512 points maximum (No.0 to No.511), but the same area as a timer is used.
- A timer number and a counter number cannot be used repeatedly.
- A counter cannot be used singly. (The condition is necessary in front of a coil.)



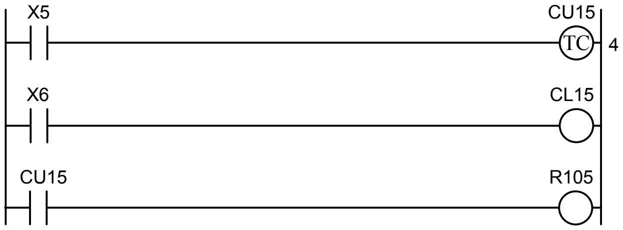
- A rising of a startup condition is ignored while a counter clear CLn is on.
- Since a startup condition of a counter is the edge detection, it is impossible to detect the condition of one scan (R7E3) after the system started to run.

  If a set value is set to ‘0’, a coil is controlled by CLn as a result of being always on.
- Counter clear is performed in a counter coil. (A state of the counter clear is monitored in the counter coil, and is cleared.) If the counter coil is cleared, it is necessary to switch on the counter clear before the counter coil is executed.

Time chart



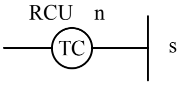
Program example



[Program description]

- The progress value is updated at the rising edge of X5.
- If set value ≤ progress value, the counter coil (CU15) is turned on.
- The counter value does not exceed 65,535.
- If the counter coil (CL15) is switched on, the progress value and the counter coil are cleared.

Name	Ring counter

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	—	6	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		• Timer points are 512 points. (0 - 511 / decimal) • Set value is 0 through 65,535.
Condition	Time	Condition	Time	
—	0.82	—	—	

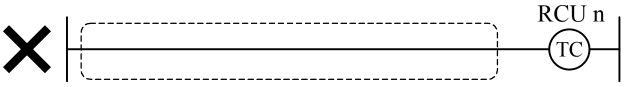
Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	Timer number																	✓
s	Set value								✓	✓	✓	✓						✓

Function

- '1' is added to a progress value at each rising edge detection of a startup condition. The progress value returns to '0' (zero clear) at progress value ≥ set value, and a coil for one scan is switched on. If a counter clear CLn is turned on, the progress value is set to '0' and the coil is also turned off.
- A progress value gets into TC n. The progress value does not exceed a set value.
- If a progress value is updated while a system is running, the system operates in accordance with a new progress value at that time. If I/O is specified to a set value, the set value can be changed during operation by changing I/O value because of taking in the set value at each scan.

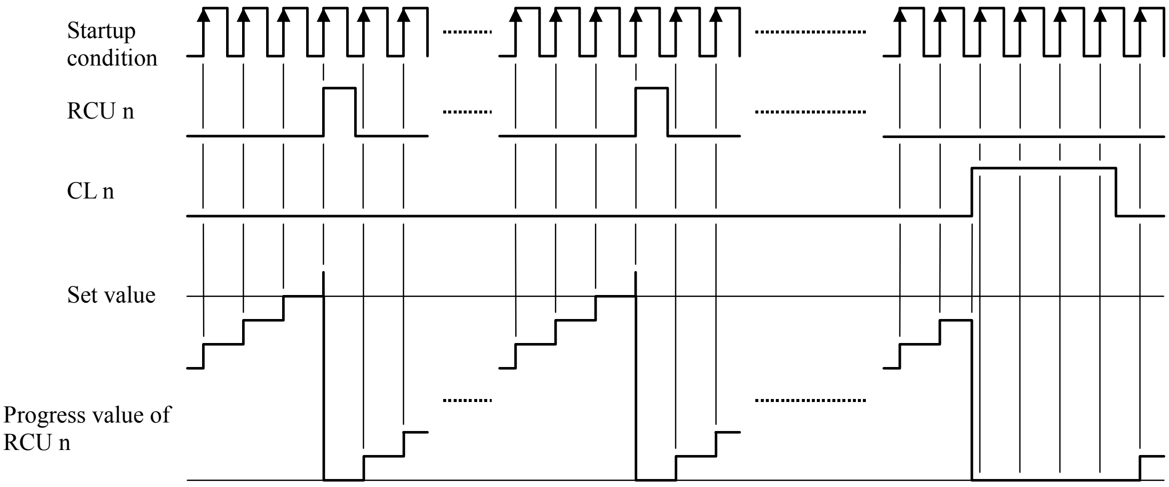
Cautionary notes

- A counter can be used to 512 points maximum (No.0 to No.511), but the same area as a timer is used.
- A timer number and a counter number cannot be used repeatedly.
- A counter cannot be used singly. (A condition is necessary in front of the coil.)

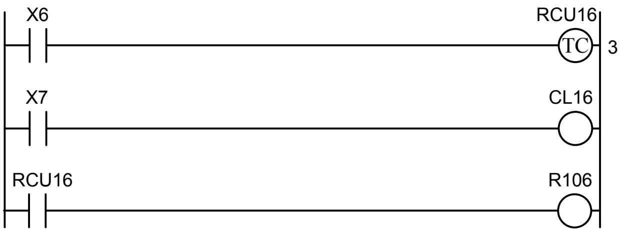


- A rising of a startup condition is ignored while a counter clear CLn is on.
- Since a startup condition of a counter is the edge detection, it is impossible to detect the condition of one scan (R7E3) after the system started to run.
  If a set value is set to '0', a coil is controlled by CLn as a result of being always on.
- A counter clear is performed in a counter coil. (A state of the counter clear is monitored in the counter coil and cleared.) If the counter coil is cleared, it is necessary to switch on the counter clear before the counter coil is executed.

Time chart



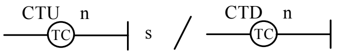
Program example



[Program description]

- The progress value (count value) is updated at the rising edge of X6.
- If set value = progress value, the count coil (RCU16) is turned on for one scanning time and the progress value returns to '0' (zero clear).
- If the counter clear (CL16) is turned on, the progress value returns to '0' (zero clear). The progress value is not updated while the counter clear is on.

Name	Up counter, Down counter

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	CTU n s	6	●	●	●	●	●
	CTD n	4					

Command processing time (μs)				Remarks
Average		Maximum		• Timer points are 512 points. (0 - 511 / decimal) • Set value is 0 through 65,535.
Condition	Time	Condition	Time	
CTU n s	0.70	—	—	
CTD n	0.54			

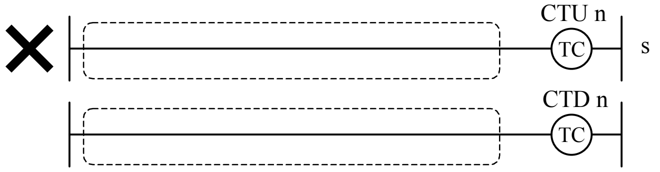
Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	Timer number																✓
s	Set value							✓	✓	✓	✓						✓

Function

- Up counter adds ‘1’ to a progress value and Down counter subtracts ‘1’ out of a progress value at each rising edge detection of a startup condition. A coil is turned on at progress value ≥ set value and the coil is turned off at progress value < set value. If a counter clear CLn is turned on, the progress value returns to ‘0’ (zero clear), and the coil is also turned off.
- A progress value gets into TC n. The progress value is 0 through 65,535 (decimal).
- If a progress value is updated while a system is running, the system operates in accordance with a new progress value at that time. If I/O is specified to a set value, the set value can be changed during operation by changing I/O value because of taking in the set value at each scan.

Cautionary notes

- The number of an up coil and a down coil should be same.
- A counter can be used to 512 points maximum (No.0 to No.511), but the same area as a timer is used.
- A timer number and a counter number cannot be used repeatedly.
- A counter cannot be used singly. (The condition is necessary in front of a coil.)

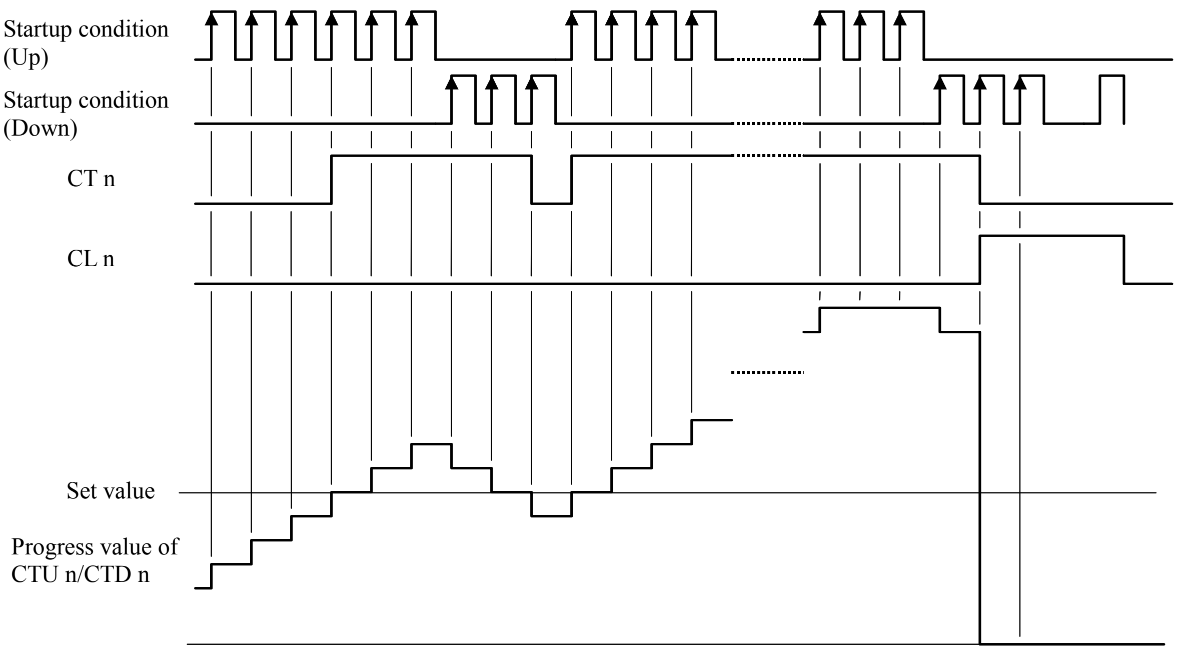


- A rising of a startup condition is ignored while a counter clear CLn is on.
- Since a startup condition of a counter is the edge detection, it is impossible to detect the condition of one scan (R7E3) after a system started to run.

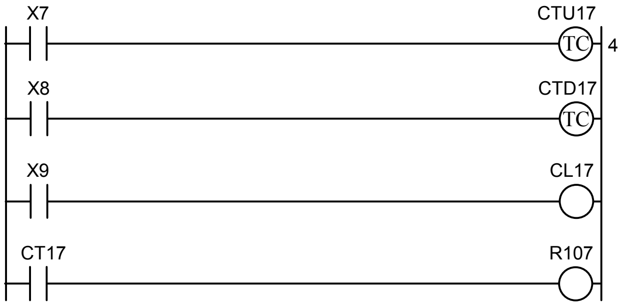
If a set value is set to ‘0’, a coil is controlled by CLn as a result of being always on.

- A counter clear is performed in a counter coil. (A state of a counter clear is monitored in the counter coil, and cleared.) If the counter coil is cleared, it is necessary to switch on the counter clear before the counter coil is executed.

Time chart



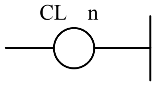
Program example



[Program description]

- The progress value (count value) is up-counted at the rising edge of X7.
- A counter coil is turned on at set value ≤ progress value.
- If the startup conditions the up coil and the down coil are turned on simultaneously, the progress value does not change.
- The progress value is down-counted at the rising edge of X8.
- A counter coil is turned off at set value > progress value.
- The progress value does not exceed 65,535 and fall below 0 either.
- If a counter clear (CL17) is turned on, the progress value and the counter coil are cleared. The progress value is not updated while the counter clear is on.

Name	Counter clear

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	–	1	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		Timer points are 512 points. (0 - 511 / decial)
Condition	Time	Condition	Time	
–	0.04	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
n	Timer number																✓

Function

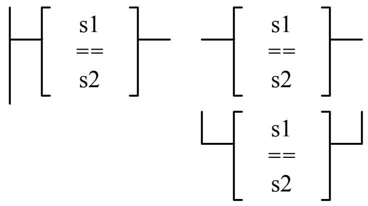
- A progress value of in integral timer returns to ‘0’ (zero clear) and then a timer coil is switched off.
- As for WDT, time monitor is checked. (See WDT for details.)
- As for a counter, a progress value is cleared and a counter coil is turned off.

  Clear operation is performed in the coil of a counter and a timer which are adapting to a clear coil. (A state of the clear coil is monitored in the coil of the counter and the timer, and cleared.)

Cautionary notes

- If a timer is switched off and a progress value is cleared, CLn of the same number as the timer should be switched on. This is the same also when clearing a counter.

Name	= Comparative box

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)				
Average				
Condition	Processing time			
	Word		Double word	
LD (s1 == s2) s1:I/O, s2: I/O	0.10	[5]	0.96	[6]
LD (s1 == s2) s1:I/O, s2:C	0.10	[5]	0.92	[7]
LD (s1 == s2) s1:C, s2:I/O	0.10	[5]	0.92	[7]
AND (s1 == s2) s1:I/O, s2: I/O	0.10	[5]	0.96	[6]
AND (s1 == s2) s1:I/O, s2:C	0.10	[5]	0.92	[7]
AND (s1 == s2) s1:C, s2: I/O	0.10	[5]	0.92	[7]
OR (s1 == s2) s1:I/O, s2: I/O	0.12	[6]	0.98	[7]
OR (s1 == s2) s1:I/O, s2:C	0.12	[6]	0.94	[8]
OR (s1 == s2) s1:C, s2: I/O	0.12	[6]	0.94	[8]

Remarks

- C means a constant
- [ ]; number of steps (in Processing time column)

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1	Comparative number 1							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Comparative number 2							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- s1 and s2 are compared as unsigned integers.

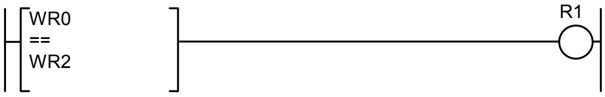
  When s1 = s2, it is a continuity state (ON)

  When s1 ≠ s2, it is a non-continuity state (OFF)

- When s1 and s2 are Word, 0 - 65,535 (decimal), H0000 - HFFFF (hexadecimal)

  When s1 and s2 are Double word, 0 - 4,294,967,295 (decimal), H00000000 - HFFFFFFFF (hexadecimal)

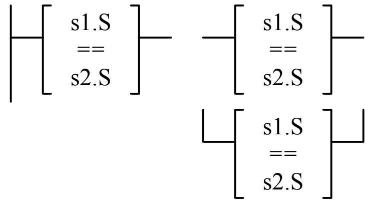
Program example



[Program description]

When WR0 = WR2, R1 is turned on, and when WR0 ≠ WR2, R1 is turned off.

Name	= Comparative box (Signed integer)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)					Remarks
Average					
Condition	Processing time				
	Word		Double word		
LD (s1.S == s2.S) s1:I/O, s2: I/O	0.10	[5]	0.96	[6]	• C means a constant. • [ ]; number of steps (in Processing time column)
LD (s1.S == s2.S) s1:I/O, s2:C	0.10	[5]	0.92	[7]	
LD (s1.S == s2.S) s1:C, s2:I/O	0.10	[5]	0.92	[7]	
AND (s1.S == s2.S) s1:I/O, s2: I/O	0.10	[5]	0.96	[6]	
AND (s1.S == s2.S) s1:I/O, s2:C	0.10	[5]	0.92	[7]	
AND (s1.S == s2.S) s1:C, s2: I/O	0.10	[5]	0.92	[7]	
OR (s1.S == s2.S) s1:I/O, s2: I/O	0.12	[6]	0.98	[7]	
OR (s1.S == s2.S) s1:I/O, s2:C	0.12	[6]	0.94	[8]	
OR (s1.S == s2.S) s1:C, s2: I/O	0.12	[6]	0.94	[8]	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1.S	Comparative number 1								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Comparative number 2								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- s1.S and s2.S are compared as signed integers.

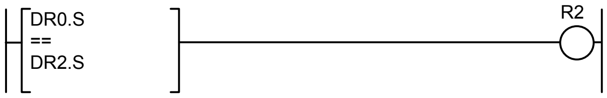
  When s1.S = s2.S, it is a continuity state (ON).

  When s1.S ≠ s2.S, it is a non-continuity state (OFF).

- When s1.S and s2.S are Word, −32,768 - 32,767 (decimal), H8000 - H7FFF (hexadecimal)

  When s1.S and s2.S are Double word, −2,147,483,648 - 2,147,483,647 (decimal), H80000000 - H7FFFFFFF (hexadecimal)

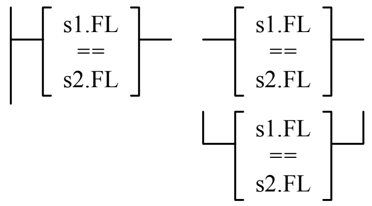
Program example



[Program description]

When DR0.S = DR2.S, R2 is turned on, when DR0.S ≠ DR2.S, R2 is turned off.

Name	= Comparative box (Floating decimal point)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)			Remarks
Average			• Floating decimal point is specified by Double word. • C means a constant. A constant is 20 digits maximum. • [ ]; number of steps (in Processing time column)
Condition	Processing time		
LD (s1.FL == s2.FL) s1:I/O, s2: I/O	0.94	[6]	
LD (s1.FL == s2.FL) s1:I/O, s2:C	0.90	[7]	
LD (s1.FL == s2.FL) s1:C, s2:I/O	0.90	[7]	
AND (s1.FL == s2.FL) s1:I/O, s2: I/O	0.94	[6]	
AND (s1.FL == s2.FL) s1:I/O, s2:C	0.90	[7]	
AND (s1.FL == s2.FL) s1:C, s2: I/O	0.90	[7]	
OR (s1.FL == s2.FL) s1:I/O, s2: I/O	0.96	[7]	
OR (s1.FL == s2.FL) s1:I/O, s2:C	0.92	[8]	
OR (s1.FL == s2.FL) s1:C, s2: I/O	0.92	[8]	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1.FL	Comparative number 1													✓	✓	✓	✓	✓
s2.FL	Comparative number 2													✓	✓	✓	✓	✓

Function

- s1.FL and s2.FL are compared as floating decimal points.

  When s1.FL = s2.FL, it is a continuity state (ON).

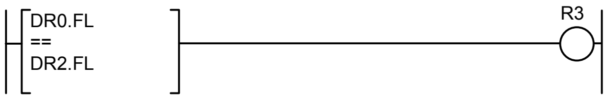
  When s1.FL ≠ s2.FL, it is a non-continuity state (OFF).
- s1.FL, s2.FL: -3.40282×10^{38} - 3.40282×10^{38} (decimal),

  HFF7FFFFF - H80800000, H00800000 - H7F7FFFFF (hexadecimal)

Cautionary notes

Since there is an error in a floating decimal point, there are cases where the value disagrees by the error even if the value is in agreement on calculation. We recommend deciding after comparing floating decimal points by not agreement or disagreement, by “range”.

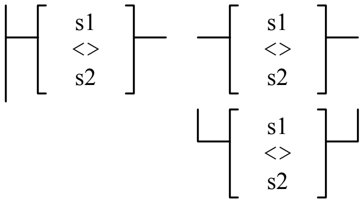
Program example



[Program description]

When DR0.FL = DR2.FL, R3 is turned on and when DR0.FL ≠ DR2.FL, R3 is turned off.

Name	≠ Comparative box

<table>
<tr><th rowspan="2">Ladder format</th><th colspan="2">Number of steps</th><th colspan="5">Condition code</th></tr>
<tr><th>Condition</th><th>Steps</th><th>R7F4
DER</th><th>R7F3
ERR</th><th>R7F2
SD</th><th>R7F1
V</th><th>R7F0
C</th></tr>
<tr><td> </td><td colspan="2">(See Command processgin time column)</td><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
</table>

<table>
<tr><th colspan="5">Command processing time (μs)</th><th>Remarks</th></tr>
<tr><th colspan="5">Average</th><td rowspan="12">• C means a constant.
• [ ]; number of steps
(in Processing time column)</td></tr>
<tr><th rowspan="2">Condition</th><th colspan="4">Processing time</th></tr>
<tr><th colspan="2">Word</th><th colspan="2">Double word</th></tr>
<tr><td>LD (s1 <> s2) s1:I/O, s2: I/O</td><td>0.10</td><td>[5]</td><td>0.96</td><td>[6]</td></tr>
<tr><td>LD (s1 <> s2) s1:I/O, s2:C</td><td>0.10</td><td>[5]</td><td>0.92</td><td>[7]</td></tr>
<tr><td>LD (s1 <> s2) s1:C, s2:I/O</td><td>0.10</td><td>[5]</td><td>0.92</td><td>[7]</td></tr>
<tr><td>AND (s1 <> s2) s1:I/O, s2: I/O</td><td>0.10</td><td>[5]</td><td>0.96</td><td>[6]</td></tr>
<tr><td>AND (s1 <> s2) s1:I/O, s2:C</td><td>0.10</td><td>[5]</td><td>0.92</td><td>[7]</td></tr>
<tr><td>AND (s1 <> s2) s1:C, s2: I/O</td><td>0.10</td><td>[5]</td><td>0.92</td><td>[7]</td></tr>
<tr><td>OR (s1 <> s2) s1:I/O, s2: I/O</td><td>0.12</td><td>[6]</td><td>0.98</td><td>[7]</td></tr>
<tr><td>OR (s1 <> s2) s1:I/O, s2:C</td><td>0.12</td><td>[6]</td><td>0.94</td><td>[8]</td></tr>
<tr><td>OR (s1 <> s2) s1:C, s2: I/O</td><td>0.12</td><td>[6]</td><td>0.94</td><td>[8]</td><td></td></tr>
</table>

<table>
<tr><th colspan="2" rowspan="2">Usable I/O</th><th colspan="6">Bit</th><th colspan="5">Word</th><th colspan="4">Double word</th><th rowspan="2">Constant</th></tr>
<tr><th>X</th><th>Y</th><th>EX
EY</th><th>R,
L,
M</th><th>TD, TDN,
SS, WDT,
MS, TMR,
CU, RCU,
CT</th><th>WR,
WN
(.m)</th><th>WX</th><th>WY</th><th>WEX,
WEY</th><th>WR,
WL,
WM,
WN</th><th>TC</th><th>DX</th><th>DY</th><th>DEX,
DEY</th><th>DR,
DL,
DM</th></tr>
<tr><td>s1</td><td>Comparative number 1</td><td></td><td></td><td></td><td></td><td></td><td></td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td></tr>
<tr><td>s2</td><td>Comparative number 2</td><td></td><td></td><td></td><td></td><td></td><td></td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td><td>✓</td></tr>
</table>

Function

- s1and s2 are compared as unsigned integers.

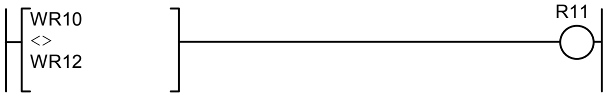
  When s1 ≠ s2, it is a continuity state (ON).

  When s1 = s2, it is a non-continuity state (OFF).

- When s1 and s2 are Word, 0 - 65,535 (decimal), H0000 - HFFFF (hexadecimal)

  When s1 and s2 are Double word, 0 - 4,294,967,295 (decimal), H00000000 - HFFFFFFFF (hexadecimal)

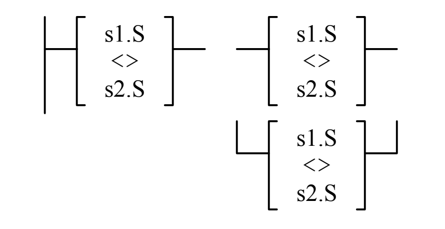
Program example



[Program description]

When WR10 ≠ WR12, R11 is turned on and when WR10 = WR12, R11 is turned off.

Name	≠ Comparative box (Singed integer)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)					Remarks
Average					• C means a constant. • [ ]; number of steps (in Processing time column)
Condition	Processing time				
	Word		Double word		
LD (s1.S <> s2.S) s1:I/O, s2: I/O	0.10	[5]	0.96	[6]	
LD (s1.S <> s2.S) s1:I/O, s2:C	0.10	[5]	0.92	[7]	
LD (s1.S <> s2.S) s1:C, s2:I/O	0.10	[5]	0.92	[7]	
AND (s1.S <> s2.S) s1:I/O, s2: I/O	0.10	[5]	0.96	[6]	
AND (s1.S <> s2.S) s1:I/O, s2:C	0.10	[5]	0.92	[7]	
AND (s1.S <> s2.S) s1:C, s2: I/O	0.10	[5]	0.92	[7]	
OR (s1.S <> s2.S) s1:I/O, s2: I/O	0.12	[6]	0.98	[7]	
OR (s1.S <> s2.S) s1:I/O, s2:C	0.12	[6]	0.94	[8]	
OR (s1.S <> s2.S) s1:C, s2: I/O	0.12	[6]	0.94	[8]	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1.S	Comparative number 1								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Comparative number 2								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- s1.S and s2.S are compared as signed integers.

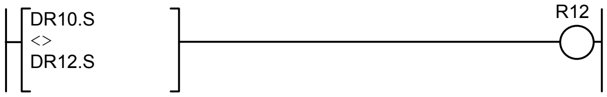
  When s1.S ≠ s2.S, it is a continuity state (ON).

  When s1.S = s2.S, it is a non-continuity state (OFF).

- When s1.S and s2.S are Word, −32,768 -32,767 (decimal), H8000 - H7FFF (hexadecimal)

  When s1.S and s2.S are Double word, −2,147,483,648 - 2,147,483,647 (decimal), H80000000 - H7FFFFFFF (hexadecimal)

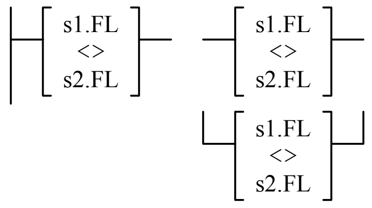
Program example



[Program description]

When DR10.S ≠ DR12.S, R12 is turned on and when DR10.S = DR12.S, R12 is turned off.

Name	≠ Comparative box (Floating decimal points)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
 s1.FL <> s2.FL s1.FL <> s2.FL s1.FL <> s2.FL	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)			Remarks
Average			• Floating decimal point is specified by Double word. • C means a constant. Constant is 20 digit maximum. • [ ]; number of steps (in Processing time column)
Condition	Processing time		
LD (s1.FL <> s2.FL) s1:I/O, s2: I/O	0.94	[6]	
LD (s1.FL <> s2.FL) s1:I/O, s2:C	0.90	[7]	
LD (s1.FL <> s2.FL) s1:C, s2:I/O	0.90	[7]	
AND (s1.FL <> s2.FL) s1:I/O, s2: I/O	0.94	[6]	
AND (s1.FL <> s2.FL) s1:I/O, s2:C	0.90	[7]	
AND (s1.FL <> s2.FL) s1:C, s2: I/O	0.90	[7]	
OR (s1.FL <> s2.FL) s1:I/O, s2: I/O	0.96	[7]	
OR (s1.FL <> s2.FL) s1:I/O, s2:C	0.92	[8]	
OR (s1.FL <> s2.FL) s1:C, s2: I/O	0.92	[8]	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1.FL	Comparative number 1													✓	✓	✓	✓	✓
s2.FL	Comparative number 2													✓	✓	✓	✓	✓

Function

- s1.FL and s2.FL are compared as floating decimal points.

  When s1.FL ≠ s2.FL, it is a continuity (ON).

  When s1.FL = s2.FL, it is a non-continuity (OFF).

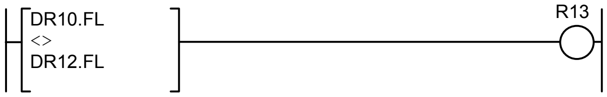
- s1.FL, s2.FL: -3.40282×10^{38} - 3.40282×10^{38} (decimal),

  HFF7FFFFF - H80800000, H00800000 - H7F7FFFFF (hexadecimal)

Cautionary notes

Since there is an error in a floating decimal point, there are cases where the value disagrees by the error even if the value is in agreement on calculation. We recommend deciding after comparing floating decimal points by not agreement or disagreement, but “range”.

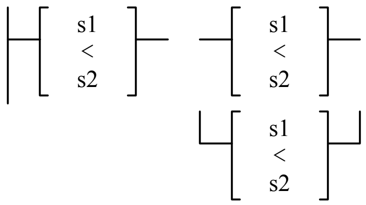
Program example



[Program description]

When DR10.FL ≠ DR12.FL, R13 is turned on and when DR10.FL = DR12.FL, R13 is turned off.

Name	< Comparative box

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)					Remarks
Average					• C means a constant. • [ ]; number of steps (in Processing time column)
Condition	Processing time				
	Word		Double word		
LD (s1 < s2) s1:I/O, s2: I/O	0.10	[5]	0.94	[6]	
LD (s1 < s2) s1:I/O, s2:C	0.10	[5]	0.90	[7]	
LD (s1 < s2) s1:C, s2:I/O	0.10	[5]	0.90	[7]	
AND (s1 < s2) s1:I/O, s2: I/O	0.10	[5]	0.94	[6]	
AND (s1 < s2) s1:I/O, s2:C	0.10	[5]	0.90	[7]	
AND (s1 < s2) s1:C, s2: I/O	0.10	[5]	0.90	[7]	
OR (s1 < s2) s1:I/O, s2: I/O	0.12	[6]	0.96	[7]	
OR (s1 < s2) s1:I/O, s2:C	0.12	[6]	0.92	[8]	
OR (s1 < s2) s1:C, s2: I/O	0.12	[6]	0.92	[8]	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1	Comparative number 1							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Comparative number 2							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- s1 and s2 are compared as unsigned integers.

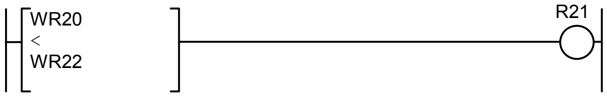
  When s1 < s2, it is a continuity state (ON).

  When s1 ≥ s2, it is a non-continuity state (OFF).

- When s1 and s2 are Word, 0 - 65,535 (decimal), H0000 - HFFFF (hexadecimal)

  When s1 and s2 are Double word, 0 - 4,294,967,295 (decimal), H00000000 - HFFFFFFFF (hexadecimal)

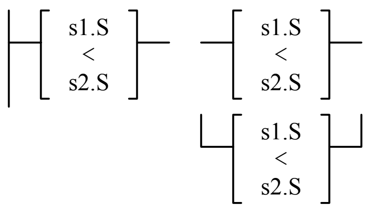
Program example



[Program description]

When WR20 < WR22, R21 is turned on and when WR20 ≥ WR22, R21 is turned off.

Name	< Comparative box (Signed integer)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)					Remarks
Average					• C means a constant. • [ ]; number of steps (in Processing time column)
Condition	Processing time				
	Word		Double word		
LD (s1.S < s2.S) s1:I/O, s2: I/O	1.00	[5]	1.16	[6]	
LD (s1.S < s2.S) s1:I/O, s2:C	0.96	[5]	1.12	[7]	
LD (s1.S < s2.S) s1:C, s2:I/O	0.96	[5]	1.12	[7]	
AND (s1.S < s2.S) s1:I/O, s2: I/O	1.00	[5]	1.16	[6]	
AND (s1.S < s2.S) s1:I/O, s2:C	0.96	[5]	1.12	[7]	
AND (s1.S < s2.S) s1:C, s2: I/O	0.96	[5]	1.12	[7]	
OR (s1.S < s2.S) s1:I/O, s2: I/O	1.02	[6]	1.18	[7]	
OR (s1.S < s2.S) s1:I/O, s2:C	0.98	[6]	1.14	[8]	
OR (s1.S < s2.S) s1:C, s2: I/O	0.98	[6]	1.14	[8]	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1.S	Comparative number 1								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Comparative number 2								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- s1.S and s2.S are compared as signed integers.

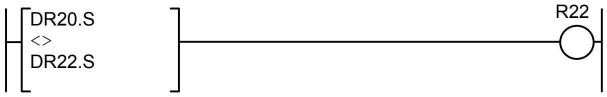
  When s1.S < s2.S, it is a continuity state (ON).

  When s1.S ≥ s2.S, it is a non-continuity state (OFF).

- When s1.S and s2.S are Word, −32,768 - 32,767 (decimal), H8000 - H7FFF (hexadecimal)

  When s1.S and s2.S are Double word, −2,147,483,648 - 2,147,483,647 (decimal), H80000000 - H7FFFFFFF (hexadecimal)

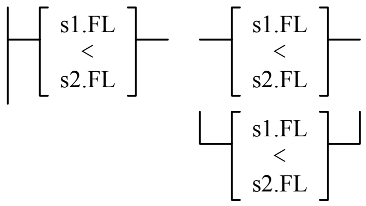
Program example



[Program description]

When DR20.S < DR22.S, R22 is turned on and DR20.S ≥ DR22.S, R22 is turned off.

Name	< Comparative box (Floating decimal point)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)			Remarks
Average			• Floating decimal point is specified by Double word. • C means a constant. Constant is 20 digit maximum. • [ ]; number of steps (in Processing time column)
Condition	Processing time		
LD (s1.FL < s2.FL) s1:I/O, s2: I/O	2.22	[6]	
LD (s1.FL < s2.FL) s1:I/O, s2:C	2.28	[7]	
LD (s1.FL < s2.FL) s1:C, s2:I/O	2.16	[7]	
AND (s1.FL < s2.FL) s1:I/O, s2: I/O	2.22	[6]	
AND (s1.FL < s2.FL) s1:I/O, s2:C	2.28	[7]	
AND (s1.FL < s2.FL) s1:C, s2: I/O	2.16	[7]	
OR (s1.FL < s2.FL) s1:I/O, s2: I/O	2.24	[7]	
OR (s1.FL < s2.FL) s1:I/O, s2:C	2.30	[8]	
OR (s1.FL < s2.FL) s1:C, s2: I/O	2.18	[8]	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1.FL	Comparative number 1													✓	✓	✓	✓	✓
s2.FL	Comparative number 2													✓	✓	✓	✓	✓

Function

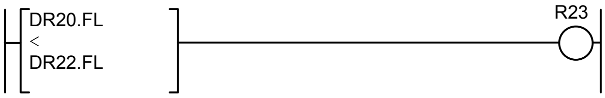
- s1.FL and s2.FL are compared as floating decimal points.

  When s1.FL < s2.FL, it is a continuity state (ON).

  When s1.FL ≥ s2.FL, it is a non-continuity state (OFF).
- s1.FL, s2.FL: -3.40282×10^{38} - 3.40282×10^{38} (decimal),

  HFF7FFFFF - H80800000, H00800000 - H7F7FFFFF (hexadecimal)

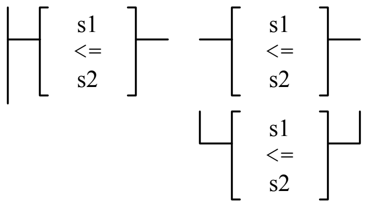
Program example



[Program description]

When DR20.FL < DR22.FL, R23 is turned on and when DR20.FL ≥ DR22.FL, R23 is turned off.

Name	≤ Comparative box

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)					Remarks
Average					• C means a constant. • [ ]; number of steps (in Processing time column)
Condition	Processing time				
	Word		Double word		
LD (s1 <= s2) s1:I/O, s2: I/O	0.10	[5]	1.06	[6]	
LD (s1 <= s2) s1:I/O, s2:C	0.10	[5]	1.02	[7]	
LD (s1 <= s2) s1:C, s2:I/O	0.10	[5]	1.02	[7]	
AND (s1 <= s2) s1:I/O, s2: I/O	0.10	[5]	1.06	[6]	
AND (s1 <= s2) s1:I/O, s2:C	0.10	[5]	1.02	[7]	
AND (s1 <= s2) s1:C, s2: I/O	0.10	[5]	1.02	[7]	
OR (s1 <= s2) s1:I/O, s2: I/O	0.12	[6]	1.08	[7]	
OR (s1 <= s2) s1:I/O, s2:C	0.12	[6]	1.04	[8]	
OR (s1 <= s2) s1:C, s2: I/O	0.12	[6]	1.04	[8]	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1	Comparative number 1							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Comparative number 2							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- s1 and s2 are compared as unsigned integers.

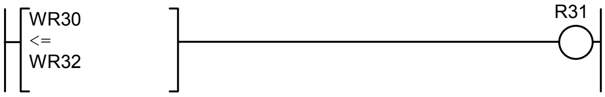
  When s1 ≤ s2, it is a continuity state (ON).

  When s1 > s2, it is a non-continuity state (OFF).

- When s1 and s2 are Word, 0 - 65,535 (decimal), H0000 - HFFFF (hexadecimal)

  When s1 and s2 are Double word, 0 - 4,294,967,295 (decimal), H00000000 - HFFFFFFFF (hexadecimal)

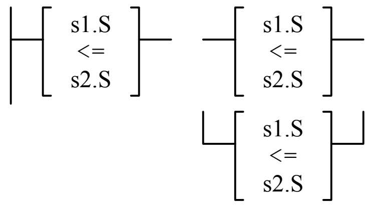
Program example



[Program description]

When WR30 ≤ WR32, R31 is turned on and when WR30 > WR32, R31 is turned off.

Name	≤ Comparative box (Signed integer)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)					Remarks
Average					• C means a constant. • [ ]; number of steps (in Processing time column)
Condition	Processing time				
	Word		Double word		
LD (s1.S <= s2.S) s1:I/O, s2: I/O	0.98	[5]	1.22	[6]	
LD (s1.S <= s2.S) s1:I/O, s2:C	0.96	[5]	1.18	[7]	
LD (s1.S <= s2.S) s1:C, s2:I/O	0.96	[5]	1.14	[7]	
AND (s1.S <= s2.S) s1:I/O, s2: I/O	0.98	[5]	1.22	[6]	
AND (s1.S <= s2.S) s1:I/O, s2:C	0.96	[5]	1.18	[7]	
AND (s1.S <= s2.S) s1:C, s2: I/O	0.96	[5]	1.14	[7]	
OR (s1.S <= s2.S) s1:I/O, s2: I/O	1.00	[6]	1.24	[7]	
OR (s1.S <= s2.S) s1:I/O, s2:C	0.98	[6]	1.20	[8]	
OR (s1.S <= s2.S) s1:C, s2: I/O	0.98	[6]	1.16	[8]	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1.S	Comparative number 1							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Comparative number 2							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- s1.S and s2.S are compared as signed integers.

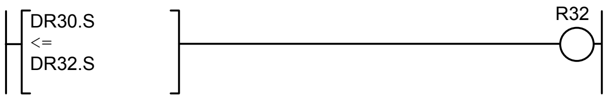
  When s1.S ≤ s2.S, it is a continuity state (ON).

  When s1.S > s2.S, it is a non-continuity state (OFF).

- When s1.S and s2.S are Word, −32,768 - 32,767 (decimal), H8000 - H7FFF (hexadecimal)

  When s1.S and s2.S are Double word, −2,147,483,648 - 2,147,483,647 (decimal), H80000000 - H7FFFFFFF (hexadecimal)

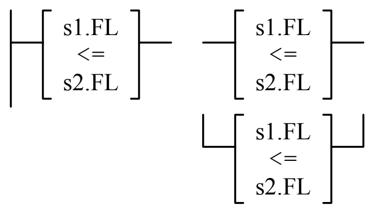
Program example



[Program description]

When DR30.S ≤ DR32.S, R32 is turned on and when DR30.S > DR32.S, R32 is turned off.

Name	≤ Comparative box (Floating decimal point)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
 s1.FL <= s2.FL s1.FL <= s2.FL s1.FL <= s2.FL	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)			Remarks
Average			• Floating decimal point is specified by Double word. • C means a constant. Constant is 20 digit maximum. • [ ]; number of steps (in Processing time column)
Condition	Processing time		
LD (s1.FL <= s2.FL) s1:I/O, s2: I/O	2.24	[6]	
LD (s1.FL <= s2.FL) s1:I/O, s2:C	2.28	[7]	
LD (s1.FL <= s2.FL) s1:C, s2:I/O	2.16	[7]	
AND (s1.FL <= s2.FL) s1:I/O, s2: I/O	2.24	[6]	
AND (s1.FL <= s2.FL) s1:I/O, s2:C	2.28	[7]	
AND (s1.FL <= s2.FL) s1:C, s2: I/O	2.16	[7]	
OR (s1.FL <= s2.FL) s1:I/O, s2: I/O	2.26	[7]	
OR (s1.FL <= s2.FL) s1:I/O, s2:C	2.30	[8]	
OR (s1.FL <= s2.FL) s1:C, s2: I/O	2.18	[8]	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1.FL	Comparative number 1													✓	✓	✓	✓	✓
s2.FL	Comparative number 2													✓	✓	✓	✓	✓

Function

- s1.FL and s2.FL are compared as floating decimal points.

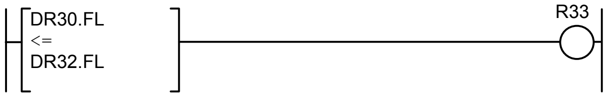
  When s1.FL ≤ s2.FL, it is a continuity state (ON).

  When s1.FL > s2.FL, it is a non-continuity state (OFF).

- s1.FL, s2.FL: -3.40282×10^{38} - 3.40282×10^{38} (decimal),

  HFF7FFFFF - H80800000, H00800000 - H7F7FFFFF (hexadecimal)

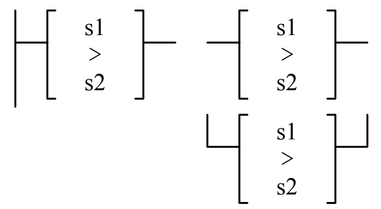
Program example



[Program description]

When DR30.FL ≤ DR32.FL, R33 is turned on and when DR30.FL > DR32.FL, R33 is turned off.

Name	> Comparative box

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)					Remarks
Average					• C means a constant. • [ ]; number of steps (in Processing time column)
Condition	Processing time				
	Word		Double word		
LD (s1 > s2) s1:I/O, s2: I/O	0.10	[5]	1.00	[6]	
LD (s1 > s2) s1:I/O, s2:C	0.10	[5]	0.96	[7]	
LD (s1 > s2) s1:C, s2:I/O	0.10	[5]	0.96	[7]	
AND (s1 > s2) s1:I/O, s2: I/O	0.10	[5]	1.00	[6]	
AND (s1 > s2) s1:I/O, s2:C	0.10	[5]	0.96	[7]	
AND (s1 > s2) s1:C, s2: I/O	0.10	[5]	0.96	[7]	
OR (s1 > s2) s1:I/O, s2: I/O	0.12	[6]	1.02	[7]	
OR (s1 > s2) s1:I/O, s2:C	0.12	[6]	0.98	[8]	
OR (s1 > s2) s1:C, s2: I/O	0.12	[6]	0.98	[8]	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1	Comparative number 1							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Comparative number 2							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- s1 and s2 are compared as unsigned integers.

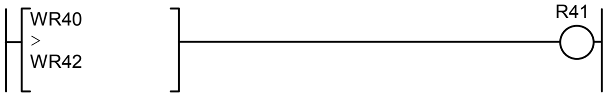
  When s1 > s2, it is a continuity state (ON).

  When s1 ≤ s2, it is a non-continuity state (OFF).

- When s1 and s2 are Word, 0 - 65,535 (decimal), H0000 - HFFFF (hexadecimal)

  When s1 and s2 are Double word, 0 - 4,294,967,295 (decimal), H00000000 - HFFFFFFFF (hexadecimal)

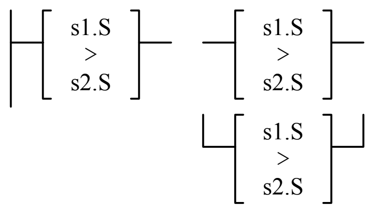
Program example



[Program description]

When WR40 > WR42, R41 is turned on and when WR40 ≤ WR42, R41 is turned off.

Name	> Comparative box (Signed integer)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)					Remarks
Average					• C means a constant. • [ ]; number of steps (in Processing time column)
Condition	Processing time				
	Word		Double word		
LD (s1.S > s2.S) s1:I/O, s2: I/O	1.00	[5]	1.16	[6]	
LD (s1.S > s2.S) s1:I/O, s2:C	0.98	[5]	1.12	[7]	
LD (s1.S > s2.S) s1:C, s2:I/O	0.98	[5]	1.12	[7]	
AND (s1.S > s2.S) s1:I/O, s2: I/O	1.00	[5]	1.16	[6]	
AND (s1.S > s2.S) s1:I/O, s2:C	0.98	[5]	1.12	[7]	
AND (s1.S > s2.S) s1:C, s2: I/O	0.98	[5]	1.12	[7]	
OR (s1.S > s2.S) s1:I/O, s2: I/O	1.02	[6]	1.18	[7]	
OR (s1.S > s2.S) s1:I/O, s2:C	1.00	[6]	1.14	[8]	
OR (s1.S > s2.S) s1:C, s2: I/O	1.00	[6]	1.14	[8]	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1.S	Comparative number 1								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Comparative number 2								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

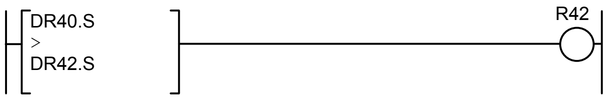
- s1.Sand s2.S are compared as signed integers.

  When s1.S > s2.S, it is a continuity state (ON).

  When s1.S ≤ s2.S, it is a non-continuity state (OFF).
- When s1.S and s2.S are Word, −32,768 - 32,767 (decimal), H8000 - H7FFF (hexadecimal)

  When s1.S and s2.S are Double word, −2,147,483,648 - 2,147,483,647 (decimal), H80000000 - H7FFFFFFF (hexadecimal)

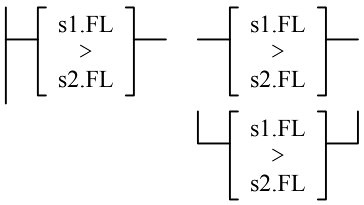
Program example



[Program description]

When DR40.S > DR42.S, R42 is turned on and DR40.S ≤ DR42.S, R42 is turned off.

Name	> Comparative box (Floating decimal point)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)		
Average		
Condition	Processing time	
LD (s1.FL > s2.FL) s1:I/O, s2: I/O	2.22	[6]
LD (s1.FL > s2.FL) s1:I/O, s2:C	2.28	[7]
LD (s1.FL > s2.FL) s1:C, s2:I/O	2.16	[7]
AND (s1.FL > s2.FL) s1:I/O, s2: I/O	2.22	[6]
AND (s1.FL > s2.FL) s1:I/O, s2:C	2.28	[7]
AND (s1.FL > s2.FL) s1:C, s2: I/O	2.16	[7]
OR (s1.FL > s2.FL) s1:I/O, s2: I/O	2.24	[7]
OR (s1.FL > s2.FL) s1:I/O, s2:C	2.30	[8]
OR (s1.FL > s2.FL) s1:C, s2: I/O	2.18	[8]

Remarks
- Floating decimal point is specified by Double word.
- C means a constant. Constant is 20 digit maximum.
- [ ]; number of steps (in Processing time column)

Usable I/O		Bit							Word					Double word				
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	Constant
s1.FL	Comparative number 1													✓	✓	✓	✓	✓
s2.FL	Comparative number 2													✓	✓	✓	✓	✓

Function

- s1.FL and s2.FL are compared as floating decimal points.

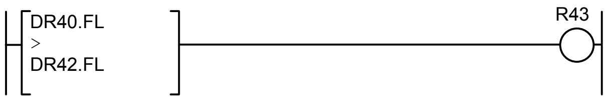
  When s1.FL > s2.FL, it is a continuity state (ON).

  When s1.FL ≤ s2.FL, it is a non-continuity state (OFF).

- s1.FL, s2.FL: -3.40282×10^{38} - 3.40282×10^{38} (decimal),

  HFF7FFFFF - H80800000, H00800000 - H7F7FFFFF (hexadecimal)

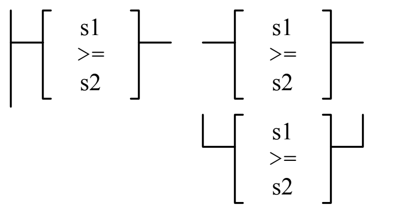
Program example



[Program description]

When DR40.FL > DR42.FL, R43 is turned on and when DR40.FL ≤ DR42.FL, R43 is turned off.

Name	≥ Comparative box

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)					Remarks
Average					• C means a constant. • [ ]; number of steps (in Processing time column)
Condition	Processing time				
	Word		Double word		
LD (s1 >= s2) s1:I/O, s2: I/O	0.10	[5]	1.06	[6]	
LD (s1 >= s2) s1:I/O, s2:C	0.10	[5]	1.02	[7]	
LD (s1 >= s2) s1:C, s2:I/O	0.10	[5]	1.02	[7]	
AND (s1 >= s2) s1:I/O, s2: I/O	0.10	[5]	1.06	[6]	
AND (s1 >= s2) s1:I/O, s2:C	0.10	[5]	1.02	[7]	
AND (s1 >= s2) s1:C, s2: I/O	0.10	[5]	1.02	[7]	
OR (s1 >= s2) s1:I/O, s2: I/O	0.12	[6]	1.08	[7]	
OR (s1 >= s2) s1:I/O, s2:C	0.12	[6]	1.04	[8]	
OR (s1 >= s2) s1:C, s2: I/O	0.12	[6]	1.04	[8]	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1	Comparative number 1							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Comparative number 2							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- s1 and s2 are compared as unsigned integers.

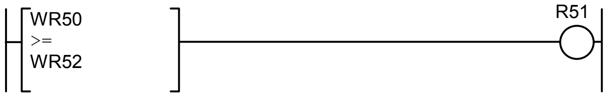
  When s1≥ s2, it is a continuity state (ON).

  When s1 < s2, it is a non-continuity state (OFF).

- When s1 and s2 are Word, 0 - 65,535 (decimal), H0000 - HFFFF (hexadecimal)

  When s1 and s2 are Double word, 0 - 4,294,967,295 (decimal), H00000000 - HFFFFFFFF (hexadecimal)

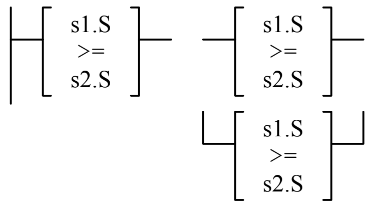
Program example



[Program description]

When WR50 ≥ WR52, R51 is turned on and WR50 < WR52, R51 is turned off.

Name	≥ Comparative box (Signed integer)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
 s1.S >= s2.S s1.S >= s2.S s1.S >= s2.S	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)					Remarks
Average					
Condition	Processing time				
	Word		Double word		
LD (s1.S >= s2.S) s1:I/O, s2: I/O	0.98	[5]	1.22	[6]	• C means a constant. • [ ]; number of steps (in Processing time column)
LD (s1.S >= s2.S) s1:I/O, s2:C	0.96	[5]	1.18	[7]	
LD (s1.S >= s2.S) s1:C, s2:I/O	0.96	[5]	1.14	[7]	
AND (s1.S >= s2.S) s1:I/O, s2: I/O	0.98	[5]	1.22	[6]	
AND (s1.S >= s2.S) s1:I/O, s2:C	0.96	[5]	1.18	[7]	
AND (s1.S >= s2.S) s1:C, s2: I/O	0.96	[5]	1.14	[7]	
OR (s1.S >= s2.S) s1:I/O, s2: I/O	1.00	[6]	1.24	[7]	
OR (s1.S >= s2.S) s1:I/O, s2:C	0.98	[6]	1.20	[8]	
OR (s1.S >= s2.S) s1:C, s2: I/O	0.98	[6]	1.16	[8]	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
s1.S	Comparative number 1							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Comparative number 2							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- s1.S and s2.S are compared as signed integers.

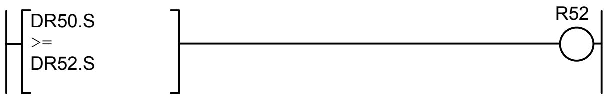
  When s1.S ≥ s2.S, it is a continuity state (ON).

  When s1.S < s2.S, it is a non-continuity state (OFF).

- When s1.S and s2.S are Word, −32,768 - 32,767 (decimal), H8000 - H7FFF (hexadecimal)

  When s1.S and s2.S are Double word, −2,147,483,648 - 2,147,483,647 (decimal), H80000000 - H7FFFFFFF (hexadecimal)

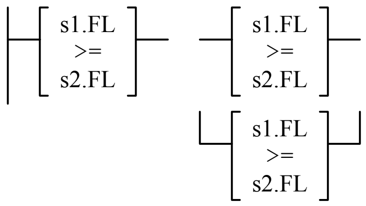
Program example



[Program description]

When DR50.S ≥ DR52.S, R52 is turned on and when DR50.S < DR52.S, R52 is turned off.

Name	≥ Comparative box (Floating decimal point)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	(See Command processgin time column)		●	●	●	●	●

Command processing time (μs)			Remarks
Average			
Condition	Processing time		
LD (s1.FL >= s2.FL) s1:I/O, s2: I/O	2.24	[6]	• Floating decimal point is specified by Double word. • C means a constant. Constant is 20 digit maximum. • [ ]; number of steps (in Processing time column)
LD (s1.FL >= s2.FL) s1:I/O, s2:C	2.28	[7]	
LD (s1.FL >= s2.FL) s1:C, s2:I/O	2.16	[7]	
AND (s1.FL >= s2.FL) s1:I/O, s2: I/O	2.24	[6]	
AND (s1.FL >= s2.FL) s1:I/O, s2:C	2.28	[7]	
AND (s1.FL >= s2.FL) s1:C, s2: I/O	2.16	[7]	
OR (s1.FL >= s2.FL) s1:I/O, s2: I/O	2.26	[7]	
OR (s1.FL >= s2.FL) s1:I/O, s2:C	2.30	[8]	
OR (s1.FL >= s2.FL) s1:C, s2: I/O	2.18	[8]	

Usable I/O		Bit							Word					Double word				
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	Constant
s1.FL	Comparative number 1													✓	✓	✓	✓	✓
s2.FL	Comparative number 2													✓	✓	✓	✓	✓

Function

- s1.FL and s2.FL are compared as floating decimal points.

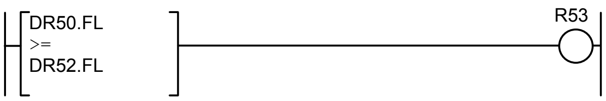
  When s1.FL ≥ s2.FL, it is a continuity state (ON).

  When s1.FL < s2.FL, it is a non-continuity state (OFF).

- s1.FL, s2.FL: -3.40282×10^{38} - 3.40282×10^{38} (decimal),

  HFF7FFFFF - H80800000, H00800000 - H7F7FFFFF (hexadecimal)

Program example



[Program description]

When DR50.FL ≥ DR52.FL, R53 is turned on and when DR50.FL < DR52.FL, R53 is turned off.

[1] Basic commands

[2] Arithmetic commands

[3] Application commands

[4] Control commands

[5] CPU serial communications commands

[6] High-function module transfer commands

Name	Substitution statement

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = s d.m_1 = s d.m_1 = s.m_2 d = s.m_2	(See Command processing time column)		↕	●	●	●	●

Command processing time (μs)								Remarks
Maximum								• m_1 and m_2 are 0 through F. • Bit in Word cannot be used in the array. • W means Word I/O. • DW means Double I/O. • C means a constant. • [ ]; number of steps (in Processing time column)
d	s	d, s: Bit		d, s: Word		d, s: Double word		
I/O	I/O / C	0.06	[3]	0.06	[3]	0.46	[5]	
I/O.m_1	I/O / C	0.52	[5]	—		—		
I/O	I/O.m_2	0.42	[5]	—		—		
I/O.m_1	I/O.m_2	0.58	[6]	—		—		
Array (W / C)	I/O / C	1.54	[5]	1.60	[5]	1.92	[6]	
Array (DW)	I/O / C	1.62	[6]	1.68	[6]	1.98	[7]	
I/O	Array (W / C)	2.06	[5]	1.98	[5]	2.60	[5]	
I/O	Array (DW)	2.12	[6]	2.04	[6]	2.66	[6]	
Array (W / C)	Array (W / C)	3.16	[6]	3.12	[6]	3.44	[6]	
Array (DW)	Array (W / C)	3.22	[7]	3.18	[7]	3.50	[7]	
Array (W / C)	Array (DW)	3.22	[7]	3.18	[7]	3.50	[7]	
Array (DW)	Array (DW)	3.22	[8]	3.18	[8]	3.56	[7]	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓				✓	✓	✓	✓		✓	✓	✓	
d. m_1	Substitution destination						✓										
s	Substitution source	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s.m_2	Substitution source						✓										
( )	Index value							✓	✓	✓	✓						

Function

- d = s A content of ‘s’ is substituted for ‘d’.
- d.m_1 = s A content of bit ‘s’ is substituted for the m_1-th bit of word data ‘d’.

  e.g. WR0.4=R0 A state of R0 is stored in the 4th bit of WR0.

  If WR0=0, R0=1, WR0=H0010.
- d = s.m_2 The m_2-th bit of word data ‘s’ is substituted for bit ‘d’.

  e.g. R10=WR0.F A state of MSB (Most Significant Bit) of WR0 is stored in R10.

  If WR0=HFFFF, R10=1.
- d.m_1 = s.m_2 The m_2-th bit of word data ‘s’ is substituted for them$_1$-th bit of word data ‘d’.

  e.g. WR100.8=WR10.0 A state of LSB (Less Significant Bit) of WR10 is stored in the 8th bit of WR100.

  If WR100=H0F00 and WR10=H0000, WR100=H0E00.
- An array variable can be used for ‘d’ and ‘s’.
- A constant can be used in the following range,

  at Word 0 - 65,535 (decimal), H0000 - HFFFF (hexadecimal)

  at Double word 0 - 4,294,967,295 (decimal), H00000000 - HFFFFFFFF (hexadecimal)

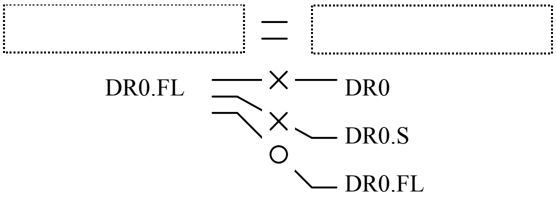
Cautionary notes

- The type is not converted in a substitution formula. Although description to substitute an internal output with an extension (.S, .FL) for an internal output with other extension by changing / deleting the extension temporarily is possible, 16 bits data (or 32 bits data) are substituted unchanged. 16 bits (or 32 bits) data is substituted without changing.

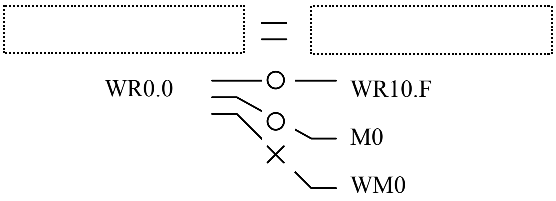
  e.g. When DR0.S = −2000(HFFFFF830),

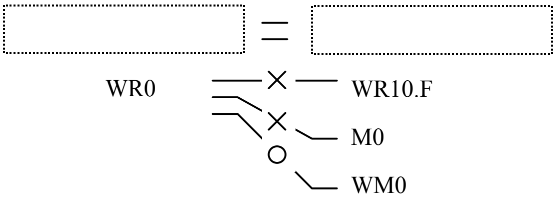
  DM0 = DR0

  (If DR0.S and DM0 are monitored by hexadecimal, ‘HFFFFF830’ are both displayed. But if it is done by decimal, ‘-2000’ is displayed in DR0.S and ‘4294965296’ is displayed in DM0.)



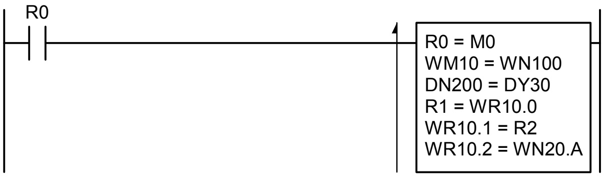
- Write that the type of the left-hand side and the right-hand side is in agreement.





- When an array variable is being used, DER is set to ‘1’ if it exceeds the maximum of usable I/O number, and DER is set to ‘0’ if it is normal.
- If a constant is used for an index of the array, the constant is valid from 0 through 65,535 (decimal) and from H0000 through HFFFF (hexadecimal).
- ‘d.m_1’ and ‘s.m_2’ can specified only the word internal output.
- Since ‘d.m_1’ and ‘s.m_2’ are the bit in the word internal output, if they are used for the substitution statement, a substitution destination or a substitution source should be set to the bit.

Program example



```
R0 = M0
WM10 = WN100
DN200 = DY30
R1 = WR10.0
WR10.1 = R2
WR10.2 = WN20.A
```

[Program description]

At the rising edge of R0,

- a state of M0 is substituted for R0.
- a value of WN100 is substituted for WM10.
- a value of DY30 is substituted for DN200.
- a state of the 0th bit of WR10 is substituted for R1.
- a state of R2 is substituted for the 1st bit of WR10
- a state of the Ath bit (the 10th bit) of WN20 is substituted for the 2nd bit of WN10.

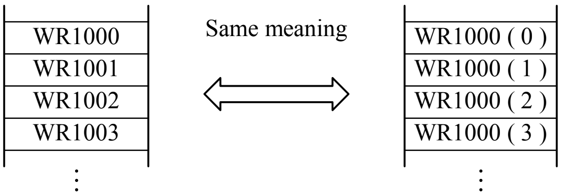
Reference Array designation for Substitution statement

Array is a means to change I/O specified by the index dynamically. This is convenient when setting a value, updating I/O address using FOR sentence.

Index value is specified by a constant and word I/O (WX, WY, WEX, WEY, WR, WM, WL, WN). In addition, the commands which can use an array variable are only a substitution statement, MOV, and COPY.

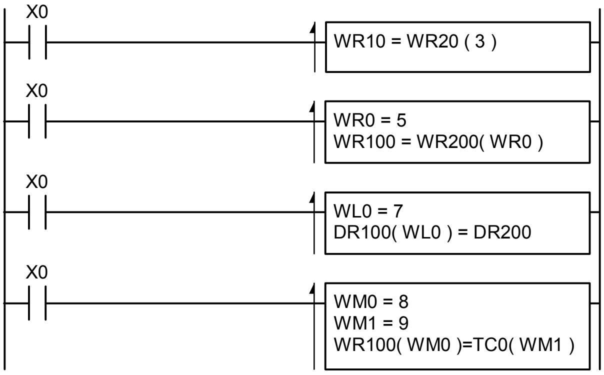
(1) Meaning of Array variable

Array variable is represented by the form of '□ a(b)'. '□' represents I/O type, and 'a' represents I/O address, and 'b' represents a constant or word I/O. And '□ a' is called "I/O of array variable" and 'inner b of ( )' is called "Content of index".



(When 'B' part is a constant, '□ a(b)' means '□ a + b'.)

(2) Example



```
WR10 = WR20 ( 3 )
```
➔ WR10 = WR23

```
WR0 = 5
WR100 = WR200( WR0 )
```
➔ WR100 = WR205

```
WL0 = 7
DR100( WL0 ) = DR200
```
➔ DR107 = DR200

```
WM0 = 8
WM1 = 9
WR100( WM0 )=TC0( WM1 )
```
➔ WR108 = TC9

(3) Caution

- An index is zero or a positive integer. Minus cannot be specified.
- An array variable can be used only for the substitution statement. It is impossible to use as follows,

  WR10(WR20) = WR100 + 1

  R0 = WR10(WR20) < WR30
- An array of Bit in Word cannot be used. It is impossible to use as follows,

  WR10.8 (WR20) = 1

  R0 = WR10.0 (WR20)
- An array of Bit extension XY cannot be used. It is impossible to use as follows,

  EY100 (WR20) = 1

  M0 = EX0.0 (WR20)

Name	Substitution statement (Signed integer)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.S = s.S	(See Command processing time column)		↕	●	●	●	●

Command processing time (μs)					
Maximum					
d.S	s.S	d.S, s.S: Word		d.S, s.S: Double word	
I/O	I/O / C	0.06	[3]	0.46	[5]
Array (W / C)	I/O / C	1.60	[5]	1.92	[6]
Array (DW)	I/O / C	1.68	[6]	1.98	[7]
I/O	Array (W / C)	1.98	[5]	2.60	[5]
I/O	Array (DW)	2.04	[6]	2.66	[6]
Array (W / C)	Array (W / C)	3.12	[6]	3.44	[6]
Array (DW)	Array (W / C)	3.18	[7]	3.50	[7]
Array (W / C)	Array (DW)	3.18	[7]	3.50	[7]
Array (DW)	Array (DW)	3.18	[8]	3.56	[7]

Remarks

- W means Word I/O.
- DW means Double I/O.
- C means a constant.
- [ ]; number of steps (in Processing time column)

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
d.S	Substitution destination								✓	✓	✓	✓		✓	✓	✓	
s.S	Substitution source							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- A content of ‘s.S’ is substituted for ‘d.S’.
- An array variable can be used for ‘d.S’ and ‘s.S’.
- A constant can be used in the following range.

  at Word −32,768 - 32,767 (decimal), H8000 - H7FFF (hexadecimal)

  at Double word −2,147,483,648 - 2,147,483,647 (decimal), H80000000 - H7FFFFFFF (hexadecimal)
- The combination of ‘d.S’ and ‘s.S’ are as follows.

d.S	s.S
Word	Word
Double word	Double word

Cautionary notes

- When using an array variable, DER is set to ‘1’ if it exceeds the maximum of usable I/I number, and DER is set to ‘0’ if it is normal.
- A type is not converted tin the substitution statement. Although description to substitute an internal output with an extension (.S, .FL) for an internal output with other extension by changing / deleting the extension temporary is possible, 16 bits (or 32 bits) data is substituted without changing.

  e.g. When DR0.FL = −259（HC3818000),

  DM0.S = DR0.S

  (If DR0.FL and DM0.S are monitored by hexadecimal, ‘HC3818000’ are both displayed. But if it is done by decimal ‘-269’ is displayed in DR0.FL and ‘-1014923264’ is displayed in DM0.S.)
- If a constant is used for an index of the array, the constant is valid in the range 0 to 65,535 (decimal) and H0000 to HFFFF (hexadecimal)

Name	Substitution (Floating decimal point)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = s.FL	(See Command processing time column)		↕	●	●	●	●

Command processing time (μs)				Remarks
Maximum				• Floating decimal point is specified by Double word. • W means Word I/O. • DW means Double I/O. • C means a constant. The maximum of constant is 20 digits. • [ ]; number of steps (in Processing time column)
d.FL	s.FL	d.FL, s.FL: Double word		
I/O	I/O / C	0.46	[5]	
Array (W / C)	I/O / C	1.92	[6]	
Array (DW)	I/O / C	1.98	[7]	
I/O	Array (W / C)	2.60	[5]	
I/O	Array (DW)	2.66	[6]	
Array (W / C)	Array (W / C)	3.44	[6]	
Array (DW)	Array (W / C)	3.50	[7]	
Array (W / C)	Array (DW)	3.50	[7]	
Array (DW)	Array (DW)	3.56	[7]	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEX, DEY	DR, DL, DM	
d.FL	Substitution destination														✓	✓	✓	
s.FL	Substitution source													✓	✓	✓	✓	✓

Function

- A content of ‘s.FL’ is substituted for ‘d.FL’.
- An array variable can be used for ‘d.FL’ and ‘s.FL’.
- A constant can be use in the following range,

  d.FL, s.FL -3.40282×10^{38} - 3.40282×10^{38} (decimal),

  HFF7FFFFF - H80800000, H00800000 - H7F7FFFFF (hexadecimal)
- The combination of ‘d.FL’ and ‘s.FL’ area as follows.

d.FL	s.FL
Double word	Double word

Cautionary notes

- When using an array variable, DER is set to ‘1’ if it exceeds the maximum of usable I/O number, and DER is set to’0’ if it is normal.
- A type is not converted in the substitution statement. Although description to substitute an internal output with an extension (.S, .FL) for an internal output other extension by changing / deleting the extension temporarily is possible, 16 bits (or 32 bits) is data is substituted without changing.

  e.g. When DR0.S=1073741824(H40000000),

  DM0.FL=DR0.FL

  (If DR0.S and DM0/FL are monitored by hexadecimal, ‘H40000000’ are both displayed, and if it is done by decimal, ‘1073741824’ is displayed in ‘Dr0.S and ‘2’ is displayed in ‘Dm0.FL’.)
- If a constant is used for an index of the array, the constant is valid from 0 to 65,535 (decimal), and from H0000 to HFFFF (hexadecimal).

Name	Binary addition

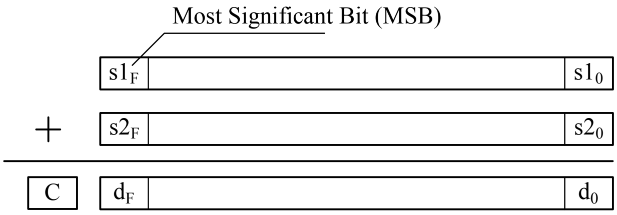
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
d = s1 + s2	Word	4	●	●	●	↕	↕
	Double word	7					

Command processing time (µs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	0.08	—	—	
Double word	0.78	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination								✓	✓	✓	✓		✓	✓	✓	
s1	Augend							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Addend							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- ‘s1’ and ‘s2’ are added as binary data, and the result is substituted for ‘d’ in binary data.



- C flag (Carry: R7F0) is reset to ‘0’ if the operation result is in the following range, and it is set to ‘1’ if it is beyond the following range.

at Word	0 to 65,535 (decimal), H0000 to HFFFF (hexadecimal)
at Double word	0 to 4,294,967,295 (decimal), H00000000 to HFFFFFFFF (hexadecimal)

$$C = s1_F \cdot s2_F + s1_F \cdot \overline{d_F} + s2_F \cdot \overline{d_F}$$

- V flag (Overflow: R7F1) is set to ‘1’ if the operation results are meaningless as signed binary data, and it is reset to ‘0’ if it is meaningful. (See the following table)

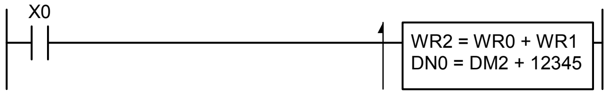
s1	s2	d	V
Positive	Positive	Positive	0
Positive	Positive	Negative	1
Positive	Negative	Positive / Negative	0
Negative	Positive	Negative / Positive	0
Negative	Negative	Positive	1
Negative	Negative	Negative	0

$$V = s1_F \cdot s2_F \cdot \overline{d_F} + \overline{s1_F} \cdot \overline{s2_F} \cdot d_F$$

- The combination of ‘d’ and ‘s’ are as follows.

d	s1	s2
Word	Word	Word
Double word	Double word	Double word

Program example



[Program description]

At the rising edge of X0,

- a value of WR1 is added to a value of WR, and the result is substituted for WR2.
- ‘12345’ is added to a value of DM2 and the result is substituted for DN0.

Name	Binary addition (Singed integer)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.S = s1.S + s2.S	Word	4	●	●	●	↕	↕
	Double word	7					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	0.08	–	–	
Double word	0.82	–	–	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.S	Substitution destination									✓	✓	✓	✓		✓	✓	✓	
s1.S	Augend								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Addend								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- ‘s1.S’ and ‘s2.S’ are added as signed binary data, and the result is substituted for ‘d.S’ in binary data.
- C flag (Carry: R7F0) is reset to ‘0’ if the results are in the following range, and it is set to ‘1’ if it is beyond the range.

  at Word −32,768 to 32,767 (decimal), H8000 to H7FFF (hexadecimal)

  at Double word −2,147,483,648 to 2,147,483,647 (decimal), H80000000 to H7FFFFFFF (hexadecimal)

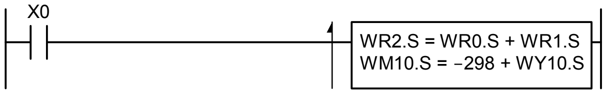
  $C = s1_F \cdot s2_F + s1_F \cdot \overline{d_F} + s2_F \cdot \overline{d_F}$
- A control of V flag (Overflow: R7F1) is same as the binary addition (d=s1+s2).
- The combination of ‘d’ and ‘s’ are as follows.

D	s1	s2
Word	Word	Word
Double word	Double word	Double word

Cautionary notes

An extension “.S” is necessary for ‘d’, ‘s1’, and ‘s2’.

Program example



```
WR2.S = WR0.S + WR1.S
WM10.S = −298 + WY10.S
```

[Program description]

At the rising edge of X0,

- a value of WR1.S is added to a value of WR0.S, and the result is substituted for WR2.S.
- a value of WY10.S is added to ‘-298’ and the result is substituted for WM10.S.
  (A value which subtracted 298 from WY10.S is substituted for WM10.S.)

Name	BCD addition

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = s1 B+ s2	Word	5	↕	●	●	●	↕
	Double word	7					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	3.18	—	—	
Double word	5.64	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination								✓	✓	✓	✓		✓	✓	✓	
s1	Augend							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Addend							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- ‘s1’ and ‘s2’ are added as BCD data, and the result is substituted for ‘d’ in BCD data.
- If a carry is in an operation result, C flag (Carry: R7F0) is set to ‘1’, and if no carry is in, it is reset to ‘0’.
- DER flag (Data error: R7F4) is set to ‘1’ if a content of s1 or s2 is not correct as BCD data. In this case, the output to ‘d’ is not performed remaining the state of C unchanged as a result of not performing the operation.

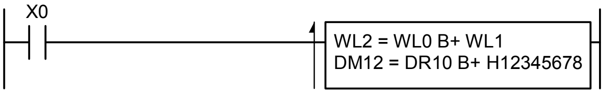
  If calculation is correct, the operation result is output to ‘d’ as a result of resetting DER flag to ‘0’.
- ‘s1’ and ‘s2’ are valid in the following range.

at Word	H0000 to 9999 (BCD)
at Double word	H00000000 to 99999999 (BCD)

- The combination of ‘d’ and ‘s’ are as follows.

d	s1	s2
Word	Word	Word
Double word	Double word	Double word

Program example



[Program description]

At the rising edge of X0,

- a value of WL1 is added to a value of WL0 and the result is substituted for WL2 in BCD data.
- ’H12345678’ is added to a value of DR10 and the result is substituted for DM12.

Name	Floating decimal point addition

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = s1.FL + s2.FL	–	7	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		• Floating decimal point is specified by Double word. • The maximum of constant is 20 digits.
Condition	Time	Condition	Time	
–	19	–	–	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.FL	Substitution destination														✓	✓	✓	
s1.FL	Augend													✓	✓	✓	✓	✓
s2.FL	Addend													✓	✓	✓	✓	✓

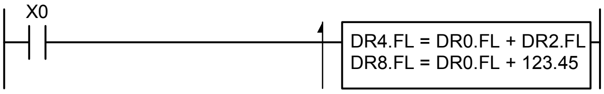
Function

- ‘s1.FL’ and ‘s2.FL’ are added as floating decimal point data, and the result is substituted for ‘d.FL’ in floating decimal point data.
- DER flag (Data error: R7F4) is set to ‘1’ if a content of ‘s1.FL’ or ‘s2.FL’ is not correct as floating decimal point.
  In this case, the output to ‘d.FL’ is not performed as a result of not performing the operation.
  If operation is correct, the operation result is output to ‘d.FL’ as a result of resetting DER flag to ‘0’.

Cautionary notes

- If the operation result is outside the range from −1e+37 to 1e+37, the operation is not performed because of DER=1.
- An extension “.FL“ is necessary for ‘d’, ‘s1’ and ‘s2’.

Program example



[Program description]

At the rising edge of X0,

- a value of DR2.FL is added to a value of DR0.FL and the result is substituted for DR4.FL.
- ‘123.45’ is added to a value of DR0.FL and the result is substituted for DR8.FL.

Name	Binary subtraction

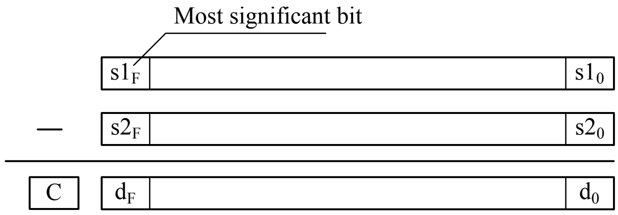
Ladder format	Number of steps		Condition code				
d = s1 – s2	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	Word	4	●	●	●	↕	↕
	Double word	7					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	0.08	—	—	
Double word	0.82	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination								✓	✓	✓	✓		✓	✓	✓	
s1	Minuend							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Subtrahend							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1’ and ‘s2’ as binary data, ‘s2’ is subtracted from ‘s1’. And the result is substituted for ‘d’ in binary data.



- C flag (Carry: R7F0) is set to ‘1’, if a carry-down occurred in the operation result. It is reset to ‘0’ if there is not a carry-down.

  When $s1 < s2$, ‘1’ is set in the operation result C.

  When $s1 \geqq s2$, ‘0’ is set in the operation result C.

  $C = \overline{s1_F} \cdot s2_F + \overline{s1_F} \cdot d_F + s2_F \cdot d_F$

- V flag (Overflow: R7F1) is set to ‘1’ if the operation result is meaningless as signed binary data, and it is reset to ‘0’ if it is meaningful. (See the following tabel)

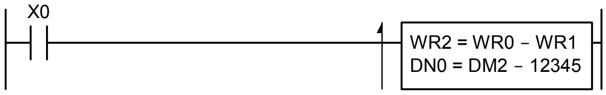
s1	s2	d	V
Positive	Positive	Positive / Negative	0
Negative	Negative	Positive / Negative	0
Positive	Negative	Positive	0
Positive	Negative	Negative	1
Negative	Positive	Positive	1
Negative	Positive	Negative	0

$V = \overline{s1_F} \cdot s2_F \cdot d_F + \overline{s1_F} \cdot \overline{s2_F} \cdot d_F$

- The combination of ‘d’ and ‘s’ are as follows.

D	s1	s2
Word	Word	Word
Double word	Double word	Double word

Program example



[Program description]

At the rising edge of X0,

- a value of WR1 is subtracted from a value of WR0 and the result is substituted for WR2.
- ‘12345’ is subtracted from a value of DM2 and the result is substituted for DN0.

Name	Binary subtraction (Signed integer)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.S = s1.S − s2.S	Word	4	●	●	●	↕	↕
	Double word	7					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	0.08	—	—	
Double word	0.86	—	—	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.S	Substitution destination									✓	✓	✓	✓		✓	✓	✓	
s1.S	Minuend								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Subtrahend								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1.S’ and ‘s2.S’ as signed binary data, ’s2.S’ is subtracted from ‘s1.S’ and the result is substituted for ‘d.S’ in signed binary data.
- C flag (Carry: R7F0) is set to ‘1’ if a carry-down occurred in the operation result, and it is reset to ‘0’ if there is no carry-down.

  When s1.S < s2.S, ‘1’ is set in the operation result C.

  When s1.S≧s2.S, ‘0’ is set in the operation result C.

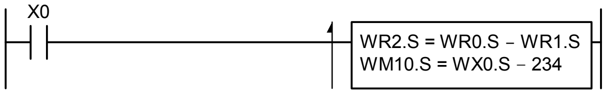
  $C = \overline{s1}_F \cdot s2_F + \overline{s1}_F \cdot d_F + s2_F \cdot d_F$
- A control of V flag (Overflow: R7F1) is same as the binary subtraction (d=s1−s2).
- The combination of ‘d’ and ‘s’ are as follows.

d	s1	s2
Word	Word	Word
Double word	Double word	Double word

Cautionary notes

“.S” is necessary for ‘d’, ‘s1’, and ‘s2’.

Program example



```
WR2.S = WR0.S – WR1.S
WM10.S = WX0.S – 234
```

[Program description]

At the rising edge of X0,

- a value of WR1.S is subtracted from a value of WR0.S and the result is substituted for WR2.S.
- ‘234’ is subtracted from a value of WX0.S and the result is substituted for WM10.S.

Name	BCD subtraction

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = s1 B- s2	Word	5	↕	●	●	●	↕
	Double word	7					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	2.84	—	—	
Double word	4.94	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination								✓	✓	✓	✓		✓	✓	✓	
s1	Minuend							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Subtrahend							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

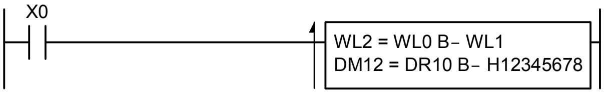
- Treating ‘s1’ and ‘s2’ as BCD data, ‘s2’ is subtracted from ‘s1’ and the result is substituted for ‘d’ in BCD data.
- C flag (Carry: R7F0) is set to ‘1’ if a carry-down occurred in the operation result, and it is reset to ‘0’ if there is no carry-down.
- DER flag (Data error: R7F4) is set to ‘1’ if a content of ‘s1’ or ‘s2’ is not correct as BCD data. In this case, the output to ‘d’ is not performed remaining the state of C unchanged as a result of not performing the operation.
  If calculation is correct, the operation result is output to ‘d’ as a result of resetting DER flag to ‘0’.
- ‘s1’ and ‘s2’ are valid in the following range,

at Word	H0000 to 9999 (BCD)
at Double word	H00000000 to 99999999 (BCD)

- The combination of ‘d’ and ‘s’ are as follows.

d	s1	s2
Word	Word	Word
Double word	Double word	Double word

Program example



[Program description]

At the rising edge of X0,

- a value of WL1 is subtracted from a value of WL0 and the result is substituted for WL2 in BCD data.
- ‘H12345678’ is subtracted from a value of DR10 and the result is substituted for DM12.

Arithmetic

d = s1 B- s2

Name	Floating decimal point subtraction

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = s1.FL – s2.FL	—	7	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		• Floating decimal point is specified by Double word. • The maximum of constant is 20 digits.
Condition	Time	Condition	Time	
—	19	—	—	

Usable I/O		Bit							Word						Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.FL	Substitution destination														✓	✓	✓	
s1.FL	Minuend													✓	✓	✓	✓	✓
s2.FL	Subtrahend													✓	✓	✓	✓	✓

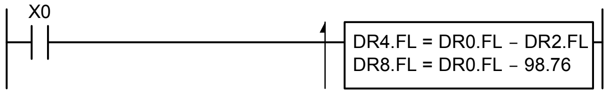
Function

- Treating ‘s1.FL’ and ‘s2.FL’ as floating decimal point data, ‘s2.FL’ is subtracted from ‘s1.FL’ and the result is substituted for ‘d.FL’ in floating decimal point data.
- DER flag (Data error: R7F4) is set to ‘1’ if a content of ‘s1.FL’ or ‘s2.FL’ is not correct as floating decimal point data. In this case, the output to ‘d.FL’ is not performed as a result of not performing the operation.
  If calculation is correct, the operation result is output to ‘d.FL’ as a result of resetting DER flag to ‘0’.

Cautionary notes

- If the operation result is outside the range from −1e+37 to 1e+37, the operation is not performed because of DER=1.
- “.FL” is necessary for ‘d’, ‘s1’, and ‘s2’.

Program example



[Program description]

At the rising edge of X0,

- a value of DR2.FL is subtracted from a value of DR0.FL and the result is substituted for DR4.FL.
- ’98.76’ is subtracted from a value of DR0.FL and the result is substituted for DR8.FL.

Name	Binary multiplication

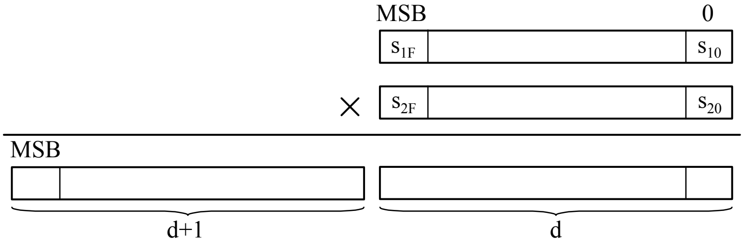
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = s1 * s2	Word	5	●	●	●	●	●
	Double word	7					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	8.30	—	—	
Double word	27.98	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination								✓	✓	✓	✓		✓	✓	✓	
s1	Multiplicand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Multiplier							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Multiplication of ‘s1’ and ‘s2’ are performed as binary data, and the results are substituted for ‘d+1’ (upper) and ‘d’ (lower) in binary data.



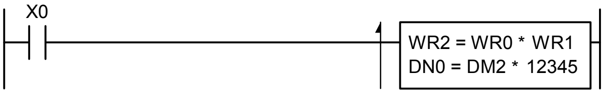
- The combination of ‘d’ and ‘s’ are as follows.

d	s1	s2	Remarks
Word	Word	Word	Calculation is stored in 2 words.
Double word	Double word	Double word	Calculation is stored in 2 double words.

Cautionary notes

- The operation results are substituted for ‘d’ and ‘d+1’. Attention is required if Word or Double word of ‘d+1’ is used for other purpose.
- If ‘d+1’ exceeds I/O range, inputting a circuit is impossible.

Program example



[Program description]

At the rising edge of X0,

- multiplication of a value of WR0 and a value of WR1 is performed, and the result is substituted for WR2 and WR3 (DR2).
- multiplication of a value of DM2 and ‘12345’ is performed, and the result is substituted for DN0 and DN2.

Name	Binary multiplication (Signed integer)

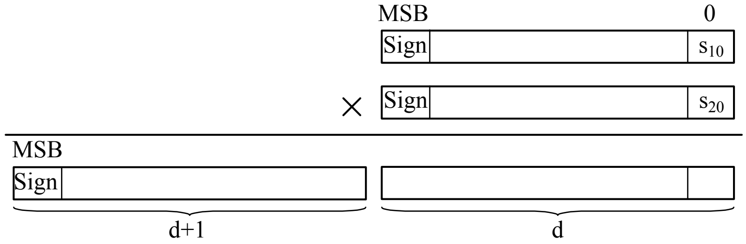
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
d.S = s1.S * s2.S	Word	5	●	●	●	●	●
	Double word	7					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	8.86	–	–	
Double word	28.70	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.S	Substitution destination								✓	✓	✓	✓		✓	✓	✓	
s1.S	Multiplicand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Multiplier							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Multiplication of ‘s1.S’and ‘s2.S’ is performed as signed binary data, the result is substituted for ‘d+1.S’ (upper) and ‘d.S’ (lower) in signed binary data.



- Sign of the operation result is stored in Most significant bit of ‘d+1’.
- ‘s1.S’ and ‘s2.S’ are valid in the following range.

at Word −32,768 to 32,767 (decimal), H8000 to H7FFF (hexadecimal)

at Double word −2,147,483,648 to 2,147,483,647 (decimal), H80000000 to H7FFFFFFF (hexadecimal)

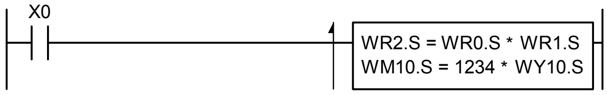
- The combination of ‘d’ and ’s’ are as follows.

d	s1	s2	Remarks
Word	Word	Word	Calculation is stored in 2 words.
Double word	Double word	Double word	Calculation is stored in 2 double words.

Cautionary notes

- “.S” is necessary for ‘d’, ‘s1’, and ‘s2’.
- The operation result is substituted for ‘d’ and ‘d+1’. Attention is required when Word or Double word of ‘d+1’ is used for other purpose.
- If ‘d+1’ exceeds I/O range, inputting a circuit is impossible.

Program example



[Program description]

At the rising of X0,

- multiplication of a value of WR0.S and a value of WR1.S is performed, and the result is substituted for WR2 and WR3 (DR2.S).
- multiplication of ‘1234’ and a value of WY10.S is performed, and the result is substituted for WM10 and WM12(DM10.S).

Name	BCD multiplication

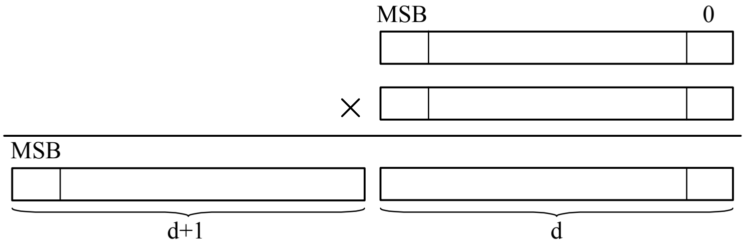
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = s1 B* s2	Word	5	↕	●	●	●	●
	Double word	7					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	15.12	—	—	
Double word	60.34	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination								✓	✓	✓	✓		✓	✓	✓	
s1	Multiplicand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Multiplier							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Multiplication of ‘s1’ and ‘s2’ is performed as BCD data, the result is substituted for ‘d+1’ (upper) and ‘d’ (lower) in BCD data.



- If a content of ‘s1’ or ‘s2’ is not correct as BCD data, the operation is not performed as a result of setting ‘1’ to DER flag (Data error: R7F4). If ‘s1’ and ‘s2’ are correct as BCD data, the operation result is output to ‘d’ as a result of resetting ‘0’.
- ‘s1’ and ‘s2’ are valid in the following range.

at Word	H0000 to 9999 (BCD)
at Double word	H00000000 to 99999999 (BCD)

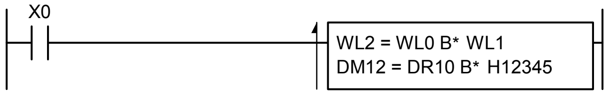
- The combination of ‘d’ and ’s’ are as follows.

d	s1	s2	Remarks
Word	Word	Word	Calculation is stored in 2 words.
Double	Double word	Double word	Calculation is stored in 2 Doube word.

Cautionary notes

- The operation result is substituted for ‘d’ and ‘d+1’. Attention is required when Word or Double word of ‘d+1’ is used for other purpose.
- If ‘d+1’ exceeds I/O range, inputting a circuit is impossible.

Program example



[Program description]

At the rising edge of X0,

- multiplication of a value of WL0 and a value of WL1 is performed, and the result is substituted for WL2 and WL3 (DL2) in BCD data.
- multiplication of a value of DR10 and ‘12345’ is performed, and the result is substituted for DM12 and DM14 in BCD data.

Arithmetic

d.FL = s1.FL * s2.FL

Name	Floating decimal point multiplication

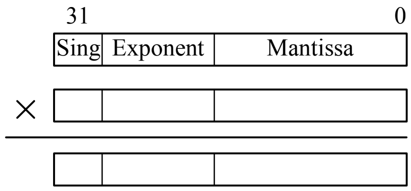
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = s1.FL * s2.FL	—	7	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		• Floating decimal point is specified by Double word. • The maximum of constant is 20 digits.
Condition	Time	Condition	Time	
—	17	—	—	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.FL	Substitution destination														✓	✓	✓	
s1.FL	Multiplicand													✓	✓	✓	✓	✓
s2.FL	Multiplier													✓	✓	✓	✓	✓

Function

- Multiplication of ‘s1.FL’ and ‘s2.FL’ is performed as floating decimal point data, and the result is substituted for ‘d.FL’ in floating decimal points data.

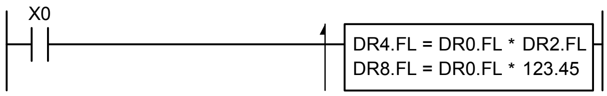


- DER flag (Data error: R7F4) is set to ‘1’ if a content of ‘s1.FL’ or ‘s2.FL’ is not correct as floating decimal point data. In this case, the output to ‘d.FL’ is not performed as a result of not performing the operation.
  If calculation is correct, the operation is output to ‘d.FL’ as a result of resetting DER flag to ‘0’.

Cautionary notes

- If the operation result is outside the range from −1e+37 to 1e+37, the operation is not performed because of DER=1.
- An extension “.FL” is necessary for ‘d’, ‘s1’, and ‘s2’.

Program example



```
DR4.FL = DR0.FL * DR2.FL
DR8.FL = DR0.FL * 123.45
```

[Program description]

At the rising edge of X0,

- multiplication of a value of DR0.FL and a value of DR2.FL is performed, and the result is substituted for DR4.FL.
- multiplication of a value of DR0.FL and ‘123.45’ is performed, and the result is substituted for DR8.FL.

Name	Binary division

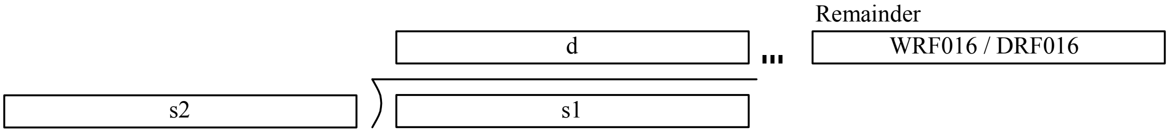
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
d = s1 / s2	Word	5	↕	●	●	●	●
	Double word	7					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	4.78	—	—	
Double word	10.42	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination								✓	✓	✓	✓		✓	✓	✓	
s1	Dividend							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Divisor							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1’ and ‘s2’ as binary data, ‘s1’ is divided by ‘s2’. And the result is substituted for ‘d’ in binary data. The remainder (s1 mod s2) is stored in a special internal output WRF016 at Word and DRF016 at Double word.
- DER flag (Data error: R7F4) is set to ‘1’ if s2 = 0. The operation is not performed. If s2≠0, the operation is performed as a result of resetting DER flag to ‘0’.



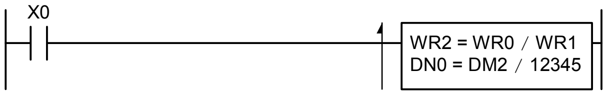
- The combination of ‘d’ and ‘s’ are as follows.

d	s1	s2
Word	Word	Word
Double word	Double word	Double word

Cautionary notes

- If ‘s1’ and ‘s2’ are Word, a special internal output WRF0107 in which the remainder of division is stored is not used. (The value before operation remains unchanged.)

Program example



```
WR2 = WR0 / WR1
DN0 = DM2 / 12345
```

[Program description]

At the rising edge of X0,

- a value of WR0 is divided by a value of WR1, the result is substituted for WR2. The remainder is substituted for a special internal output WRF016.
- a value of DM2 is divided by ‘12345’, and the result is substituted for DN0. The remainder is substituted for a special internal output DRF016.

Name	Binary division (Singed integer)

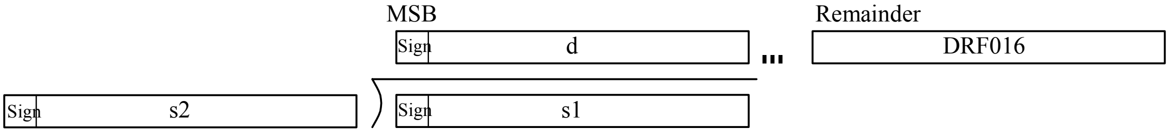
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
d.S = s1.S / s2.S	Word	5	↕	●	●	↕	●
	Double word	7					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	5.90	–	–	
Double word	12.32	–	–	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.S	Substitution destination									✓	✓	✓	✓		✓	✓	✓	
s1.S	Dividend								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Divisor								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1’ and ‘s2’ as signed binary data, ‘s1’ is divided by ‘s2’. And the result is substituted for ’d’ in signed binary data. The remainder (s1 mod s2) is stored in a special internal output WRF016 at Word operation and DRF016 at Double word operation. (Sing is stored in Most significant bit.)



- DER flag (Data error: R7F4) is set to ‘1’ if s2 = 0. The operation is not performed.
  If s2 ≠ 0, the operation result is performed as a result of resetting DER flag to ‘0’.
- V(Overflow:R7F0) is set to ‘1’ if the quotient is positive and also exceeds ‘H7FFF’ or ‘H7FFFFFFF (hexadecimal)’. All other cases are reset to ‘0’.
- Ranges of ‘s1.S’ and ‘s2.S’ are as follows,

at Word	−32,768 to 32,767 (decimal), H8000 to H7FFF (hexadecimal)
at Double word	−2,147,483,648 to 2,147,483,647 (decimal), H80000000 to H7FFFFFFF (hexadecimal)

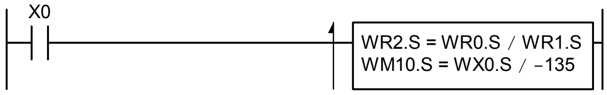
- The combination of ‘d’ and ‘s’ are as follows.

d	s1	s2
Word	Word	Word
Double word	Double word	Double word

Cautionary notes

- An extension “.S” is necessary for ‘d’, ‘s1’, and ‘s2’.
- If ‘s1’ and ‘s2’ are Word, a special internal output WRF017 in which a remainder of division is stored is not used. (The value before operation remains unchanged.)

Program example



[Program description]

At the rising edge of X0,

- a value of WR0.S is divided by a value of WR1.S, and the result is substituted for WR2.S. A remainder is substituted for a special internal output WRF016 in signed binary data.
- a value of WX0.S is divided by '−135', and the result is substituted for WM10.S. A remainder is substituted for a special internal output WRF016 in signed binary data.

Name	BCD division

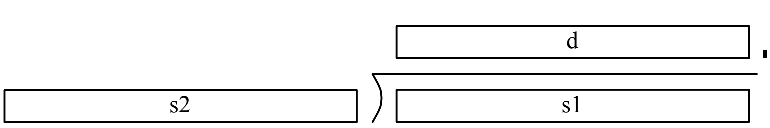
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
d = s1 B/ s2	Word	5	↕	●	●	●	●
	Double word	7					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	9.54	—	—	
Double word	25.16	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination								✓	✓	✓	✓		✓	✓	✓	
s1	Dividend							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Divisor							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1’ and ‘s2’ as BCD data, ‘s1’ is divided by ‘s2’. And the result is substituted for ‘d’ in BCD data.
  A remainder (s1 mod s2) is stored in a special internal output WRF016 at Word and DRF016 at Double word in BCD data



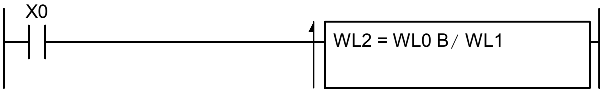
- If a content of ‘s1’ or ‘s2’ is not correct, or if s2 = 0, the operation is not performed as a result of setting ‘1’ to DER flag (Data error: R7F4). If ‘s1’ and ‘s2’ are correct as BCD data and also s2 ≠ 0, the operation is output to ‘d’ as a result of resetting DER flag to ‘0’.
- ‘s1’ and ‘s2’ are valid in the following ranges,

at Word	H0000 to 9999 (BCD)
at Double word	H00000000 to 99999999 (BCD)

- The combination of ‘d’ and ‘s’ are as follows.

d	s1	s2
Word	Word	Word
Double word	Doube word	Double word

Program example



[Program description]

A value of WL0 is divided by a value of WL1 at the rising edge of X0, and the result is substituted for WL2 in BCD data. A remainder is substituted for a special internal output WRF016 in BD data.

Name	Floating decimal point division

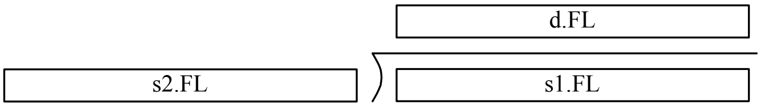
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = s1.FL / s2.FL	—	7	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		• Floating decimal point is specified by Double word. • The maximum of constant is 20 digits.
Condition	Time	Condition	Time	
—	85	—	—	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.FL	Substitution destination														✓	✓	✓	
s1.FL	Dividend													✓	✓	✓	✓	✓
s2.FL	Divisor													✓	✓	✓	✓	✓

Function

- Treating 's1.FL' and 's2.FL' as floating decimal point data, 's1.FL' is divided by 's2.FL'. And the result is substituted for 'd.FL' in floating decimal point data.



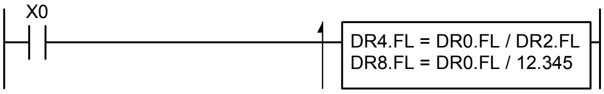
- DER flag (Data error: R7F4) is set to '1' if a content of 's1.FL' or 's2.FL'is not correct as floating decimal point data, or if s2 = 0. In this case, the output to 'd.FL' is not performed as a result of not performing the operation.

  If 's1.FL' and 's2.FL' are correct data and also s2 ≠ 0, the operation result is output to 'd.FL' as a result of resetting DER flag to '0'.

Cautionary notes

- If the operation result is outside the range from −1e+37 to 1e+37, the operation is not performed because of DER=1.
- An extension ".FL" is necessary for 'd', 's1', and 's2'.
- There is no remainder in floating decimal point division.

Program example



[Program description]

At the rising edge of X0,

- a value of DR0.FL is divided by a value of DR2.FL, the result is substituted for DR4.FL.
- a value of DR0.FL is divided by '12.345', and the result is substituted for DR8.FL.

Name	Logical disjunction (OR)

Ladder format	Number of steps				Condition code				
	Condition			Steps ( ) indicates DW	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1 OR s2	I/O	I/O	I/O	3 (6)	●	●	●	●	●
d = $s1.m_1$ OR s2	I/O	I/O.m	I/O	6 (−)					
d = s1 OR $s2.m_2$	I/O	I/O	I/O.m	6 (−)					
d = $s1.m_1$ OR $s2.m_2$	I/O	I/O.m	I/O.m	7 (−)					
$d.m_0$ = s1 OR s2	I/O.m	I/O	I/O	6 (−)					
$d.m_0$ = $s1.m_1$ OR s2	I/O.m	I/O.m	I/O	7 (−)					
$d.m_0$ = s1 OR $s2.m_2$	I/O.m	I/O	I/O.m	7 (−)					
$d.m_0$ = $s1.m_1$ OR $s2.m_2$	I/O.m	I/O.m	I/O.m	8 (−)					

Command processing time (μs)								Remarks
Average				Maximum				When EX and Ey are used, number of steps and processing time are same as I/O.m.
Condition			Time ( ) indicates DW	Condition			Time ( ) indicates DW	
d	s1	s2		d	s1	s2		
I/O	I/O	I/O	0.06 (0.56)	I/O	I/O	I/O	−	
I/O	I/O.m	I/O	0.54	I/O	I/O.m	I/O	−	
I/O	I/O	I/O.m	0.54	I/O	I/O	I/O.m	−	
I/O	I/O.m	I/O.m	0.66	I/O	I/O.m	I/O.m	−	
I/O.m	I/O	I/O	0.62	I/O.m	I/O	I/O	−	
I/O.m	I/O.m	I/O	0.80	I/O.m	I/O.m	I/O	−	
I/O.m	I/O	I/O.m	0.80	I/O.m	I/O	I/O.m	−	
I/O.m	I/O.m	I/O.m	0.90	I/O.m	I/O.m	I/O.m	−	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓			✓	✓	✓	✓	✓		✓	✓	✓	
$d.m_0$	Substitution destination						✓										
s1	Comparand	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
$s1.m_1$	Comparand						✓										
s2	Comparative value	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
$s2.m_2$	Comparative value						✓										

Function

- Logical OR obtained from ‘s1’ and ‘s2’ is substituted for ‘d’.

s1	s2	d
0	0	0
0	1	1
1	0	1
1	1	1

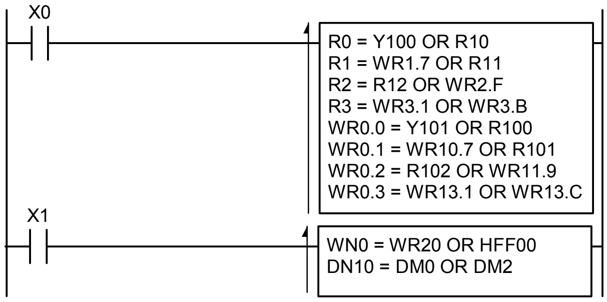
- The combination of ‘d’ and ‘s’ are as follows.

d	s1	s2
Bit	Bit	Bit
	Bit in Word	Bit
	Bit	Bit in Word
	Bit in Word	Bit in Word
Bit in Word	Bit	Bit
	Bit in Word	Bit
	Bit	Bit in Word
	Bit in Word	Bit in Word
Word	Word	Word
Double word	Double word	Double word

Cautionary notes

- Only the word internal output of WR and WN can be specified to ‘d.m_0’, ‘s1.m_1’, and ‘s2.m_2’.
- ‘m_0’, ‘m_1’, and ‘m_2’ are from 0 to F.

Program example



```
R0 = Y100 OR R10
R1 = WR1.7 OR R11
R2 = R12 OR WR2.F
R3 = WR3.1 OR WR3.B
WR0.0 = Y101 OR R100
WR0.1 = WR10.7 OR R101
WR0.2 = R102 OR WR11.9
WR0.3 = WR13.1 OR WR13.C
```

```
WN0 = WR20 OR HFF00
DN10 = DM0 OR DM2
```

[Program description]

At the rising edge of X0,

- logical OR obtained from Y100 and R10 is substituted for R0.
- logical OR obtained from the 7th bit of WR1and R11is substituted for R1.
- logical OR obtained from R12 and the F-th bit (the 15th bit) of WR2 is substituted for R2.
- logical OR obtained from the 1st bit of WR3 and the B-th bit (the 11th bit) of WR3 is substituted for R3.
- logical OR obtained from Y101and R100 is substituted for the 0th bit of WR0.
- logical OR obtained from the 7th bit of WR10 and R101 is substituted for the 1st bit of WR0.
- logical OR obtained from R102 and the 9th bit of WR11 is substituted for the 2nd bit of WR0.
- logical OR obtained from the 1st bit of WR13 and the C-th bit (the 12th bit) of WR13 is substituted for the 3rd bit of WR0.

At the rising edge of X1,

- logical OR obtained from WR20 and HFF00 is substituted for WN0.
- logical OR obtained from DM0 and DM2 is substituted for DN10.

Name	Logical conjunction (AND)

Ladder format	Number of steps				Condition code				
	Condition			Steps ( ) indicates DW	R7F4	R7F3	R7F2	R7F1	R7F0
d = s1 AND s2	d	s1	s2		DER	ERR	SD	V	C
d = $s1.m_1$ AND s2	I/O	I/O	I/O	3 (6)	●	●	●	●	●
d = s1 AND $s2.m_2$	I/O	I/O.m	I/O	6 (−)					
d = $s1.m_1$ AND $s2.m_2$	I/O	I/O	I/O.m	6 (−)					
$d.m_0$ = s1 AND s2	I/O	I/O.m	I/O.m	7 (−)					
$d.m_0$ = $s1.m_1$ AND s2	I/O.m	I/O	I/O	6 (−)					
$d.m_0$ = s1 AND $s2.m_2$	I/O.m	I/O.m	I/O	7 (−)					
$d.m_0$ = $s1.m_1$ AND $s2.m_2$	I/O.m	I/O	I/O.m	7 (−)					
	I/O.m	I/O.m	I/O.m	8 (−)					

Command processing time (μs)								Remarks
Average				Maximum				When EX and EY are used, number of steps and processing time are same as I/O.m.
Condition			Time	Condition			Time	
d	s1	s2	( ) indicates DW	d	s1	s2	( ) indicates DW	
I/O	I/O	I/O	0.06 (0.60)	I/O	I/O	I/O	–	
I/O	I/O.m	I/O	0.54	I/O	I/O.m	I/O	–	
I/O	I/O	I/O.m	0.54	I/O	I/O	I/O.m	–	
I/O	I/O.m	I/O.m	0.70	I/O	I/O.m	I/O.m	–	
I/O.m	I/O	I/O	0.62	I/O.m	I/O	I/O	–	
I/O.m	I/O.m	I/O	0.80	I/O.m	I/O.m	I/O	–	
I/O.m	I/O	I/O.m	0.80	I/O.m	I/O	I/O.m	–	
I/O.m	I/O.m	I/O.m	0.90	I/O.m	I/O.m	I/O.m	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓			✓	✓	✓	✓	✓		✓	✓	✓	
$d.m_0$	Substitution destination						✓										
s1	Comparand	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
$s1.m_1$	Comparand						✓										
s2	Comparative value	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
$s2.m_2$	Comparative value						✓										

Function

- Logical AND obtained from ‘s1’ and ‘s2’ is substituted for ‘d’.

s1	s2	d
0	0	0
0	1	0
1	0	0
1	1	1

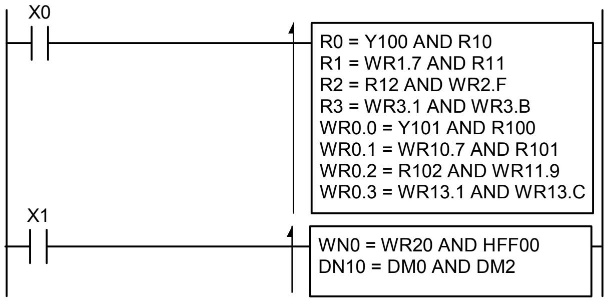
- The combination of ‘d’ and ‘s’ are as follows.

d	s1	s2
Bit	Bit	Bit
	Bit in Word	Bit
	Bit	Bit in Word
	Bit in Word	Bit in Word
Bit in Word	Bit	Bit
	Bit in Word	Bit
	Bit	Bit in Word
	Bit in Word	Bit in Word
Word	Word	Word
Double word	Double word	Double word

Cautionary notes

- Only the word internal output of WR and WN can be specified to ‘d.m$_0$’, ‘s1.m$_1$’, and ‘s2.m$_2$’.
- ‘m$_0$’, ‘m$_1$’, and ‘m$_2$’ are from 0 to F.

Program example



```
R0 = Y100 AND R10
R1 = WR1.7 AND R11
R2 = R12 AND WR2.F
R3 = WR3.1 AND WR3.B
WR0.0 = Y101 AND R100
WR0.1 = WR10.7 AND R101
WR0.2 = R102 AND WR11.9
WR0.3 = WR13.1 AND WR13.C
```

```
WN0 = WR20 AND HFF00
DN10 = DM0 AND DM2
```

[Program description]

At the rising edge of X0,

- logical AND obtained from Y100 and R10 is substituted for R0.
- logical AND obtained from the 7th bit of WR1 and R11 is substituted for R1.
- logical AND obtained from R12 and the F-th bit (the 15th bit) of WR2 is substituted for R2.
- logical AND obtained from the 1st bit of WR3 and the B-th bit (the 11th bit) of WR3 is substituted for R3.
- logical AND obtained from Y101 and R100 is substituted for the 0th bit of WR0.
- logical AND obtained from the 7th bit of WR10 and R101 is substituted for the 1st bit of WR0.
- logical AND obtained from R102 and the 9th bit of WR11 is substituted for the 2nd bit of WR0.
- logical AND obtained from the 1st bit of WR13 and the C-th bit of WR13 is substituted for the 3rd bit of WR0.

At the rising edge of X1,

- logical AND obtained from WR20 and HFF00 is substituted for WN0.
- logical AND obtained from DM0 and DM2 is substituted for DN10.

Name	Exclusive disjunction (XOR)

Ladder format	Number of steps				Condition code				
	Condition			Steps ( ) indicates DW	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1 XOR s2	I/O	I/O	I/O	3 (6)	●	●	●	●	●
d = s1.m$_1$ XOR s2	I/O	I/O.m	I/O	6 (−)					
d = s1 XOR s2.m$_2$	I/O	I/O	I/O.m	6 (−)					
d = s1.m$_1$ XOR s2.m$_2$	I/O	I/O.m	I/O.m	7 (−)					
d.m$_0$ = s1 XOR s2	I/O.m	I/O	I/O	6 (−)					
d.m$_0$ = s1.m$_1$ XOR s2	I/O.m	I/O.m	I/O	7 (−)					
d.m$_0$ = s1 XOR s2.m$_2$	I/O.m	I/O	I/O.m	7 (−)					
d.m$_0$ = s1.m$_1$ XOR s2.m$_2$	I/O.m	I/O.m	I/O.m	8 (−)					

Command processing time (μs)								Remarks
Average				Maximum				When EX and EY are used, number of steps and processing time is same as I/O.m.
Condition			Time	Condition			Time	
d	s1	s2	( ) indicates DW	d	s1	s2	( ) indicates DW	
I/O	I/O	I/O	0.06 (0.60)	I/O	I/O	I/O	−	
I/O	I/O.m	I/O	0.54	I/O	I/O.m	I/O	−	
I/O	I/O	I/O.m	0.54	I/O	I/O	I/O.m	−	
I/O	I/O.m	I/O.m	0.70	I/O	I/O.m	I/O.m	−	
I/O.m	I/O	I/O	0.62	I/O.m	I/O	I/O	−	
I/O.m	I/O.m	I/O	0.80	I/O.m	I/O.m	I/O	−	
I/O.m	I/O	I/O.m	0.80	I/O.m	I/O	I/O.m	−	
I/O.m	I/O.m	I/O.m	0.88	I/O.m	I/O.m	I/O.m	−	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓			✓	✓	✓	✓	✓		✓	✓	✓	
d.m$_0$	Substitution destination						✓										
s1	Comparand	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s1.m$_1$	Comparand						✓										
s2	Comparative value	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.m$_2$	Comparative value						✓										

Function

- Exclusive or (XOR) obtained from ‘s1’ and ‘s2’ is substituted for ‘d’.

s1	s2	d
0	0	0
0	1	1
1	0	1
1	1	0

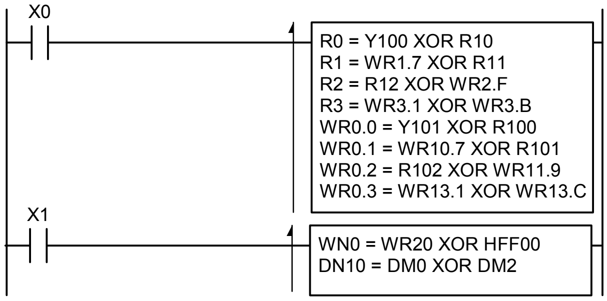
- The combination of ‘d’ and ‘s’ are as follows.

d	s1	s2
Bit	Bit	Bit
	Bit in Word	Bit
	Bit	Bit in Word
	Bit in Word	Bit in Word
Bit in Word	Bit	Bit
	Bit in Word	Bit
	Bit	Bit in Word
	Bit in Word	Bit in Word
Word	Word	Word
Double word	Double word	Double word

Cautionary notes

- Only the word internal output of WR and WN can be specified to ‘d.m_0’, ‘s1.m_1’, and ‘s2.m_2’.
- ‘m_0’, ‘m_1’, and ‘m_2’ are from 0 to F.

Program example



```
R0 = Y100 XOR R10
R1 = WR1.7 XOR R11
R2 = R12 XOR WR2.F
R3 = WR3.1 XOR WR3.B
WR0.0 = Y101 XOR R100
WR0.1 = WR10.7 XOR R101
WR0.2 = R102 XOR WR11.9
WR0.3 = WR13.1 XOR WR13.C
```

```
WN0 = WR20 XOR HFF00
DN10 = DM0 XOR DM2
```

[Program description]

At the rising edge of X0,

- exclusive OR obtained from Y100 and R10 is substituted for R0.
- exclusive OR obtained from the 7th bit of WR1 and R11 is substituted for R1.
- exclusive OR obtained from R12 and the F-th bit (the 15th bit) of WR2 is substituted for R2.
- exclusive OR obtained from the 1st bit of WR3 and the B-th bit (the 11th bit) of WR3 is substituted for R3.
- exclusive OR obtained from Y101 and R100 is substituted for the 0th bit of WR0.
- exclusive OR obtained from the 7th bit of WR10 and R101 is substituted for the 1st bit of WR0.
- exclusive OR obtained from R102 and the 9th bit of WR11 is substituted for the 2nd bit of WR0.
- exclusive OR obtained from the 1st bit of WR13 and the C-th bit (the 12th bit) of WR13 is substituted for the 3rdbit of WR0.

At the rising edge of X1,

- exclusive OR obtained from WR20 and HFF00 is substituted for WN0.
- exclusive OR obtained from DM0 and DM2 is substituted fro DN10.

Arithmetic

Exclusive OR
(d = s1 XOR s2)

Name	= Comparison expression

Ladder format	Number of steps				Condition code				
	Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1 == s2 d.m_0 = s1 == s2	B	W	W	3	●	●	●	●	●
	B	DW	DW	7					
	B(.m)	W	W	6					
	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, W means Word I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as WordI/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:W	0.06	—	—	
d:B / s1, s2:DW	0.68	—	—	
d:B(.m) / s1, s2:W	0.58	—	—	
d:B(.m) / s1, s2:DW	0.80	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
d.m_0	Substitution destination						✓										
s1	Comparand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Comparative value							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1’ and ‘s2’ as binary data, when s1=s2, ‘1’ is substituted for ‘d’, and in all other case ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as binary data, when s1=s2, ‘1’ is substituted for the m-th bit of word data ‘d’, and in all other cases ‘0’ is substituted.
- The combinations of ‘d’, ‘s1’, and ‘s2’ are as follows.

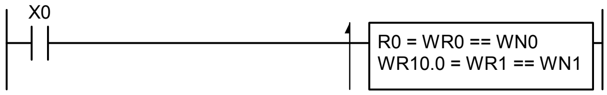
d	s1	s2
Bit	Word	Word
Bit	Double word	Double word

d.m	s1	s2
Word (the m-th bit)	Word	Word
Word (the mth bit)	Double word	Double word

Cautionary notes

- Only the word internal output of WR and WN can be specified to ‘d.m_0’.
- ‘m_0’ is from 0 to F.

Program example



[Program description]

At the rising edge of X0,

- when the value of WR0 and WN0 are the same, R0 is set to ‘1’. In all other cases, it is reset to ‘0’.
- when the value of WR1 and WN1 are the same, the 0th bit of WR10 is set to ‘1’. In all other cases, it is reset to ‘0’.

Name	= Comparison expression (Signed integer)

Ladder format
d = s1.S == s2.S $d.m_0$ = s1.S == s2.S

Number of steps			
Condition			Steps
d	s1	s2	
B	W	W	3
B	DW	DW	7
B(.m)	W	W	6
B(.m)	DW	DW	8

Condition code				
R7F4	R7F3	R7F2	R7F1	R7F0
DER	ERR	SD	V	C
●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, W means Word I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:W	0.06	—	—	
d:B / s1, s2:DW	0.68	—	—	
d:B(.m) / s1, s2:W	0.58	—	—	
d:B(.m) / s1, s2:DW	0.80	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
$d.m_0$	Substitution destination						✓										
s1.S	Comparand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Comparative value							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1’ and ‘s2’ as signed binary data, when s1.S=s2.S, ‘1’ is substituted for ‘d’, and in all other cases ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as signed binary data, when s1.S=s2.S, ‘1’ is substituted for the m-th bit of word data ‘d’, andin all other cases ‘0’ is substituted.
- Ranges of ‘s1.S’ and ‘s2.S’ are as follows.

  at Word −32,768 to 32,767 (decimal), H8000 to H7FFF (hexadecimal)

  at Double word −2,147,483,648 to 2,147,483,647 (decimal), H80000000 to H7FFFFFFF (hexadecimal)
- The combination of ‘d’, ’s1’, and ‘s2’ are as follows.

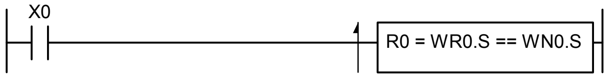
d	s1	s2
Bit	Word	Word
Bit	Double word	Double word

d.m	s1	s2
Word (the m-th bit)	Word	Word
Word (the m-th bit)	Double word	Double word

Cautionary notes

- Only the word internal output can be specified to ‘$d.m_0$’.
- ‘m_0’ is from 0 to F.

Program example



[Program description]

The value of WR0.S and WN0.S are the same at the rising edge of X0, R0 is set to ‘1’. In all other cases, it is reset to ‘0’.

Name	= Comparison expression (Floating decimal point)

Ladder format	Number of steps				Condition code				
	Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1.FL == s2.FL	B	DW	DW	7	●	●	●	●	●
d.m$_0$ = s1.FL == s2.FL	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, and DW means Double word I/O. When EX and EY are used, number of steps and processing time are the same as word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:DW	0.64	–	–	
d:B.m / s1, s2:DW	0.80	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
d.m$_0$	Substitution destination						✓										
s1.FL	Comparand												✓	✓	✓	✓	✓
s2.FL	Comparative value												✓	✓	✓	✓	✓

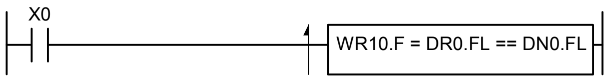
Function

- Treating 's1' and 's2' as floating decimal point data, when s1.FL=s2.FL, '1' is substituted for 'd', and in all other cases '0' is substituted.
- Treating 's1' and 's2' as floating decimal point data, when s1.FL=s2.FL, '1' is substituted for the m-th bit of worddata 'd', and in all other cases '0' is substituted.
- Ranges of 's1' and 's2' -3.40282×10^{38} to 3.40282×10^{38} (decimal),
  HFF7FFFFF to H80800000, H00800000 to H7F7FFFFF (hexadecimal)

Cautionary notes

- Since there is an error in floating decimal point, there are cases where the value disagrees by the error even if the value is in agreement on calculation. We recommend deciding after comparing floating decimal points by not agreement or disagreement, but "range".
- Only the word internal output of WR and WM can be specified to 'd.m$_0$'.
- 'm$_0$' is from 0 to F.

Program example



[Program description]

When the value of DR0.FL and DN0.FL are the same at the rising edge of X0, the F-th bit of WR10 is set to '1'. In all other cases, it is reset to '0'.

Name	<> Comparison expression

Ladder format

d = s1 <> s2
$d.m_0$ = s1 <> s2

Number of steps			
Condition			Steps
d	s1	s2	
B	W	W	3
B	DW	DW	7
B(.m)	W	W	6
B(.m)	DW	DW	8

Condition code				
R7F4	R7F3	R7F2	R7F1	R7F0
DER	ERR	SD	V	C
●	●	●	●	●

Command processing time (μs)			
Average		Maximum	
Condition	Time	Condition	Time
d:B / s1, s2:W	0.06	—	—
d:B / s1, s2:DW	0.68	—	—
d:B(.m) / s1, s2:W	0.58	—	—
d:B(.m) / s1, s2:DW	0.78	—	—

Remarks

B means Bit I/O, W means Word I/O, and DW means Double word I/O.
When EX and EY are used, numberof steps and processing time are the same is Word I/O.m.

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
$d.m_0$	Substitution destination						✓										
s1	Comparand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Comparative value							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1’ and ‘s2’ as binary data, when s1≠s2, ‘1’ is substituted for ‘d’, and in all other cases ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as binary data, when s1≠s2, ‘1’ is substituted for the m-th bit of word data ‘d’, and in all other cases ‘0’ is substituted.
- The combination of ‘d’, ‘s1’ and ‘s2’ are as follows.

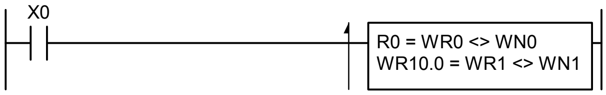
d	s1	s2
Bit	Word	Word
Bit	Double word	Double word

d.m	s1	s2
Word (the m-th bit)	Word	Word
Word (the m-th bit)	Double word	Double word

Cautionary notes

- Only the word internal output of WR and WN can be specified to ‘$d.m_0$’.
- ‘m_0’ is from 0 to F.

Program example



[Program description]

At the rising edge of X0,

- When the value of WR0 and WN0 are different, R0 is set to ‘1’. When the value is the same, it is reset to ‘0’.
- When the value of WR1 and WN1 are different, the 0th bit of WR10 is set to ‘1’. When the value is the same, it is reset to ‘0’.

Name	<> Comparison expression (Signed integer)

Ladder format	Number of steps				Condition code				
	Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1.S <> s2.S $d.m_0$ = s1.S <> s2.S	B	W	W	3	●	●	●	●	●
	B	DW	DW	7					
	B(.m)	W	W	6					
	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, W meand Word I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:W	0.06	–	–	
d:B / s1, s2:DW	0.68	–	–	
d:B(.m) / s1, s2:W	0.58	–	–	
d:B(.m) / s1, s2:DW	0.80	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
$d.m_0$	Substitution destination						✓										
s1.S	Comparand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Comparative value							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1’ and ‘s2’ as signed binary data, when s1.S≠s2.S, ‘1’ is substituted for ‘d’, and in all other cases ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as signed binary data, when s1.S≠s2.S, ‘1’ is substituted for the m-th bit of word data ‘d’, andin all other cases ‘0’ is substituted.
- Ranges of ‘s1.S’ and ‘s2.S’ are as follows,

  at Word −32,768 to 32,767 (decimal), H8000 to H7FFF (hexadecimal)

  at Double word −2,147,483,648 to 2,147,483,647 (decimal), H80000000 to H7FFFFFFF (hexadecimal)
- The combination of ‘d’, ‘s1’ and ‘s2’ are as follows.

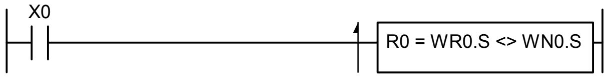
d	s1	s2
Bit	Word	Word
Bit	Double word	Double word

d.m	s1	s2
Word (the m-th bit)	Word	Word
Word (the m-th bit)	Double word	Double word

Cautionary notes

- Only the word internal output can be specified to ‘$d.m_0$’.
- ‘m_0’ is from 0 to F.

Program example



[Program description]

When the value of WR0.S and WN0.S are different at the rising edge of X0, R0 is set to ‘1’. When the value is the same, it is reset to ‘0’.

Name	<> Comparison expression (Floating decimal point)

Ladder format	Number of steps				Condition code				
	Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1.FL <> s2.FL $d.m_0$ = s1.FL <> s2.FL	B	DW	DW	7	●	●	●	●	●
	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:DW	0.68	—	—	
d:B.m / s1, s2:DW	0.78	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
$d.m_0$	Substitution destination						✓										
s1.FL	Comparand												✓	✓	✓	✓	✓
s2.FL	Comparative value												✓	✓	✓	✓	✓

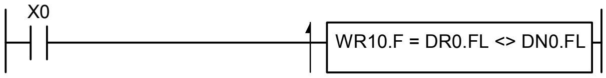
Function

- Treating ‘s1’ and ‘s2’ as floating decimal point data, when s1.FL≠s2.FL, ‘1’ is substituted for ‘d’, and in all other cases, ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as floating decimal point data, when s1.FL≠s2.FL, ‘1’ is substituted for the m-th bit of word data ‘d’, and in all other cases, ‘0’ is substituted.
- Ranges of ‘s1’ and ‘s2’ -3.40282×10^{38} to 3.40282×10^{38} (decimal),
  HFF7FFFFF to H80800000, H00800000 to H7F7FFFFF (hexadecimal)

Cautionary notes

- Since there is an error in floating decimal point, there are cases where the value disagrees by the error even if the value is in agreement on calculation. We recommend decidinf after comparing floating decimal points by not agreement or disagreement, but “range”.
- Only the word internal output of WR and WN can be specified to ‘$d.m_0$’.
- ‘m_0’ is from 0 to F.

Program example



[Program description]

When the value of DR0.FL and DN0.FL are different at the rising edge of X0, the F-th bit of WR10 is set to ‘1’. When the value is the same, it is reset to ‘0’.

Name	< Comparison expression

Ladder format	Number of steps				Condition code				
	Condition			Steps ( ) indicates DW	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1 < s2 d.m_0 = s1 < s2	B	W	W	3	●	●	●	●	●
	B	DW	DW	7					
	B(.m)	W	W	6					
	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, W means Word I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:W	0.06	–	–	
d:B / s1, s2:DW	0.64	–	–	
d:B(.m) / s1, s2:W	0.58	–	–	
d:B(.m) / s1, s2:DW	0.76	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
d.m_0	Substitution destination						✓										
s1	Comparand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Comparative value							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1’ and ‘s2’ as binary data, when s1<s2, ‘1’ is substituted for ‘d’, and in all other cases, ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as binary data, when s1<s2, ‘1’ is substituted for the m-th bit of word data ‘d’, and in all othe cases, ‘0’ is substituted.
- The combination of ‘d’, ‘s1’ and ‘s2’ are as follows.

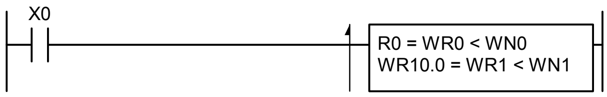
d	s1	s2
Bit	Word	Word
Bit	Double word	Double word

d.m	s1	s2
Word (the m-th bit)	Word	Word
Word (the m-th bit)	Double word	Double word

Cautionary notes

- Only the word internal output of WR and WN can be specified to ‘d.m_0’.
- ‘m_0’ is from 0 to F.

Program example



[Program description]

At the rising edge of X0,

- when WR0<WN0, ‘1’ is set to R0. When WR0≥WN0, R0 is reset to ‘0’.
- when WR1<WN1, the 0th bit of WR10 is set to ‘1’. When WR0≥WN0, the 0th bit of WR10 is reset to ‘0’.

Name	< Comparison expression (Singed integer)

Ladder format	Number of steps				Condition code				
	Condition			Steps ( ) indicates DW	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1.S < s2.S d.m_0 = s1.S < s2.S	B	W	W	5	●	●	●	●	●
	B	DW	DW	7					
	B(.m)	W	W	6					
	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, W means Word I/O, and DW means Double word I/O. When EX and EY are used, number of steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:W	0.66	—	—	
d:B / s1, s2:DW	0.82	—	—	
d:B(.m) / s1, s2:W	0.82	—	—	
d:B(.m) / s1, s2:DW	0.98	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
d.m_0	Substitution destination						✓										
s1.S	Comparand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Comparative value							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1’ and ‘s2’ as signed binary data, when s1.S<s2.S, ‘1’ is substituted for ‘d’, and in all other cases, ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as signed binary data, when s1.S<s2.S, ‘1’ is substituted for the m-th bit of word data ‘d’, and in all other cases, ‘0’ is substituted.
- Ranges of ‘s1.S’ and ‘s2.S’ are as follows,

  at Word −32,768 to 32,767 (decimal), H8000 to H7FFF (hexadecimal)

  at Double word −2,147,483,648 to 2,147,483,647 (decimal), H80000000 to H7FFFFFFF (hexadecimal)
- The combination of ‘d’, ‘s1’ and ‘s2’ are as follows.

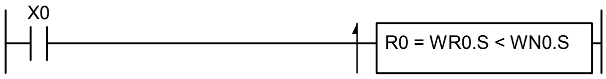
d	s1	s2
Bit	Word	Word
Bit	Double word	Double word

d.m	s1	s2
Word (the m-th bit)	Word	Word
Word (the m-th bit)	Double word	Double word

Cautionary notes

- Only the word internal output of WR and WN can be specified to ‘d.m_0’.
- ‘m_0’ is from 0 to F.

Program example



[Program description]

When WR0.S<WN0.S at the rising edge of X0, R0 is set to ‘1’. When WR0.S≥WN0.S, R0 is reset to ‘0’.

Name	< Comparison expression (Floating decimal point)

Ladder format	Number of steps				Condition code				
	Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
d = s1.FL < s2.FL $d.m_0$ = s1.FL < s2.FL	d	s1	s2		DER	ERR	SD	V	C
	B	DW	DW	7	●	●	●	●	●
	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		Bmeans Bit I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:DW	2.52	–	–	
d:B.m / s1, s2:DW	2.72	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
$d.m_0$	Substitution destination						✓										
s1.FL	Comparand												✓	✓	✓	✓	✓
s2.FL	Comparative value												✓	✓	✓	✓	✓

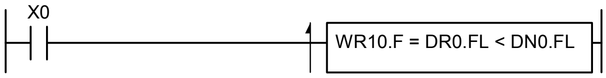
Function

- Treating 's1' and 's2' as floating decimal point data, when s1.FL<s2.FL, '1' is substituted for 'd', and in all other cases, '0' is substituted.
- Treating 's1' and 's2' as floating decimal point data, whens1.FL<s2.FL, '1' is substituted for the m-th bit of word data 'd', and in all other cases, '0' is substituted.
- Ranges of 's1' and 's2' -3.40282×10^{38} to 3.40282×10^{38} (decimal),
  HFF7FFFFF to H80800000, H00800000 to H7F7FFFFF (hexadecimal)

Cautionary notes

- Since there is an error in floating decimal point, there are cases where the value disagrees by the erroro even if thevalue is in agreement on calculation. We recommend deciding after comparing floating decimal points by not agreement or disagreement, but "range".
- Only the word internal output of WR and WN can be specified to '$d.m_0$'.
- 'm_0' is from 0 to F.

Program example



[Program description]

When DR0.FL<DN0.FL, the F-th bit of WR10 is set to '1' at the rising edge of X0.

When DR0.FL≥DN0.FL, the F-th bit of WR10 is reset to '0'.

Name	<= Comparison expression

Ladder format	Number of steps				Condition code				
	Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1 <= s2 d.m$_0$ = s1 <= s2	B	W	W	3	●	●	●	●	●
	B	DW	DW	7					
	B(.m)	W	W	6					
	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, W means Word I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:W	0.06	–	–	
d:B / s1, s2:DW	0.72	–	–	
d:B(.m) / s1, s2:W	0.58	–	–	
d:B(.m) / s1, s2:DW	0.86	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
d.m$_0$	Substitution destination						✓										
s1	Comparand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Comparative value							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1’ and ‘s2’ as binary data, when s1≤s2, ‘1’ is substituted for ‘d’, and in all other cases, ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as binary data, when s1≤s2, ‘1’ is substituted for the m-th bit of word data ‘d’, and in all other cases, ‘0’ is substituted.
- The combination of ‘d’, ‘s1’ and ‘s2’ are as follows.

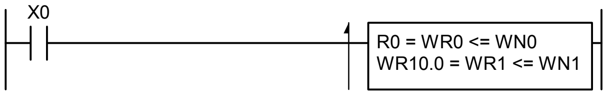
d	s1	s2
Bit	Word	Word
Bit	Double word	Double word

d.m	s1	s2
Word (the m-th bit)	Word	Word
Word (the m-th bit)	Double word	Double word

Cautionary notes

- Only the word internal output of WR and WN can be specified to ‘d.m$_0$’.
- ‘m$_0$’ is from 0 to F.

Program example



[Program description]

At the rising edge of X0,

- when WR0≤WN0, R0 is set to ‘1’. When WR0>WN0, R0 is rest to ‘0’.
- when WR1≤WN1, the 0th bit of WR10 is set to ‘1’. When WR1>WN1, the 0th bit of WR10 is reset to ‘0’.

Name	<= Comparison expression (Signed integer)

Ladder format	Number of steps				Condition code				
	Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1.S <= s2.S $d.m_0$ = s1.S <= s2.S	B	W	W	5	●	●	●	●	●
	B	DW	DW	7					
	B(.m)	W	W	6					
	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, W means Word I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:W	0.68	—	—	
d:B / s1, s2:DW	0.90	—	—	
d:B(.m) / s1, s2:W	0.82	—	—	
d:B(.m) / s1, s2:DW	1.06	—	—	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)		WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓													
$d.m_0$	Substitution destination						✓											
s1.S	Comparand								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Comparative value								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

機　能

- Treating ‘s1’ and ‘s2’ as signed binary data, when s1.S≤s2.S, ‘1’ is substituted for ‘d’, and in all other cases ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as signed binry data, when s1.S≤s2.S, ‘1’ is substituted for the m-th bit of word data ‘d’, in all other cases ‘0’ is substituted.
- Ranges of ‘s1.S’ and ‘s2.S’ are as follows,

at Word −32,768 to 32,767 (decimal), H8000 to H7FFF (hexadecimal)

at Double word −2,147,483,648 to 2,147,483,647 (decimal), H80000000 to H7FFFFFFF (hexadecimal)

- The combination of ‘d’, ‘s1’ and ‘s2’ are as follows.

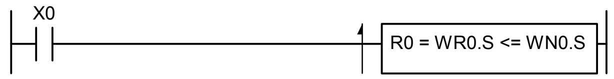
d	s1	s2
Bit	Word	Word
Bit	Double word	Double word

d.m	s1	s2
Word (the m-th bit)	Word	Word
Word (the m-th bit)	Double word	Double word

Cautionary notes

- Only the word internal output of WR and WN can be specified to ‘$d.m_0$’.
- m_0は 0～F までです。

Program example



[Program description]

When WR0.S≤WN0.S, R0 is set to ‘1’ at the rising edge of X0. When WR0.S>WN0.S, R0 is reset to ‘0’.

Name	<= Comparison expression (Floating decimal point)

Ladder format

d = s1.FL <= s2.FL
d.m$_0$ = s1.FL <= s2.FL

Number of steps				Condition code				
Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
d	s1	s2		DER	ERR	SD	V	C
B	DW	DW	7	●	●	●	●	●
B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:DW	2.52	–	–	
d:B.m / s1, s2:DW	2.72	–	–	

Usable I/O		Bit							Word					Double word				
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	Constant
d	Substitution destination		✓	✓	✓													
d.m$_0$	Substitution destination							✓										
s1.FL	Comparand													✓	✓	✓	✓	✓
s2.FL	Comparative value													✓	✓	✓	✓	✓

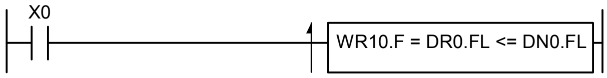
Function

- Treating ‘s1’ and ‘s2’ as floating decimal point data, when s1.FL≤s2.FL, ‘1’ is substituted for ‘d’, and in all other cases, ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as floating decimal point data, when s1.FL≤s2.FL, ‘1’ is substituted for the m-th bit of word data ‘d’, and in all other cases ‘0’ is substituted.
- Ranges of ‘s1’ and ‘s2’ -3.40282×10^{38} to 3.40282×10^{38} (decimal),
  HFF7FFFFF to H80800000, H00800000 to H7F7FFFFF (hexadecimal)

Cautionary notes

- Since there is an error in floating decimal point, there are cases where the value disagrees by the error even if the value is in agreement on calculation. We recommend deciding after comparing floating decimal points by not agreement or disagreement, but “range”.
- Only the word internal output of WR and WN can be specified to ‘d.m$_0$’.
- ‘m$_0$’ is from 0 to F.

Program example



[Program description]

When DR0.FL≤DN0.FL, the F-th bit of WR10 is set to ‘1’ at the rising edge of X0.

When DR0.FL>DN0.FL, the F-th bit of WR10 is reset to ‘0’.

Name	> Comparison expression

Ladder format	Number of steps				Condition code				
	Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1 > s2 $d.m_0$ = s1 > s2	B	W	W	3	●	●	●	●	●
	B	DW	DW	7					
	B(.m)	W	W	6					
	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, W means Word I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:W	0.06	–	–	
d:B / s1, s2:DW	0.66	–	–	
d:B(.m) / s1, s2:W	0.58	–	–	
d:B(.m) / s1, s2:DW	0.82	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
$d.m_0$	Substitution destination						✓										
s1	Comparand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Comparative value							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

機 能

- Treating ‘s1’ and ‘s2’ as binary data, when s1>s2, ‘1’ is substituted for ‘d’, and in all other cases, ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ when s1>s2, ‘1’ is substituted for the m-th bit of word data ‘d’, and in all other cases, ‘0’ is substituted.
- The combination of ‘d’, ‘s1’ and ‘s2’ are as follows.

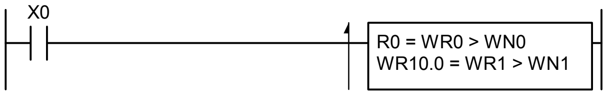
d	s1	s2
Bit	Word	Word
Bit	Double word	Double word

d.m	s1	s2
Word (the m-th bit)	Word	Word
Word (the m-th bit)	Double word	Double word

Cautionary notes

- Only the word internal output of WR and WN can be specified to ‘$d.m_0$’.
- ‘m_0’ is from 0 to F.

Program example



[Program description]

At the rising edge of X0,

- when WR0>WN0, R0 is set to ‘1’. When WR0≤WN0, R0 is reset to ‘0’.
- when WR1>WN1, the 0th bit of WR10 is set to ‘1’. When WR1≤WN1, the 0th bit of WR10 is reset to ‘0’.

Name	> Comparison expression (Singed integer)

Ladder format

d = s1.S > s2.S
$d.m_0$ = s1.S > s2.S

Number of steps				Condition code				
Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
d	s1	s2		DER	ERR	SD	V	C
B	W	W	5	●	●	●	●	●
B	DW	DW	7					
B(.m)	W	W	6					
B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, W means Word I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:W	0.66	–	–	
d:B / s1, s2:DW	0.82	–	–	
d:B(.m) / s1, s2:W	0.82	–	–	
d:B(.m) / s1, s2:DW	0.98	–	–	

Usable I/O		Bit						Word					Double word				
		X	Y	EX EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	Constant
d	Substitution destination		✓	✓	✓												
$d.m_0$	Substitution destination						✓										
s1.S	Comparand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Comparative value							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating 's1' and 's2' as signed binary data, when s1.S>s2.S, '1' is substituted for 'd', and in all other cases, '0' is substituted.
- Treating 's1' and 's2' as signed binary data, when s1.S>s2.S, '1' is substituted for the m-th bit of word data 'd', and in all other cases, '0' is substituted.
- Ranges of 's1.S' and 's2.S' are as follows,

  at Word −32,768 to 32,767 (decimal), H8000 to H7FFF (hexadecimal)

  at Double word −2,147,483,648 to 2,147,483,647 (decimal), H80000000 to H7FFFFFFF (hexadecimal)
- The combination of 'd', 's1' and 's2' are as follows.

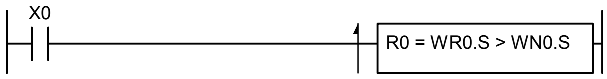
d	s1	s2
Bit	Word	Word
Bit	Double word	Double word

d.m	s1	s2
Word (the m-th bit)	Word	Word
Word (the m-th bit)	Double word	Double word

Cautionary notes

- Only the word internal output of WR and WN can be specified to '$d.m_0$'.
- 'm_0' is from 0 to F.

Program example



[Program description]

When WR0.S>WN0.S, R0 is set to '1' at the rising edge of X0. When WR0.S≤WN0.S, R0 is reset to '0'.

Name	> Comparison expression (Floating decimal point)

Ladder format	Number of steps				Condition code				
	Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1.FL > s2.FL d.m$_0$ = s1.FL > s2.FL	B	DW	DW	7	●	●	●	●	●
	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:DW	2.52	–	–	
d:B.m / s1, s2:DW	2.70	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
d.m$_0$	Substitution destination						✓										
s1.FL	Comparand												✓	✓	✓	✓	✓
s2.FL	Comparative value												✓	✓	✓	✓	✓

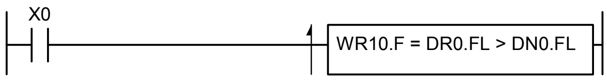
Function

- Treating ‘s1’ and ‘s2’ as floating decimal point data, when s1.FL>s2.FL, ‘1’ is substituted for ‘d’, and in all other cases, ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as floating decimal point data, when s1.FL>s2.FL, ‘1’ is substituted for the m-th bit of word data ‘d’, and in all other cases, ‘0’ is substituted.
- Ranges of ‘s1’ and ‘s2’ -3.40282×10^{38} to 3.40282×10^{38} (decimal),
  HFF7FFFFF to H80800000, H00800000 to H7F7FFFFF (hexadecimal)

Cautionary notes

- Since there is an error in floating decimal point, there are cases where the value disagrees by the error even if the value is in agreement on calculation. We recommend deciding after comparing floating decimal points by not agreement or disagreement, but “range”.
- Only the word internal output of WR and WN can be specified to ‘d.m$_0$’.
- ‘m$_0$’ is from 0 to F.

Program example



[Program description]

When DR0.FL>DN0.FL, the F-th bit of WR10 is set to ‘1’ at the rising edge of X0.

When DR0.FL≤DN0.FL, the F-th bit of WR10 is reset to’0’.

Name	>= Comparison expression

Ladder format	Number of steps				Condition code				
	Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1 >= s2 d.m$_0$ = s1 >= s2	B	W	W	3	●	●	●	●	●
	B	DW	DW	7					
	B(.m)	W	W	6					
	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, W means Word I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:W	0.06	–	–	
d:B / s1, s2:DW	0.72	–	–	
d:B(.m) / s1, s2:W	0.58	–	–	
d:B(.m) / s1, s2:DW	0.84	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
d.m$_0$	Substitution destination						✓										
s1	Comparand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2	Comparative value							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1’ and ‘s2’ as binary data, when s1≥s2, ‘1’ is substituted for ‘d’, and in all other cases, ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as binary data, when s1≥s2, ‘1’ is the m-th bit of word data ‘d’, and in all other cases, ‘0’ is substituted.
- The combination of ‘d’, ‘s1’ and ‘s2’ are as follows.

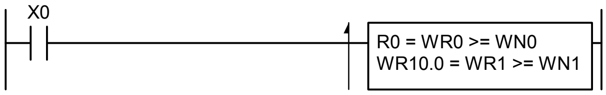
d	s1	s2
Bit	Word	Word
Bit	Double word	Double word

d.m	s1	s2
Word (the m-th bit)	Word	Word
Word (the m-th bit)	Double word	Double word

Cautionary notes

- Only the word internal output of WR and WN can be specified to ‘d.m$_0$’.
- ‘m$_0$’ is from 0 to F.

Program example



[Program description]

At the rising edge of X0,

- when WR0≥WN0, R is set to ‘1’. When WR0<WN0, R0 is reset to ‘0’.
- when WR1≥WN1, the 0th bit of WR10 is set to ‘1’. When WR1<WN1, the 0th bit of WR10 is reset to ‘0’.

Name	>= Comparison expression (Singed integer)

Ladder format	Number of steps				Condition code				
	Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1.S >= s2.S d.m$_0$ = s1.S >= s2.S	B	W	W	5	●	●	●	●	●
	B	DW	DW	7					
	B(.m)	W	W	6					
	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, W means Word I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:W	0.66	–	–	
d:B / s1, s2:DW	0.90	–	–	
d:B(.m) / s1, s2:W	0.82	–	–	
d:B(.m) / s1, s2:DW	1.06	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
d.m$_0$	Substitution destination						✓										
s1.S	Comparand							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
s2.S	Comparative value							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Treating ‘s1’ and ‘s2’ as signed binary data, when s1.S≥s2.S, ‘1’ is substituted for ‘d’, and in all other cases, ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as signed binary data, when s1.S≥s2.S, ‘1’ is substituted for the m-th bit of word data ‘d’, and in all other cases, ‘0’ is substituted.
- Ranges of ‘s1.S’ and ‘s2.S’ are as follows,

  at Word −32,768 to 32,767 (decimal), H8000 to H7FFF (hexadecimal)

  at Double word −2,147,483,648 to 2,147,483,647 (decimal), H80000000 to H7FFFFFFF (hexadecimal)
- The combination of ‘d’, ‘s1’ and ‘s2’ are as follows.

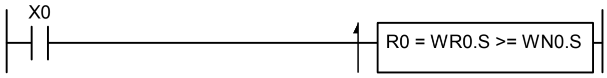
d	s1	s2
Bit	Word	Word
Bit	Double word	Double word

d.m	s1	s2
Word (the m-th bit)	Word	Word
Word (the m-th bit)	Double word	Double word

Cautionary notes

- Only the word internal output of WR and WN can be specified to ‘d.m$_0$’.
- ‘m$_0$’ is from 0 to F.

Program example



[Program description]

When WR0.S≥WN0.S, R0 is set to ‘1’ at the rising edge of C0. When WR0.S<WN0.S, R0 is reset to ‘0’.

Name	>= Comparison expression (Floating decimal point)

Ladder format	Number of steps				Condition code				
	Condition			Steps	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s1	s2		DER	ERR	SD	V	C
d = s1.FL >= s2.FL $d.m_0$ = s1.FL >= s2.FL	B	DW	DW	7	●	●	●	●	●
	B(.m)	DW	DW	8					

Command processing time (μs)				Remarks
Average		Maximum		B means Bit I/O, and DW means Double word I/O. When EX and EY are used, numberof steps and processing time are the same as Word I/O.m.
Condition	Time	Condition	Time	
d:B / s1, s2:DW	2.52	–	–	
d:B.m / s1, s2:DW	2.70	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Substitution destination		✓	✓	✓												
$d.m_0$	Substitution destination						✓										
s1.FL	Comparand												✓	✓	✓	✓	✓
s2.FL	Comparative value												✓	✓	✓	✓	✓

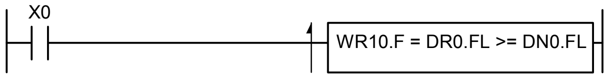
Function

- Treating ‘s1’ and ‘s2’ as floating decimal point data, when s1.FL≥s2.FL, ‘1’ is substituted for ‘d’, and in all other cases, ‘0’ is substituted.
- Treating ‘s1’ and ‘s2’ as floating decimal point data, when s1.FL≥s2.FL, ‘1’ is substituted for the m-th bit of word data ‘d’, and in all other cases, ‘0’ is substituted.
- Ranges of ‘s1’ and ‘s2’ -3.40282×10^{38} to 3.40282×10^{38} (decimal),
  HFF7FFFFF to H80800000, H00800000 to H7F7FFFFF (hexadecimal)

Cautionary notes

- Since there is an error in floating decimal point, there are cases where the value disagrees by the error even if the value is in agreement on calculation. We recommend deciding after comparing floating decimal points by not agreement or disagreement, but “range”.
- Only the word internal output of WR and WN can specified to ‘$d.m_0$’.
- ‘m_0’ is from 0 to F.

Program example



[Program description]

When DR0.FL≥DN0.FL, the F-th bit of WR10 is set to ‘1’ at the rising edge of X0.

When DR0.FL<DN0.FL, the F-th bit of WR10 is reset to ‘0’.

Name	Type Conversion (Floating decimal point ➔ Singed integer)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.S = INTG (s.FL)	—	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
d : Word	10	—	—	
d : Double word	7	—	—	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.S	Conversion result								✓	✓	✓	✓		✓	✓	✓	
s.FL	Conversion source												✓	✓	✓	✓	✓

Function

- Floating decimal point specified by s.FL is converted to integer.
  If d.S is Word, it is converted to word data. If d.S is Double word, it is converted to double word data.
- A form of substitution statement is written and the operation result is stored in d.S.
  e.g.1) WR10.S = INTG (DR0.FL)
  The result which converted DR0.FL (real number) to integer is stored in WR10.S.
  e.g.2) DR10.S = INTG (DR0.FL)
  The result which converted DR0.FL (real number) to integer is stored in DR10.S.

Parameter

d.S: An internal output (Word or Double word) to store result of calculation is specified.

s.FL: An argument is specified.

In case d is Word,

If a value in outside of the range -32,768 < s.FL < 32,767 is specified, the operation is not performed because of DER=1.

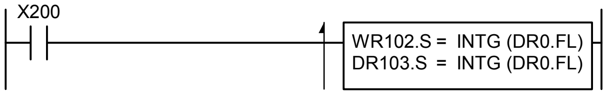
In case d is Double word,

If a value in outside the range -2,147,483,648 < s.FL < 2,147,483,647 is specified, the operation is not performed because of DER=1.

Cautionary notes

- Internal outputs for argument and for storing result of calculation should be specified within I/O number.
- An extension ".FL" is necessary for an internal output of conversion source.
- An extension ".S" is necessary for an internal output of conversion result.
- A format of floating decimal point conforms to IEEE754.

Program example



[Program description]

The result which converted real number specified by DR0.FL to integer is set into WR102 and DR103 at the rising edge of X200. (The figures below a decimal point are omitted.)

If X200 is turned on when DR0.FL is ‘123.456’, both WR102 and DR103 are set to ‘123’.

PRN ➔ PRJ

This command is equivalent to FUN100(s) / FUN101(s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 100 (s) / FUN101 (s) for EHV is as follows.

(1) FUN 100 (s) ➔ [s+2].S = INTG (s.FL), provided that ‘s’ is Double word.

e.g.) FUN 100 (WR100) ➔ WR102.S = INTG (DR100.FL)

(2) FUN 101 (s) ➔ [s+2].S = INTG (s.FL), provided that ‘s’ and ‘s+2’ are Double word.

e.g.) FUN 101 (WM10) ➔ DM12.S = INTG (DM10.FL)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Type conversion (Signed integer ➔ Floating decimal point)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = FLOAT (s.S)	d: Word	4	●	●	●	●	●
	d: Double word	5					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
d : Word	11	—	—	
d : Double word	11	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.FL	Conversion result													✓	✓	✓	
s.S	Conversion source							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Integer specified by s.S is converted to floating decimal point.
- A form of substitution statement is written and the operation result is stored in d.FL.

  e.g.1) DR10.FL=FLOAT (WR0.S)

  The result which converted WR0 (integer) to floating decimal point is stored in DR10.FL.

  e.g.2) DR10.FL=FLOAT (DR0.S)

  The result which converted DR0 (integer) to floating decimal point is stored in DR10.FL.

Parameter

d.FL: An internal output (Double word) to store result of calculation is specified.

s.S: An argument is specified.

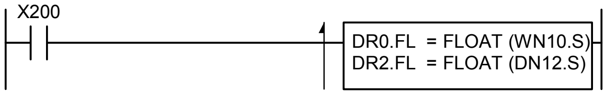
Since negative number is treated as two’s complement, convertible integer is the following ranges.

at Word,	−32,768 to 32,767
at Double word,	−2,147,483,648 to 2,147,483

Cautionary notes

- Internal outputs for argument and for storing result of calculation should be specified within I/O number.
- An extension “.S” is necessary for an internal output of conversion source.
- An extension “.FL” is necessary for an internal output for storing result of calculation.
- A format of floating decimal points conforms to IEEE754.

Program example



[Program description]

The result which converted integer specified by WN10.S to real number is set into DR0.FL at the rising edge of X200. And the result which converted integer specified by DN12.S to real number is set into DR2.FL.

If X200 is turned on when WN10.S is '−123' and DN12.S is '4567890', DR0.FL is set to '−123' and DR2.FL is set to '4567890' in floating decimal point format.

PRN ➔ PRJ

This command is equivalent to FUN102(s) / FUN103(s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 102 (s) / FUN103 (s) for EHV is as follows.

(1) FUN 102 (s) ➔ [s+1].FL = FLOAT (s.S), provided that 's+1' is Double word.

e.g.) FUN 102 (WR100) ➔ DR101.FL = FLOAT (WR100.S)

(2) FUN 103 (s) ➔ [s+2].FL = FLOAT (s.S), provided that 's' and 's+2' are Double word.

e.g.) FUN 103 (WM10) ➔ DM12.S = FLOAT (DM10.FL)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Conversion from Degree to Radian (Floating decimal point)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = RAD (s.FL)	–	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	57	–	–	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.FL	Conversion result (Radian)														✓	✓	✓	
s.FL	Conversion source													✓	✓	✓	✓	✓

Function

- The result which converted Degree which treats a real number value specified by s.FL as an argument to Radian is set into d.FL.
- If the operation is performed normally, DER is ‘0’ (DER=0).

Parameter

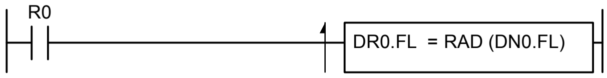
d.FL: An internal output (Double word) to store result of calculation is specified.

s.FL: An argument is specified.

Cautionary notes

- Internal outputs for argument and for storing result of calculation should be specified within I/O number.
- An extension “.FL” is necessary for internal outputs for argument and for storing result of calculation.
- When argument is integer, the integer should be converted to real number before execution of the operation. (Otherwise, you cannot get correct result of calculation.)
- If operation results are in outside the range from −1e+37 to 1e+37, DER is ‘1’ (DER=1).
- A format of floating decimal point conforms to IEEE754.

Program example



[Program description]

The result which converted Degree specified by DN0.FL to Radian is set into DR0.FL at the rising edge of R0.

When DN0.FL is ‘30’, if R0 is turned on, ‘0.5235’ is stored in DR0.FL.

PRN ➔ PRJ

This command is equivalent to FUN108(s) in the program (PRN file) of EH-CPU.
How to convert the program which has used FUN 108 (s) for EHV is as follows.

FUN 108 (s) ➔ [s+2].FL = RAD (s.FL), provided that ‘s’ and ‘s+2’ are Double word.

* If converted by a conversion tool, it is converted as mentioned above.

Name	Conversion from Radian to Degree (Floating decimal point)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = DEG (s.FL)	–	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	43	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.FL	Conversion result (Radian)													✓	✓	✓	
s.FL	Conversion source												✓	✓	✓	✓	✓

Function

- The result which converted Radian which treats a real number specified by s.FL as an argument to Degree is set into d.FL.
- If the operation is performed normally, DER is ‘0’ (DER=0).

Parameter

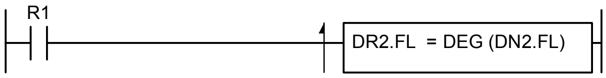
d.FL: A internal output (Double word) to store result of calculation is specified.

s.FL: An argument is specified.

Cautionary notes

- Internal outputs for argument and for storing result of calculation should be specified within I/O number.
- An extension “.FL” is necessary for internal outputs for argument and for storing result of calculation.
- When argument is integer, the integer should be converted to real number before execution of the operation. (Otherwise, you cannot get correct result of calculation.)
- If the operation result is in outside the range from −1e+37 to 1e+37, DER is ‘1’ (DER=1).
- A format of floating decimal point conforms to IEEE754.

Program example



[Program description]

The result which converted Radian specified by DN2.FL to Degree is set into DR2.FL at the rising edge of R1. When DN2.FL is ‘3.14’, if R1 is turned on, ‘179.9’ is stored in DR2.FL.

PRN ➔ PRJ

This command is equivalent to FUN109(s) in the program (PRN file) of EH-CPU.
How to convert the program which has used FUN 109 (s) for EHV is as follows.

FUN 109 (s) ➔ [s+2].FL = DEG (s.FL), provide that ‘s’ and ‘s+2’ are Double word.

* If converted by a conversion tool, it is converted as mentioned above.

Name	Absolute value

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
d = ABS (s.S)	Word	4	●	●	●	●	↕
	Double word	5					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
d : Word	0.54	—	—	
d : Double word	0.72	—	—	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to store absolute value									✓	✓	✓	✓		✓	✓	✓	
s.S	I/O to take absolute value								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

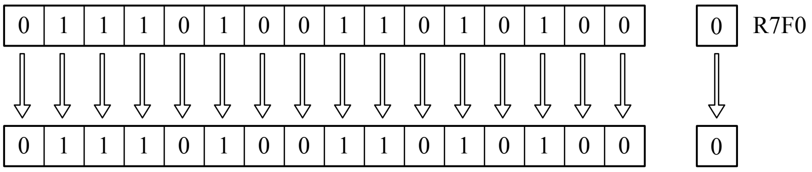
Function

- Treating s.S as a value with a sign, an absolute value is set into ‘d’.
- When s.S is positive or 0, : a content of ‘s’ is stored in ‘d’. C(R7F0) is set to ‘0’.
- When s.S is negative, : two complement of a content of ‘s’ is stored in ‘d’. C(R7F0) is set to ‘1’.
- The combination of ‘d’ and ‘s.S’ are as follows.

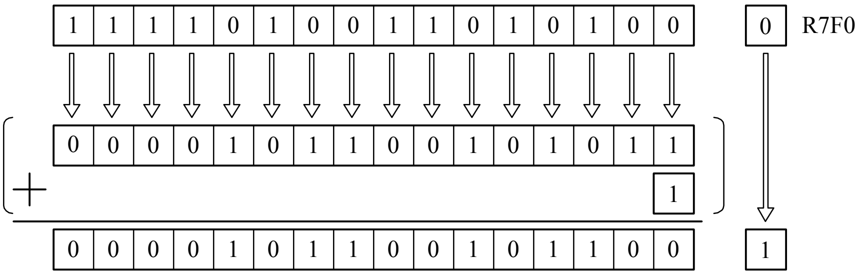
d	s.S
Word	Word
Double word	Double word

- at Word: ‘From −32,768 to −1’ are adapted to ‘from H8000 to HFFFF’.
  ‘From 0 to 32,767’ are adapted to ‘from H0000 to H7FFF’.
- at Double word: ‘From −2,147,483,648 to −1’ are adapted to ‘from H80000000 to HFFFFFFFF’.
  ‘From 0 to 2,147,483,647’ are adapted to ‘from H00000000 to H7FFFFFFF’.

When a value of ‘s’ is positive or 0,



When a value of ‘s’ is negative,



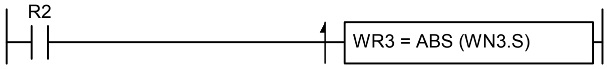
Parameter

d: An internal output to store result of calculation is specified.
s.S: An argument is specified.

Cautionary notes

A setting of a startup condition of this command should be an edge trigger.

Program example



[Program description]

An absolute value of WN3.S is set into WR3 at the rising edge of R2.

When WN3.S is ‘−12345’, if R2 is turned on, ‘12345’ is stored in WR3.

PRN ➔ PRJ

This command is equivalent to ABS (d, s) in the program (PRN file) of EH-CPU.
How to convert the program whish has used ABS (d, s) for EHV is as follows.

ABS (d, s) ➔ d = ABS (s.S)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Addition of Sing

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.S = SGET (s)	Word	4	●	●	●	●	●
	Double word	5					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
d : Word	0.50	—	—	
d : Double word	0.68	—	—	

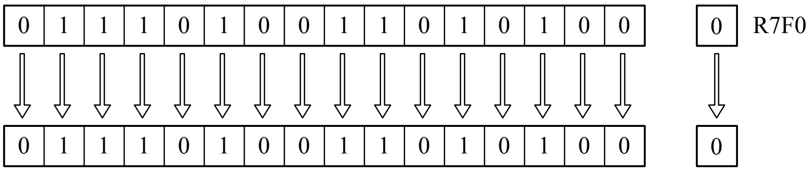
Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.S	Storage I/O after sign addition								✓	✓	✓	✓		✓	✓	✓	
s	I/O to add sign							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- When C(R7F0) is ‘0’: a content of ‘s’ is stored in d.S.
- When C(R7F0) is ‘1’: two’s complement of a content of ‘s’ is stored in d.S.
- C(R7F0) remains unchanged.
- The combination of ‘d’ and ‘s’ are as follows.

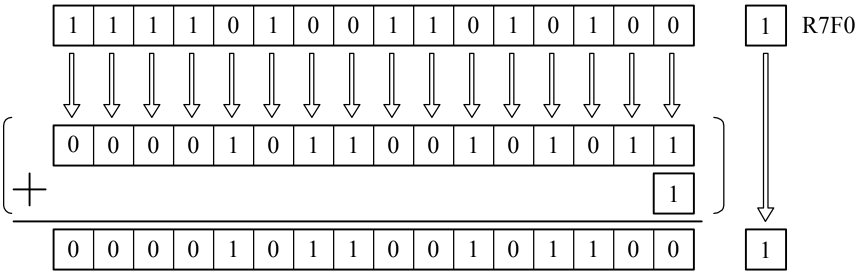
d	s
Word	Word
Double word	Double word

When C(R7F0) is 0,


0 1 1 1 0 1 0 0 1 1 0 1 0 1 0 0 0 R7F0

0 1 1 1 0 1 0 0 1 1 0 1 0 1 0 0 0

When C(R7F0) is 1,


1 1 1 1 0 1 0 0 1 1 0 1 0 1 0 0 1 R7F0

0 0 0 0 1 0 1 1 0 0 1 0 1 0 1 1

$+$ 1

0 0 0 0 1 0 1 1 0 0 1 0 1 1 0 0 1

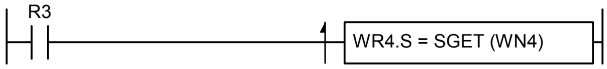
Parameter

d.S: An internal output to store result of calculation is specified.
s: An argument is specified.

Cautionary notes

A setting of a startup condition of this command should be an edge trigger.

Program example



[Program description]

The result which added a sign to a value of WN4 is set into WR4.S at the rising edge of R3.

When WN4 is ‘12345’ and C(R7F0) is ‘0’, if R3 is turned on, ‘12345’ is stored in WR4.S.

When WN4 is 1’2345’ and C(R7F0) is ‘1’, if R3 is turned on, ‘-12345’ is stored in WR4.S.

PRN ➔ PRJ

This command is equivalent to SGET (d, s) in the program (PRN file) of EH-CPU.
How to convert the program which has used SGET (d, s) for EHV is as follows.

SGET (d, s) ➔ d.S = SGET (s)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Bit extension

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
d.S = EXT (s.S, n)	Word	5	●	●	●	●	●
	Double word	5					

Command processing time (μs)				Remarks
Average		Maximum		n is 1 to16.
Condition	Time	Condition	Time	
d : Word	0.72	—	—	
d : Double word	0.72	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.S	Storage I/O after bit extension								✓	✓	✓	✓		✓	✓	✓	
s.S	I/O before bit extension							✓	✓	✓	✓	✓					✓
n	Location of sign bit																✓

Function

- A signed bit (the n-1th bit) of s.S is extended to d.S.

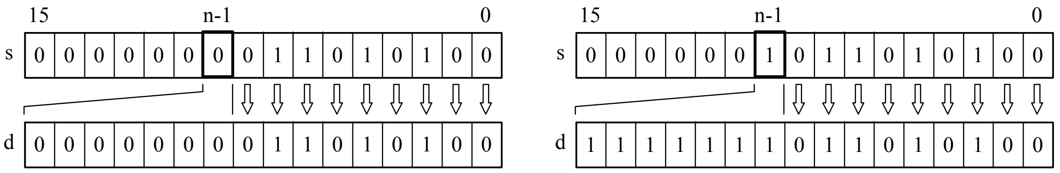
  When d.S is Word: a value of the n−1th bit is stored in from the n-1th bit to MSB of d.S.

  When d.S is Double word: a value fo the n−1th bit is stored in from the n-1th bit of d.S to MSB of upper word.

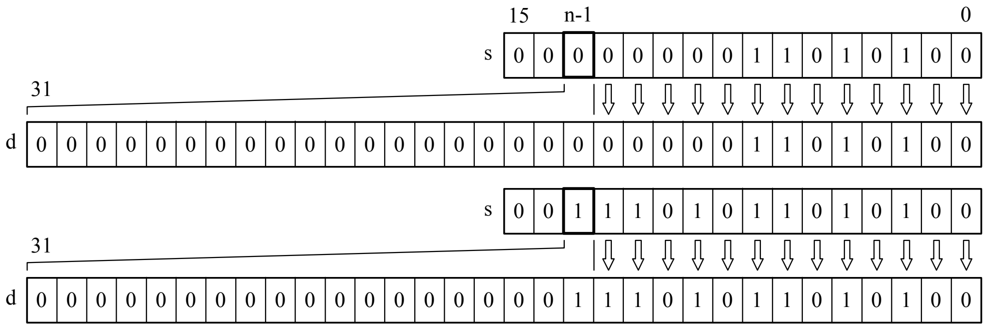
- The combination of 'd.S' and 's.S' are as follows.

d.S	s.S
Word	Word
Double word	

When s.S: Word, and d.S: Word,


15
n-1
0
s 0 0 0 0 0 0 0 0 1 1 0 1 0 1 0 0
d 0 0 0 0 0 0 0 0 1 1 0 1 0 1 0 0
15
n-1
0
s 0 0 0 0 0 0 1 0 1 1 0 1 0 1 0 0
d 1 1 1 1 1 1 1 0 1 1 0 1 0 1 0 0

when s.S: Word, and d.S: Double word,


15
n-1
0
s 0 0 0 0 0 0 0 0 1 1 0 1 0 1 0 0
31
d 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 0 1 0 0
s 0 0 1 1 1 0 1 0 1 1 0 1 0 1 0 0
31
d 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 1 1 0 1 0 1 0 0

Parameter

d.S: An internal output to store result of calculation is specified.

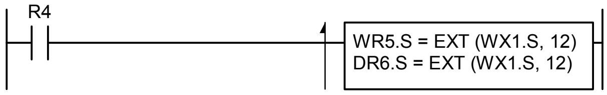
s.S: An argument is specified.

n: Bit location of signed bit is specified.

Cautionary notes

The number of bits to extend is specified to ‘n’. When extending 12-bits analog data, ‘12’ should be specified, and when extending 14-bits analog data, ‘14’ is should be specified.

Program example



```
WR5.S = EXT (WX1.S, 12)
DR6.S = EXT (WX1.S, 12)
```

[Program description]

The result which extended a singed bit (the 12th bit, b11) of a value of WX1.S to upper bit is stored in WR5.S or DR6.S at the rising edge of R4.

When WX1.S is ‘H7FF’, if R4 is turned on, ‘H07FF’ is stored in WR5.S.

When WX1.S is ‘H800’, if R4 is turned on, ’HFFFFF800’ is stored in DR6.S.

PRN ➔ PRJ

This command is equivalent to EXT (d, s) in the program (PRN file) of EH-CPU.
How to convert the program which has used EXT (d, s) for EHV is as follows.

EXT (d, s) ➔ d.S = EXT (s.S, 16), provided that ‘d.S’ is Double word,

* if converted by a conversion tool, it is converted as mentioned above.

Name	Two's complement

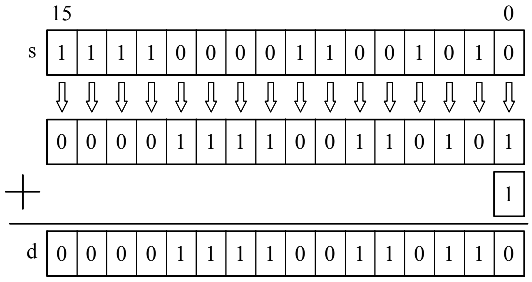
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
d = NEG (s)	Word	4	●	●	●	●	●
	Double word	5					

Command processing time (μs)				Remarks
Average		Maximum		n is 1 to15.
Condition	Time	Condition	Time	
d : Word	0.40	—	—	
d : Double wor	0.54	—	—	

Usable I/O		Bit						Word					Double word				
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	Constant
d	I/O after taking two's complement								✓	✓	✓	✓		✓	✓	✓	
s	I/O to take two's complement							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Two's complement of 'd' is calculated. ('1' is added after reversing a content of 'd'. C(R7F0) remains unchanged.)



15 0

s 1 1 1 1 0 0 0 0 1 1 0 0 1 0 1 0

0 0 0 0 1 1 1 1 0 0 1 1 0 1 0 1

+ 1

d 0 0 0 0 1 1 1 1 0 0 1 1 0 1 1 0

Parameter

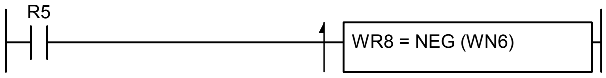
d: An internal output to store result of calculation is specified.

s: An argument is specified.

Cautionary notes

- A setting of a startup condition of this command should be an edge trigger.
- When making I/O to calculate two's complement and I/O to substitute the result the same, 'd' and 's' should be set to the same I/O.

Program example



[Program description]

Two's complement of a value of WN6 calculated is substituted for WR8 at the rising edge of R5.

When WN6 is 'H1234', if R5 is turned on, 'HEDCC' is stored in WR8.

PRN ➔ PRJ

This command is equivalent to NEG (d) in the program (PRN file) of EH-CPU.
How to convert the program which has used NEG (d) for EHV is as follows.

NEG (d) ➔ d = NEG (d)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Square root

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = SQR (s) d.FL = SQR (s.FL)	Integer	5	↕	●	●	●	●
	Floating decimal point	5					

Command processing time (μs)				Remarks
Average		Maximum		n is 1 to 15.
Condition	Time	Condition	Time	
Integer	8.06	—	—	
Floating decimal point	419	—	—	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Result of calculation									✓	✓	✓						
d.FL	Result of calculation														✓	✓	✓	
s	Object of calculation													✓	✓	✓	✓	✓
s.FL	Object of calculation													✓	✓	✓	✓	✓

Function

- If ‘s’ is specified by Double word without an extension, a square root which treats a 32-bits unsigned binary value as an argument is calculated. (Figures below a decimal point are omitted.)
- If ‘s’ is specified by Double word of floating decimal point (with an extension ‘.FL’), a square root which treats floating decimal point as an argument is calculated.
- A form of substitution statement is written and the operation result is stored in ‘d’ or ‘d.FL’.
  e.g.) WR0 = SQR (DR10) A square root of DR10 treating as an argument calculated is stored in WR0.
- If the operation is performed normally, DER is ‘0’ (DER=0).

Parameter

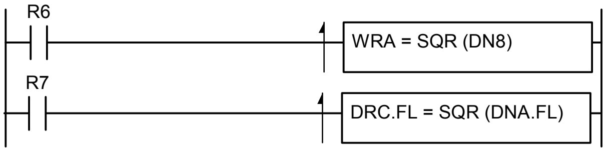
d: An internal output to store result of calculation is specified.

s: An argument is specified.

Cautionary notes

- Internal outputs to store object of calculation and result of calculation should be specified within I/O range.
- If object of calculation is specified to floating decimal point, result of calculation should be specified to floating decimal point also. But if object of calculation is specified to integer, result of calculation is also stored with integer.
- When the operation is performed in floating decimal point, object of result of calculation should be converted from integer to real number before execution of the operation.
- When the operation is performed in floating decimal point, if the operation results is in outside the range from −1e+37 to 1e+37, DER is ‘1’ (DER＝1).
- A format of floating decimal point conforms to IEEE754.

Program example



[Program description]

- A square root of DN8 calculated is set into WRA at the rising edge of R6.
  (DN8 is treated as integer. Result of calculation is also integer.)
- A square root of DNA.FL calculated is set into DRC.FL at the rising edge of R7.
  (DRC.FL is treated as floating decimal point. Result of calculation are also floating decimal point.)

PRN ➔ PRJ

This command is equivalent to FUN 60(s) [square root of integer] / FUN 116 (s) [square root of floating decimal point] in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 60 (s) / FUN 116 (s) for EHV is as follows.

(1) FUN 60 (s) ➔ s+2 = SQR (s), provided that ‘s’ is Double word.

e.g.) FUN 60 (WR100) ➔ WR102 = SQR (DR100)

(2) FUN 116 (s) ➔ [s+2].FL = SQR (s.FL), provided that ‘s’ and ‘s+2’ are Double word.

e.g.) FUN 116 (WR100) ➔ DR102.FL = SQR (DR100.FL)

* If converted by a conversion tool, it is converted as mentioned above.

Name	BCD Square root

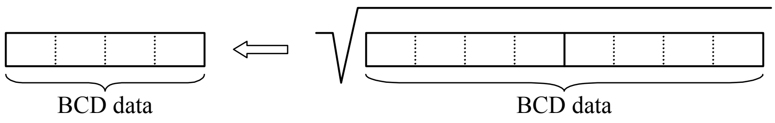
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = BSQR (s)	—	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		n is 1 to 15.
Condition	Time	Condition	Time	
—	12.40	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Result of calculation (BCD)								✓	✓	✓						
S	Object of calculation (BCD)												✓	✓	✓	✓	✓

Function

- A square root of a content of ‘s’ calculated is output to ‘d’.
- BCD data should be set into ‘s’.
- Figures below a decimal point are omitted.



Parameter

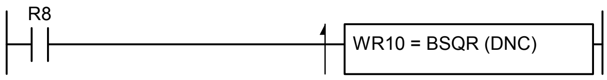
d: An internal output to store result of calculation is specified.

s: An argument is specified.

Cautionary notes

When ‘s’ is BCD data error (including values from HA to HF), the operation is not performed because DER(R7F4) is’1’.

Program example



[Program description]

A calculated square root of DNC is set into WR10 in BCD data at the rising edge of R8.

PRN ➔ PRJ

This command is equivalent to SQR (d, s) in the program (PRN file) of EH-CPU.

How to convert the program whish has used SQR (d, s) for EHV is as follows.

SQR (d, s) ➔ d = BSQR (s), provided that ‘s’ is Double word.

* If converted by a conversion tool, it is converted as mentioned above.

Name	Exponentiation

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = POW (s, n) d.FL = POW (s.FL, n.FL)	Integer	5	↕	●	●	●	●
	Floating decimal point	7					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Integer	1.10 + 7.30 × n	—	—	
Floating decimal point	460	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Result of calculation													✓	✓	✓	
d.FL	Result of calculation													✓	✓	✓	
s	Object of calculation (Base)							✓	✓	✓	✓	✓					✓
s.FL	Object of calculation (Base)												✓	✓	✓	✓	✓
n	Object of calculation (Exponent)							✓	✓	✓	✓	✓					✓
n.FL	Object of calculation (Exponent)												✓	✓	✓	✓	✓

Function

- Treating an unsigned binary value specified by ‘s’ and an exponent (binary value) specified by ‘n’ as argument, exponentiation is calculated.
- Treating a real number value specified by ‘s.FL’ and an exponent (binary value) specified by ‘n.FL’, exponentiation is calculated.
- A form of substitution statement is written and the operation result is stored in ‘d’ or ‘d.FL’.

  e.g.1) DR10 = POW (WR1, WR2) The result which calculated WR1^WR2 is stored in DR10.

  e.g.2) DR10.FL = FPOW (DR0.FL, DR2.FL) The result which calculated DR0.FL^DR2.FL is stored in DR10.FL.
- The operation is performed normally, DER is ‘0’ (DER＝0).

Parameter

d / d.FL: An internal output to store result of calculation is specified.

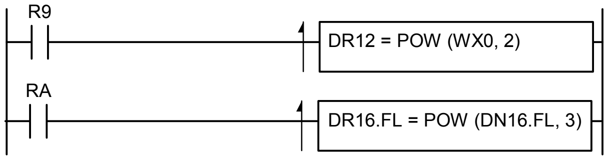
s / s.FL: An internal output stored object of calculation (base) is specified.

n / n.FL: An internal output stored object of calculation (exponent) is specified.

Cautionary notes

- Internal outputs for argument and result of calculation should be specified within I/O number.
- In case of floating decimal point operation, an extension ".FL" is necessary for internal outputs for storing object of calculation and result of calculation.
- In case of floating decimal point operation, object of calculation should be converted from integer to real number before execution of the operation. (Otherwise, you cannot get correct result of calculation.)
- In case of unsigned binary value, if the operation result is in outside the range from 0 to 4,294,967,295, DER is '1' (DER＝1).
- In case of floating decimal point, if the operation result is in outside the range from −1e+37 to 1e+37, DER is '1' (DER＝).
- A format of floating decimal point conforms to IEEE754.
- In operation by floating decimal point, if conditions of 's.FL' and 'n.FL' are as follows, getting correct operation results is impossible.
  - s.FL = 0 and n.FL < 0
  - s.FL < 0 and n.FL < | 2147483647.0 | and figures below a decimal point are in n.Fl.
  - s.FL < 0 and FL < | 2147483647.0 | and n.FL is an even number.
  - s.FL ≥0 and s.FL ≠ 1.0 and n.FL ≤ 7.097827128933839700E+02 / s.FL and n.FL < −7.097827E+02 / s.FL
  - s.FL ≥ 0 and s.FL ≠ 1.0 and n.FL > 7.097827128933839700E+02 / s.FL

Program example



[Program description]

- The value which squared a value of WX0 is substituted for DR12 at the rising edge of R9.
  (WX0 is treated as integer. The result of calculation is also integer.)
- The value which cubed a value of DN16.FL is substituted for DR16.FL at the rising edge of RA.
  (DN16.FL is treated as floating decimal point. The result of calculation is also floating decimal point.)

Name	Sine function (Degree)

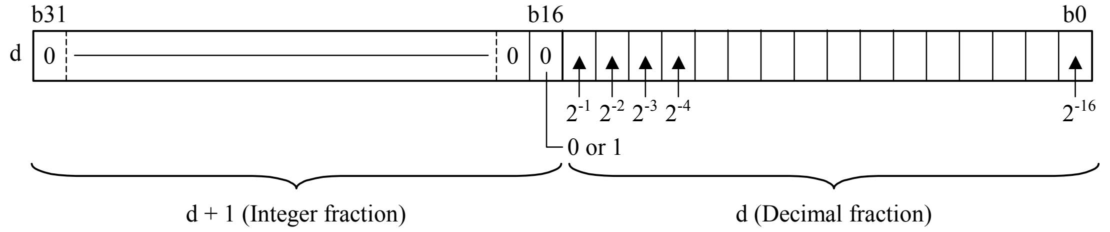
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = SIN (s)	–	4	↕	●	●	●	●

Command processing time (µs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	0.94	–	–	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Result of calculation														✓	✓	✓	
s	Argument								✓	✓	✓	✓	✓					✓

Function

- Sine function, which treats an unsigned binary value specified by 's' as an argument, is calculated.
- A form of substitution statement is written and the operation result is stored in 'd' (Double word: upper word is stored in the integer fraction and lower word is stored in the decimal fraction).



e.g.) DR100 = SIN (WR0) Sine function treating WR0 as an argument calculated is stored in DR100.

- The operation result is represented by binary value. And negative number is represented by two's complement. (The operation is performed normally, DER is '0' (DER=0).)
- A decimal fraction data ('d' parameter lower word) is the value which is a real value times 65,535.

Parameter

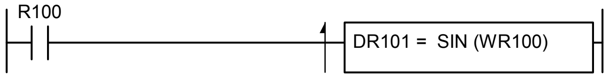
d: An internal output to store result of calculation is specified.

s: An argument (angle [Degree]) is specified. The range is 0≦s≦360.
(If a value of 's' parameter is outside the range, the operation is not performed because of DER=1.)

Cautionary notes

- Internal output for argument and internal output to store result of calculation should be specified within I/O number.
- Although a decimal point fraction is contained in result of calculation, it differs from a floating decimal point format of IEEE754.

Program example



[Program description]

Sine function of WR100 is calculated, at the rising edge of R100, which result is substituted for DR101 (WR101: decimal fraction, WR102: integer fraction).

PRN ➔ PRJ

This command is equivalent to FUN 10 (s) in the program of EH-CPU (PRN file).

How to convert the program which has used FUN 10 (s) for EHV is as follows.

FUN 10 (s) ➔ s+1 = SIN (s), provided that ‘s+1’ is Double word.

e.g.) FUN 10 (WR100) ➔ DR101 = SIN (WR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Sine function (Radian)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = SINR (s.FL)	–	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	398	–	–	

Usable I/O		Bit							Word						Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.FL	Result of calculation														✓	✓	✓	
s.FL	Argument													✓	✓	✓	✓	✓

Function

- Sine function, which treats a floating decimal point (Radian) specified by ‘s.FL’ as an argument, is calculated.
- A forma of substitution statement is written and the result is stored in ‘d.FL’.

  e.g.) DR100.FL = SINR (DR0.FL)

  Sine function, which treats DR0.FL as an argument, is calculated, which result is stored in DR100.FL
- The operation result is represented by a floating decimal point. (If the operation is performed normally, DER is ‘0’(DER=0).)

Parameter

d.FL: Internal output to store result of calculation is specified.

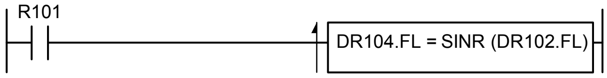
s.FL: An argument (angle [Radian]) is specified.

If the value which is s > 1.414847550405688000e+16 is specified, the operation is not performed because of DER=1.

Cautionary notes

- Internal output for argument and internal output to store result of calculation should be specified within the range of I/O numbers.
- An extension “.Fl” is necessary for internal outputs for argument and for storing result of calculation.
- If an argument is integer, the integer should be converted to a real number before execution of the operation. (Otherwise, you can not get correct result of calculation.)
- If operation result is in outside the range form −1e+37 to 1e+37, DER is ‘1’ (DER=1).
- If a value which is s.FL > 2.981568260000000000e+08, the accuracy goes down although the operation is performed. (Although the operation result comes out, DER is ‘1’ (DER=1).)
- A format of a floating decimal point conforms to IEEE754.

Program example



[Program description]

Sine function of DR102.FL is calculated at the rising edge of R101, which result is substituted for DR104.FL.

PRN ➔ PRJ

This command is equivalent to FUN 110 (s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 110 (s) for EHV is as follows.

FUN 110 (s) ➔ [s+2].FL = SINR (s.FL), provided that ‘s’ and ‘s+2’ are Double word.

e.g.) FUN 110 (WR100) ➔ DR102.FL = SINR (DR100.FL)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Cosine function (Degree)

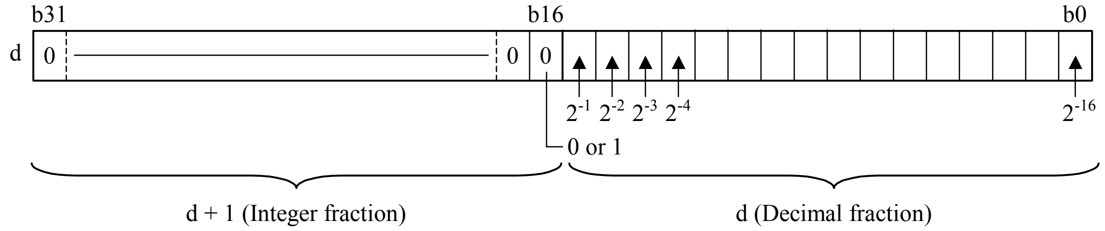
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = COS (s)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	1.02	–	–	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Result of calculation														✓	✓	✓	
s	Argument								✓	✓	✓	✓	✓					✓

Function

- Cosine function, which treats an unsigned binary value (Degree) specified by ‘s’ as an argument, is calculated.
- A form of substitution statement is written and the operation result is stored in ‘d’ (Double word: upper word is stored in an integer fraction and lower is stored in a decimal fraction).



e.g.) DR100 = COS (WR0) Cosine function which WR0 is treated as an argument, which result is stored in DR100.

- The operation result is represented by binary value. And negative number if represented by two’s complement. (If the operation is performed normally, DER is ‘0’ (DER=0).)
- A decimal fraction data (‘d’ parameter lower word) is the value which is a real value times 65,535.

Parameter

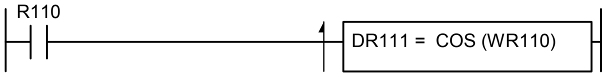
d: Internal output to store result of calculation is specified.

s: An argument (angle [Degree]) is specified. The range is 0≦s≦360.
(If a value of ‘s’ parameter is in outside the range, the operation is not performed because of DER=1.)

Cautionary notes

- Internal outputs for argument and to store result of calculation should be specified within the range of I/O numbers.
- Although a decimal point fraction is contained in a result of calculation, it differs from a floating decimal point format of IEEE754.

Program example



[Program description]

Cosine function of WR110 is calculated at the rising edge of R110, which result is substituted for DR111 (WR111: a decimal fraction, Wr112: an integer fraction).

PRN ➔ PRJ

This command is equivalent to FUN 11 (s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 11 (s) for EHV is as follows.

FUN 11 (s) ➔ s+1 = COS (s), provided that ‘s+1’ is Double word.

e.g.) FUN 11 (WR100) ➔ DR101 = COS (WR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Cosine function (Radian)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = COSR (s.FL)	—	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	422	—	—	

Usable I/O		Bit							Word						Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.FL	Result of calculation														✓	✓	✓	
s.FL	Argument													✓	✓	✓	✓	✓

Function

- Cosine function, which treats a floating decimal point (radian) specified by ‘s.FL’ as an argument, is calculated.
- A form of substitution statement is written and the operation result is stored in ‘d.FL’.
  e.g.) DR100.FL = COSR (DR0.FL)
  Cosine function which treats DR0.FL as an argument is calculated, which result is stored in DR100.FL.
- The operation result is represented by a floating decimal point. (The operation is performed normally, DER is ‘0’ (DER＝0).)

Parameter

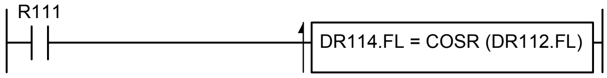
d.FL: Internal output to store result of calculation is specified.

s.FL: An argument (angle [Radian]) is specified.
If the value which is s > 1.414847550405688000e+16 is specified, the operation is not performed because of DER=1.

Cautionary notes

- Internal outputs for argument and to store result of calculation should be specified within the range of I/O numbers.
- An extension “.FL” is necessary for Internal outputs for argument and for storing result of calculation.
- If an argument is an integer, the integer should be converted to a real number before execution of the operation. (Otherwise, you cannot get correct result of calculation.)
- If the operation result is in outside the range from −1e+37 to 1e+37, DER is ‘1’ (DER=1).
- If the value which is s.FL > 2.981568260000000000e+08 is specified, the accuracy goes down although the operation is performed.
  (Although the operation result comes out, DER is ‘1’ (DER=1).)
- A format of a floating decimal point conforms to IEEE754.

Program example



[Program description]

Cosine function of DR112.FL is calculated at the rising edge of R111, which result is substituted for DR114.FL.

PRN ➔ PRJ

This command is equivalent to FUN 111 (s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 111 (s) for EHV is as follows.

FUN 111 (s) ➔ [s+2].FL = COSR (s.FL), provided that ‘s’ and ‘s+2’ are Double word.

e.g.) FUN 111 (WR100) ➔ DR102.FL = COSR (DR100.FL)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Tangent function (Degree)

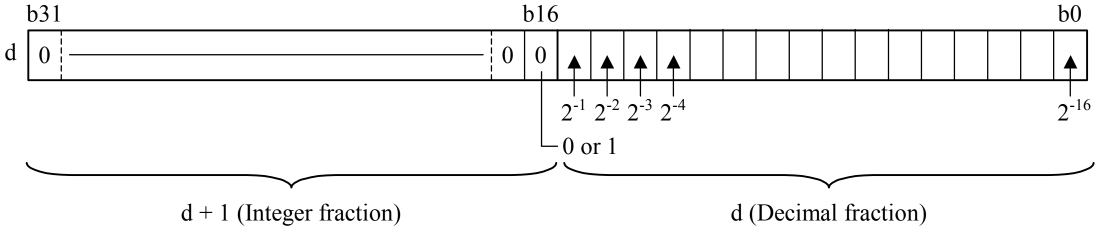
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = TAN (s)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	1.08	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Result of calculation													✓	✓	✓	
s	Argument							✓	✓	✓	✓	✓					✓

Function

- Tangent function, which treats an unsigned binary value (Degree) specified by 's' as an argument, is calculated.
- A form of substitution statement is written and the operation result is stored in 'd' (Double word: upper word is stored in a integer fraction and lower is stored in a decimal fraction)



e.g.) DR100 = TAN (WR0) Tangent function which treats WR0 as an argument is calculated, which result is stored in DR100.

- The operation result is represented by binary value. And negative number is represented by two's complement. (If the operation is performed normally, DER is '0' (DER=0).)
- A decimal fraction data ('d' parameter lower word) is the value which is a real value times 65,535.

Parameter

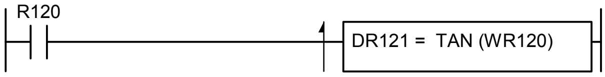
d: Internal output to store result of calculation is specified.

s: An argument (angle [Degree]) is specified. The range is 0≦s≦360 (except 90 and 270). (A value of 's' parameter is in outside the range, the operation is not performed because DER=1. And if s=90, and 270,'H7FFFFFFF' is stored in the output to store result of calculation because of DER=1.)

Cautionary notes

- Internal outputs for argument and to store result of calculation should be specified within the range of I/O numbers.
- Although a decimal points fraction is contained in result of calculation, it differs from a floating decimal point format of IEEE754.

Program example



[Program description]

Tangent function of WR120 is calculated at the rising edge of R120, which result is substituted for DR121 (WR121: a decimal fraction, WR122: an integer fraction).

PRN ➔ PRJ

This command is equivalent to FUN 12 (s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 12 (s) for EHV is as follows.

FUN 12 (s) ➔ s+1 = TAN (s), provided that ‘s+1’ is Double word.

e.g.) FUN 12 (WR100) ➔ DR101 = TAN (WR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Tangent function (Radian)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = TANR (s.FL)	—	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	515	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.FL	Result of calculation													✓	✓	✓	
s.FL	Argument												✓	✓	✓	✓	✓

Function

- Tangent function, which treats a floating point (Radian) specified by ‘s.FL’ as an argument, is calculated.
- A form of substitution statement is written and the operation result is stored in ‘d.FL’.
  e.g.) DR100.FL = TANR (DR0.FL)
  Tangent function which treats DR0.FL as an argument is calculated, which result is stored in ‘DR100.FL’.
- The operation result is represented by a floating decimal point. (If the operation is performed normally, DER is ‘0’ (DER = 0).)

Parameter

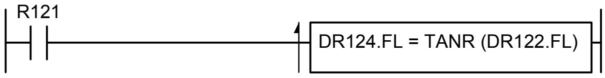
d.FL: Internal output to store result of calculation is specified.

s.FL: An argument (angle [Radian]) is specified.
If a value which is s > 1.414847550405688000e+16 is specified, the operation is not performed because of DER=1.

Cautionary notes

- Internal outputs for argument and to store result of calculation should be specified within the range of I/O numbers.
- An extension “.FL” is necessary for internal outputs for argument and for storing result of calculation.
- If an argument is an integer, the integer should be converted to a real number before execution of the operation. (Otherwise, you cannot get correct results of calculation.)
- If the operation result is in outside the range from −1e+37 to 1e+37, DER is ‘1’ (DER=1).
- If a value which is s.FL > 2.981568260000000000e+08 / 2 is specified, the accuracy goes down although the operation is performed.
  (Although the operation result comes out, DER is ‘1’ (DER=1).)
- A format of a floating decimal point conforms to IEEE754.

Program example



[Program description]

Tangent function of DR122.FL is calculated at the rising edge of R121, which result is substituted for DR124.FL.

PRN ➔ PRJ

This command is equivalent to FUN 112 (s) in the program (PRN file) of EHV-CPU.

How to convert the program which has used FUN 112 (s) for EHV is as follows.

FUN 112 (s) ➔ [s+2].FL = TANR (s.FL), provided that 's' and 's+2" are Double word.

e.g.) FUN 112 (WR100) ➔ DR102.FL = TANR (DR100.FL)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Arc sine function (System of Degree units)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = ASIN (s)	–	5	↕	●	●	●	●

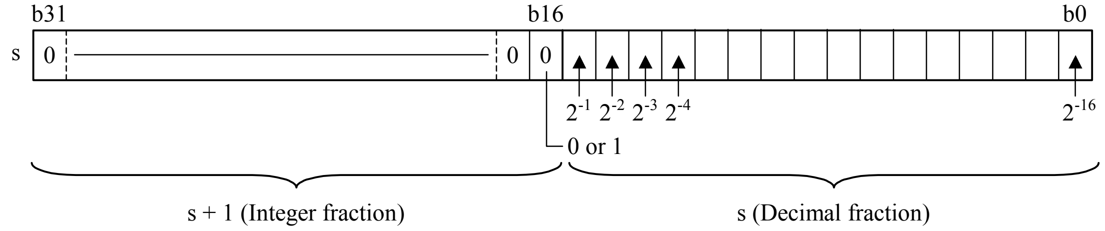
Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	5.46	–	–	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Result of calculation									✓	✓	✓	✓					
s	Argument													✓	✓	✓	✓	

Function

- Arc sine function (SIN^{-1}), which treats an unsigned binary value specified by ‘s’ (Double word: upper word is an integer fraction and lower is a decimal fraction) as an argument, is calculated.

  A decimal fraction data (‘s’ parameter lower word) is the value which is a real value times 65,535.



- A form of substitution statement is written and the operation result is stored in ‘d’.

  e.g.) WR200 = ASIN (DR2) SIN^{-1} which treats DR2 as an argument is calculated, which result is stored in WR200.
- If the operation is performed normally, DER is ‘0’ (DER=0).
- The result of calculation is a binary value and an angle (Degree) from 0 to 90 and from 180 to 270.

Parameter

d: Internal output to store result of calculation is specified.

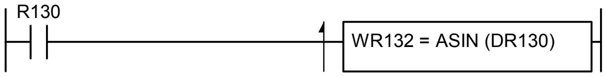
s: An argument is specified. A decimal point data (s parameter lower word) should be the value which a real value times 65,535.

If |s| > 1, the operation is not performed because of DER=1.

Cautionary notes

- Internal outputs for argument and to store result of calculation should be specified within the range of I/O numbers.
- Although a decimal point fraction is also contained in the argument, it differs from a floating decimal point format of IEEE754.

Program example



[Program description]

SIN^{-1} of DR130 (WR130: a decimal fraction, WR131: an integer fraction) is calculated at the rising edge of R130, which result is substituted for WR132.

PRN ➔ PRJ

This command is equivalent to FUN 13 (s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 13 (s) for EHV is as follows.

FUN 13 (s) ➔ s+2 = ASIN (s), provided that 's' is Double word.

e.g.) FUN 13 (WR100) ➔ WR102 = ASIN (DR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Arc sine function (System of Radian units)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = ASINR (s.FL)	–	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	202	–	–	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.FL	Result of calculation													✓	✓	✓	
s.FL	Argument												✓	✓	✓	✓	✓

Function

- Arc sine function (SIN^{-1}), which treats a floating decimal point specified by ‘s.FL’ as an argument, is calculated.
- A form of substitution statement is written and the operation result is stored in ‘d.FL’ with Radian unit.

  e.g) DR200.FL = ASINR (DR2.FL)

  SIN^{-1} which treats DR2.FL as an argument is calculated, which result is stored in DR200.FL.
- The operation result is an angle system of Radian units, and represented by a floating decimal point.
- If the operation is performed normally, DER is ‘0’ (DER=0).

Parameter

d.FL: Internal output to store result of calculation is specified.

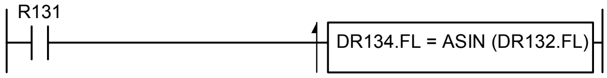
s.FL: An argument which a real number to calculate SIN^{-1} is stored in is specified.

If the value which is s.FL > 1 is specified, the operation is not performed because of DER=1.

Cautionary notes

- Internal outputs for argument and to store result of calculation should be specified within the range of I/O numbers.
- An extension “.FL” is necessary for internal outputs for argument and for storing result of calculation.
- If an argument is an integer, the integer should be converted to a real number before execution of the operation. (Otherwise, you cannot get correct results of calculation.)
- If the operation result is in outside the range from −1e+37 to 1e+37, DER is ‘1’ (DER=1).
- A format of a floating decimal point conforms to IEEE754.

Program example



[Program description]

SIN^{-1} of DR132.FL is calculated at the rising edge of R131, which result is substituted for DR134.FL.

PRN ➔ PRJ

This command is equivalent to FUN 113 (s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 113 (s) for EHV is as follows.

FUN 113 (s) ➔ [s+2].FL = ASINR (s.FL), provided that 's' and 's+2' are Double word.

e.g.) FUN 113 (WR100) ➔ DR102.FL = ASINR (DR100.FL)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Arc cosine function (System of Degree units)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = ACOS (s)	—	5	↕	●	●	●	●

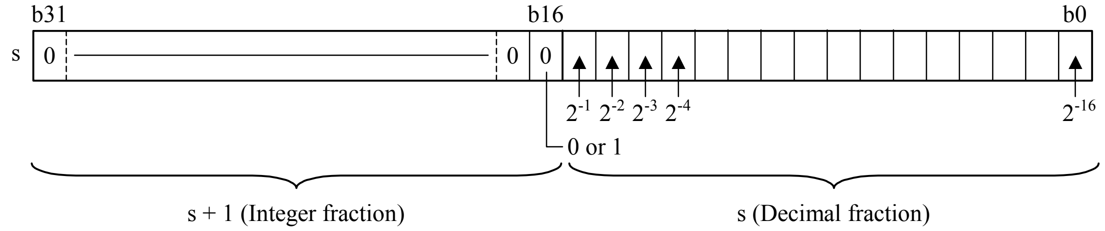
Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	5.50	—	—	

Usable I/O		Bit							Word				Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Result of calculation									✓	✓	✓	✓					
s	Argument													✓	✓	✓	✓	

Function

- Arc cosine function (COS^{-1}), which treats an unsigned binary value specified by ‘s’ (Double word: upper word is integer fraction, lower is decimal fraction) as an argument, is calculated.

  A decimal fraction data (s parameter lower word) is the value which a real value time 65,535.



- A form of substitution statement is written and the operation result is stored in ‘d’.

  e.g.) WR200 = ACOS (DR2) COS^{-1} function which treats DR2 as an argument is calculated, which result is stored in WR200.
- If the operation is performed, DER is ‘0’ (DER=0).
- The result of calculation is a binary value and an angle (Degree) which is from 0 to 180.

Parameter

d: Internal output to store result of calculation is specified.

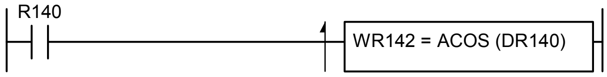
s: An argument is specified. A decimal point data (s parameter lower word) is the value which a real value times 65,535.

If |s| > 1, the operation is not performed because of DER=1.

Cautionary notes

- Internal outputs for argument and to store results of calculation should be specified within the range of I/O numbers.
- Although a decimal point fraction is contained in an argument, it differs from a floating decimal point format of IEEE754.

Program example



[Program description]

COS^{-1} of DR140 (WR140: a decimal fraction, WR141: an integer fraction) is calculated at the rising edge of R140, which result is substituted for WR142.

PRN ➔ PRJ

This command is equivalent to FUN 14 (s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 14 (s) for EHV is as follows.

FUN 14 (s) ➔ s+2 = ACOS (s), provided that 's' is Double word.

e.g.) FUN 14 (WR100) ➔ WR102 = ACOS (DR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Arc cosine function (System of Radian units)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = ACOSR (s.FL)	–	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	208	–	–	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.FL	Result of calculation														✓	✓	✓	
s.FL	Argument													✓	✓	✓	✓	✓

Function

- Arc cosine function (COS^{-1}), which treats a floating decimal point specified by 's.FL' as an argument, is calculated.
- A form of substitution statement is written and the operation result is stored in 'd.FL' with Radian units.

  e.g.) DR200.FL = ACOSR (DR2.FL)

  COS^{-1} function which treats DR2.FL as an argument is calculated, which result is stored in DR200.FL.
- The operation result is an angle of system of Radian units and represented by a floating decimal point.
- If the operation is performed normally, DER is '0' (DER=0).

Parameter

d.FL: Internal output to store results of calculation is specified.

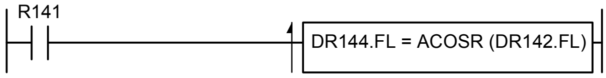
s.FL: An argument which a real number to calculate COS^{-1} is stored in is specified.

If a value which is s.FL > 1 is specified, the operation is not performed because of DER=1.

Cautionary notes

- Internal outputs for argument and to store results of calculation should be specified within the range of I/O numbers.
- An extension ".FL" is necessary for internal outputs for argument and for storing results of calculation.
- If an argument is an integer, the integer should be converted to a real number before execution of the operation. (Otherwise, you cannot get correct results of calculation.)
- If the operation result is in outside the range from −1e+37 to 1e+37, DER is '1' (DER=1).
- A format of a floating decimal point conforms to IEEE754.

Program example



[Program description]

COS^{-1} of DR142.FL is calculated at the rising edge of R141, which result is substituted for DR144.FL.

PRN ➔ PRJ

This command is equivalent to FUN 114 (s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 114 (s) for EHV is as follows.

FUN 114 (s) ➔ [s+2].FL = ACOSR (s.FL), provided that ‘s’ and ‘s+2’ are Double word.

e.g) FUN 114 (WR100) ➔ DR102.FL = ACOSR (DR100.FL)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Arc tangent function (System of Degree units)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = ATAN (s)	—	5	↕	●	●	●	●

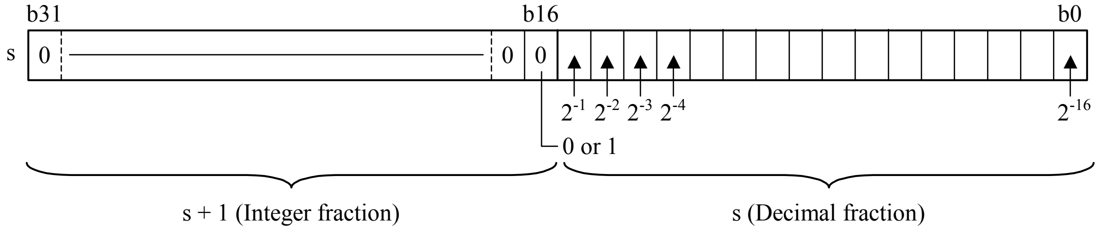
Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	6.22	—	—	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Result of calculation									✓	✓	✓	✓					
s	Argument													✓	✓	✓	✓	

Function

- TAN^{-1} function, which treats an unsigned binary value specified by 's' (Double word: upper word is an integer fraction, lower is a decimal fraction) as an argument, is calculated.

  A decimal point data (s parameter lower word) is the value which a real value times 65,535.



- A form of substitution statement is written and the operation result is stored in 'd'.

  e.g.) WR200 = ATAN (DR2) DR2 を引数とする TAN^{-1} function which treats DR2 as an argument is calculated, which result is stored in WR200.
- If the operation is performed normally, DER is '0' (DER=0).
- The result of calculation is a binary value and an angle (Degree) which is from 0 to 180.

Parameter

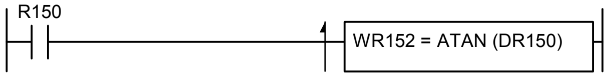
d: Internal output to store results of calculation is stored.

s: An argument is specified. A decimal point data (s parameter lower word) is the value which a real value times 65,535.

Cautionary notes

- Internal outputs for argument and to store results of calculation should be specified within the range of I/O numbers.
- Although a decimal point fraction is contained in an argument, it differs from a floating decimal point format of IEEE754.

Program example



[Program description]

TAN^{-1} of DR150 (WR150: a decimal fraction, WR151: an integer fraction) is calculated at the rising edge of R150, which result is substituted for WR152.

PRN ➔ PRJ

This command is equivalent to FUN 15 (s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 15 (s) for EHV is as follows.

FUN 15 (s) ➔ s+2 = ATAN (s), provided that ‘s’ is Double word.

e.g.) FUN 15 (WR100) ➔ WR102 = ATAN (DR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Arc tangent function (System of Radian units)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = ATANR (s.FL)	−	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
−	208	−	−	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d.FL	Result of calculation														✓	✓	✓	
s.FL	Argument													✓	✓	✓	✓	✓

Arithmetic

d.FL = ATANR (s.FL)

Function

- TAN^{-1} function, which treats a floating decimal point specified by 's.FL' as an argument, is calculated.
- A form of substitution statement is written and the operation result is stored in 'd.FL' with Radian units.

  e.g.) DR200.FL = ATANR (DR2.FL)

  TAN^{-1} function which treats DR2.FL as an argument is calculated, which result is stored in DR200.FL.
- The operation result is an angle of system of Radian units and represented by a floating decimal point.
- If the operation is performed normally, DER is '0' (DER=0).

Parameter

d.FL: Internal output to store results of calculation is specified.

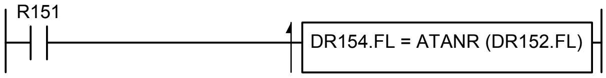
s.FL: An argument which a real number to calculate TAN^{-1} is stored in is specified.

If a value which is s.FL > 1 is specified, the operation is not performed because of DER=1.

Cautionary notes

- Internal outputs for argument and to store results of calculation should be specified within the range of I/O numbers.
- An extension ".FL" is necessary for internal outputs for argument and for storing results of calculation.
- If an argument is an integer, the integer should be converted to a real number before execution of the operation. (Otherwise, you cannot get correct results of calculation.)
- If the operation result is in outside the range from −1e+37 to 1e+37, DER is '1' (DER=1).
- A format of a floating decimal point conforms to IEEE754.

Program example



[Program description]

TAN^{-1} of DR152.FL is calculated at the rising edge of R151, which result is substituted for DR154.FL.

PRN ➔ PRJ

This command is equivalent to FUN 115 (s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 115 (s) for EHV is as follows.

FUN 115 (s) ➔ [s+2].FL = ATANR (s.FL), provided that ‘s’ and ‘s+2’ are Double word.

e.g.) FUN 115 (WR100) ➔ DR102.FL = ATANR (DR100.FL)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Exponent (Floating decimal point)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = EXP (s.FL)	–	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	395	–	–	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Result of calculation														✓	✓	✓	
s	Argument													✓	✓	✓	✓	✓

Function

- Exponent, which treats a value of the real number specified by ‘s.FL’ as an argument, is calculated.
- A form of substitution statement is written and the operation result is stored in ‘d.FL’.
  e.g.) DR10.FL = EXP (DR0.FL) Exponent of DR0.FL is calculated, which result is stored in DR10.FL.
- If the operation is performed normally, DER is ‘0’ (DER=0).

Parameter

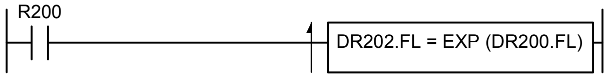
d.FL: Internal output to store results of calculation is specified.

s.FL: An argument is specified.

Cautionary notes

- Internal outputs for argument and to store results of calculation should be specified within the range of I/O numbers.
- An extension “.FL” is necessary for internal outputs for argument and for storing results of calculation.
- If an argument is an integer, the integer should be converted to a real number before execution of the operation. (Otherwise, you cannot get correct results of calculation.)
- If the operation result is in outside the range from −1e+37 to 1e+37, DER is ‘1’ (DER=1).
- If s.FL < -7.0839639e+02, DER is ‘1’ (DER=1) because the calculation is impossible.
- A format of a floating decimal point conforms to IEEE754.

Program example



[Program description]

Exponent of the real number specified by DR200.FL is calculated at the rising edge of R200, which result is substituted for DR202.FL.

PRN ➔ PRJ

This command is equivalent to FUN 117 (s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 117 (s) for EHV is as follows.

FUN 117 (s) ➔ [s+2].FL = EXP (s.FL), provided that ‘s’ and ‘s+2’ are Double word.

e.g.) FUN 117 (WR100) ➔ DR102.FL = EXP (DR100.FL)

* If converted by a conversion tool, it is converted as mentioned above.

Arithmetic
d.FL = EXP (s.FL)

Name	Natural logarithm (Floating decimal point)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = LOG (s.FL)	–	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	457	–	–	

Usable I/O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Result of calculation														✓	✓	✓	
s	Argument													✓	✓	✓	✓	✓

Function

- The logarithm with base the natural logarithm (e) by treating a real number specified by 's.FL' as an argument is calculated.
- A form of substitution statement is written and the operation results is stored in 'd.FL'.

  e.g) DR10.FL = LOG (DR0.FL) The natural logarithm of DR0.FL is calculated, which result is stored in DR10.FL.
- If the operation is performed normally, DER is '0' (DER=0).

Parameter

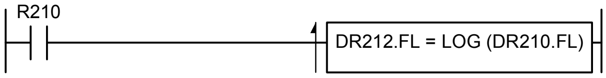
d.FL: Internal output to store results of calculation is specified.

s.FL: An argument is specified.

Cautionary notes

- Internal outputs for argument and to store results of calculation should be specified within the range of I/O numbers.
- An extension ".FL" is necessary for argument and to storing results of calculation.
- If an argument is an integer, the integer should be converted to a real number before execution of the operation. (Otherwise, you cannot get correct results of calculation.)
- If the operation result is in outside the range from −1e+37 to 1e+37, DER is '1' (DER=1).
- If s.FL ≤ 0, DER is '1' (DER=1) because the calculation is impossible.
- A format of a floating decimal point conforms to IEEE754.

Program example



[Program description]

The logarithm of the real number specified DR210.FL is calculated at the rising edge of R210, which result is substituted for DR212.FL.

PRN ➔ PRJ

This command is equivalent to FUN 118 (s) in the program (PRN file) of EH-CPU.

How to convert the program which has used FUN 118 (s) for EHV is as follows.

FUN 118 (s) ➔ [s+2].FL = LOG (s.FL), provided that ‘s’ and ‘s+2’ are Double word.

e.g) FUN 118 (WR100) ➔ DR102.FL = LOG (DR100.FL)

* If converted by a conversion tool, it is converted as mentioned above.

Name	Common logarithm (Floating decimal point)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d.FL = LOG10 (s.FL)	—	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	484	—	—	

Usable I/O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Result of calculation													✓	✓	✓	
s	Argument												✓	✓	✓	✓	✓

Function

- The common logarithm with base 10 by treating a value of the real number specified ‘s.FL’ is calculated.
- A form of substitution statement is written and the operation result is stored in ‘d.FL’.
  e.g.) DR10.FL = LOG10 (DR0.FL) The common logarithm of DR0.FL is calculated, which result is stored in DR10.FL.
- If the operation is performed normally, DER is ‘0’ (DER=0).

Parameter

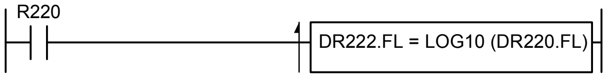
d.FL: Internal output to store results of calculation is specified.

s.FL: An argument is specified.

Cautionary notes

- Internal outputs for argument and to store results of calculation should be specified within the range of I/O numbers.
- An extension “.FL” is necessary for internal outputs for argument and for storing results of calculation.
- If an argument is an integer, the integer should be converted to a real number before execution of the operation. (Otherwise, you cannot get correct results of calculation.)
- If the operation result is in outside the range from −1e+37 to 1e+37, DER is ‘1’ (DER=1).
- If s.FL ≤ 0, DER is ‘1’ (DER=1) because the calculation is impossible.
- A format of a floating decimal point conforms to IEEE754.

Program example



[Program description]

The logarithm of the real number specified by DR220.FL is calculated at the rising edge of R220, which result is substituted for DR220.FL.

[1] Basic commands

[2] Arithmetic commands

[3] Application commands

[4] Control commands

[5] CPU serial communications commands

[6] High-function module transfer commands

Name	[Command support] Coding I/O address

Ladder format	Number of steps		Condition code				
	Maximum	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
d = ADR (s)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Maximum	Time	Maximum	Time	
–	0.24	–	–	

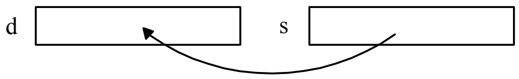
Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to store address value															✓	
s	I/O intended for coding	✓	✓		✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	

Function

- I/O address specified by ‘s’ is coded to store in ‘d’.

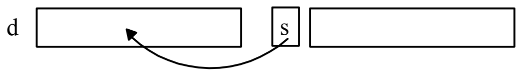
  (This is used combining with commands that require registering I/O address.)

Substitution statement case; d = s



Store data of ‘s’ in ‘d’.

Specify address case; d = ADR (s)



Code address of ‘s’ to store in ‘d’.

Parameter

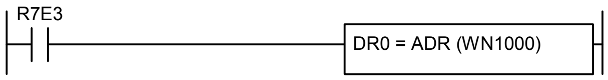
d: The internal output (Double word) to store the value that I/O address is coded to is specified.

s: I/O to code is specified.

Cautionary notes

- ‘d’ can specify only an internal output of double word.
- The address before coding cannot be distinguished even if coded address (the stored value in ‘d’) is monitored. Therefore, it should be checked in the place where the I/O address coding command in a program was described.

Program example



[Program description]

WN1000 is coded at the first scan after RUN to store in DR0.

PRN ➔ PRJ

This command is equivalent to ADRIO(d, s) in the program (PRN file) of EH-CPU.

How to convert the program which has used ADRIO (d, s) for EHV is as follows.

ADRIO (d, s) ➔ d = ADR (s), provided that 's' is Double word.

e.g.) ADRIO (WR100, WR0) ➔ DR100 = ADR (WR0)

* If converted by a conversion tool, it is converted as mentioned above.

[Attention at the time of program conversion]

I/O address after code conversion can be stored into 1-word in EH-CPU, but stored into 1-double word (2-words) in EHV-CPU.

Care must be taken that d parameter does not overlap with an are used currently for another purpose.

Reference

This command is used combining with other command.

Commands that use the coded I/O address are shown in following table.

No.	Command format	Command description
1	BSHR (d, n)	Right shift with byte unit
2	BSHL (d, n)	Left shift with byte unit
3	ASC (d, s, n)	Conversion Binary ➔ ASCII
4	HEX (d, s, n)	Conversion ASCII ➔ Binary
5	WTOB (d, s, n)	Conversion Word ➔ Byte
6	BTOW (d, s, n)	Conversion Byte ➔ Word
7	BITTOW (d, s, n)	Develop Bit data into Word data
8	WTOBIT (d, s, n)	Develop Word data into Bit data
9	SADD (d, s1, s2)	Character string combination
10	SCMP (d, s1, s2)	Character string comparison
11	INTPL (s)	Linear interpolation
12	RECSET (s, n)	Data storage (Initial setting)
13	PIDIT (s)	PID Operation (Initialization)
14	PIDOP (s)	PID Operation (Execution control)
15	PIDCL (s)	PID Operation (Calculation)
16	CCCL (s)	Generating check code
17	CCCMP (d, s)	Collating check code
18	IFR (s)	Process stepping
19	PGEN (s)	Generation of scan pulse
20	TRNS 0 (s, t)	CPU serial communication port Sending data
21	RECV 0 (s, t)	CPU serial communication port Receiving data
22	TRNS 7 (d, s, t)	Sending EH-ID data / Sending command
23	RECV 7 (d, s, t)	Read EH-ID data
24	TRNS 9 (d, s, t)	Sending EH-SIO data / Receiving data
25	FLMEIT (s)	Sending EH-FLN2 user message (Initial setting)
26	EXMEIT (s)	Sending EH-RMD/IOCD Explict message (Initial setting)
27	XYRW (s, t)	Module The 2nd XY area Read / Write
28	SCRW (s, t)	Module Status / Control area Read / Write

Name	[Bit operation] Bit set

Ladder format	Number of steps		Condition code				
	Maximum	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
BSET (d, n)	–	4	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Maximum	Time	Maximum	Time	
d : Word	0.28	–	–	
d : Double word	0.34	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to set bit								✓	✓	✓	✓		✓	✓	✓	
n	Bit position to set							✓	✓	✓	✓	✓					✓

Function

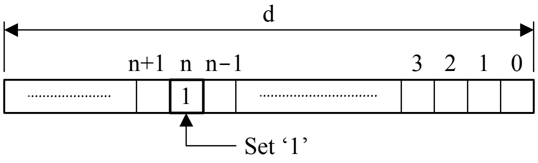
- The n-th bit in I/O (Word or Double word) specified by ‘d’ is set to ‘1’.
- Other bits remain unchanged.
- If ‘d’ is Word,

  A bit position is specified with the content (0 to 15) of the lower 4 bits (b3 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

  The value that can be specified to ‘n’ (a constant) is 0 to 15. (Decimal)
- If ‘d’ is Double word,

  A bit position is specified with the content (0 to 31) of the lower 5 bits (b4 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC) (The upper bits are ignored and it is considered to be ‘0’.)

  The value that can be specified to ‘n’ (a constant) is 0 to 31. (Decimal)



Parameter

d: I/O (word or double word) to set bits is specified.

N: The bit position to set is specified.

Program example

See the explanation pages in ”BTS (d, n)”.

Name	[Bit operation] Bit reset

Ladder format	Number of steps		Condition code				
BRES (d, n)	Maximum	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	–	4	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Maximum	Time	Maximum	Time	
d : Word	0.28	–	–	
d : Double word	0.34	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to set bits								✓	✓	✓	✓		✓	✓	✓	
n	Bit position to set							✓	✓	✓	✓	✓					✓

Function

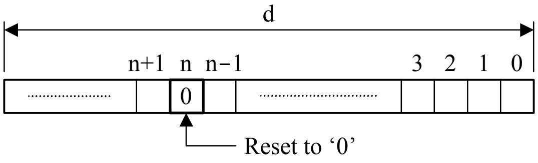
- The n-th bit in I/O (Word or Double word) specified by ‘d’ is set to ‘0’.
- Other bits remain unchanged.
- If ‘d’ is Word,

  A bit position is specified with the content (0 to 15) of the lower 4 bits (b3 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

  The value that can be specified to ‘n’ (a constant) is 0 to 15. (Decimal)
- If ‘d’ is Double word,

  A bit position is specified with the content (0 to 31) of the lower 5 bits (b4 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

  The value that can be specified to ‘n’ (a constant) is 0 to 31. (Decimal)



Parameter

d: I/O (Word or Double word) to set bits is specified.

n: The bit position to reset is specified.

Program example

See the explanation pages in “BTS (d, n)”.

Name	[Bit operation] Bit Test

Ladder format	Number of steps		Condition code				
	Maximum	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
BTS (d, n)	–	4	●	●	●	●	↕

Command processing time (μs)				Remarks
Average		Maximum		
Maximum	Time	Maximum	Time	
d : Word	0.32	–	–	
d : Double word	0.38	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to test bits									✓	✓	✓	✓		✓	✓	✓	
n	Bit position to test								✓	✓	✓	✓	✓					✓

Function

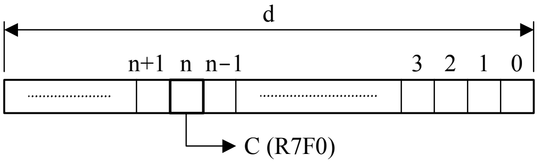
- A content of the n-th bit in I/O(Word or Double word) specified by ‘d’ is checked. If it is ‘1’ as a result, C(R7F0) is set to ‘1’ and if it is ‘0’, C is reset to ‘0’.
- Other bits remain unchanged.
- If d is Word,

  The bit position is specified with the content (0 to 15) of the lower 4 bits (b3 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

  The value that can be specified to ‘n’ (a constant) is 0 to 15. (Decimal)
- If d is Double word,

  The bit position is specified with the content (0 to 31)of the lower 5 bits ‘b4 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

  The value that can be specified to ‘n’ (a constant) is 0 to 31. (Decimal)

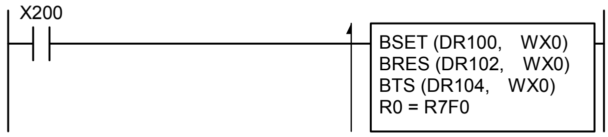


Parameter

d: I/O (Word or Double word) to test bits is specified.

n: The bit position to test is specified.

Program example



[Program description]

The bit operation in the processing box is performed at the rising edge of X200.

Supposing that X200 has turned ON in the state where WX0 : 20 (H0014), DR100 : 0, DR102 : HFFFFFFFF, and DR104 : H5555AAAA,

(1) The 20th bit in DR100 is set to ‘1’ by BSET.

b31											b20					b15															b0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1

(2) The 20th bit in DR102 is reset to ‘0’ by BRES.

b31											b20					b15															b0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

0

(3) The content of the 20th bit in DR104 is set to R7F0 by BTS.

b31											b20					b15															b0
0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0

→ C (R7F0)

[Reference]

The bit position is specified with the value of the lower 4 bits or 5 bits in I/O used as n parameter.

If the value of WX0 is H1234 in the program example mentioned above, since the lower 5 bits are valid, the bit position will bring the same result because of being 20(H0014).

	b15															b0
WX0	0	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0
	Invalid											Valid				

Application

BTS (d, n)

Name	[Bit operation] Bit Count

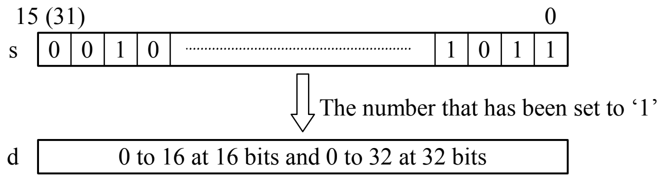
Ladder format	Number of steps		Condition code				
	Maximum	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
BCU (d, s)	Word	4	●	●	●	●	●
	Double word	5					

Command processing time (μs)				Remarks
Average		Maximum		
Maximum	Time	Maximum	Time	
d : Word	3	—	—	
d : Double word	5.4	—	—	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to store number of 1								✓	✓	✓	✓					
s	I/O to count bit of 1							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

The number of bits set to ‘1’ with a content of ‘s’ (16 bits at Word, 32 bits at Double word) is stored in ‘d’.

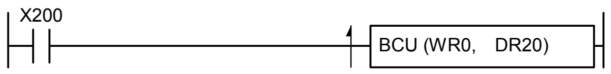

15 (31)
0
s
0 0 1 0
1 0 1 1
The number that has been set to ‘1’
d
0 to 16 at 16 bits and 0 to 32 at 32 bits

Parameter

d: I/O to store the number of ‘1’ contained in I/O specified by ‘s’ is specified.

s: I/O to count the number of ‘1’ or a constant is specified.

Program example



[Program description]

The number of ‘1’ in data of DR20 is counted at the rising edge of X200. And the result is set to WR0.

Supposing that X200 has turned ON in the state where DR20 : H12345678, WR0 is set to ‘13’.

Name	[Shift / Rotate] Shift right

Ladder format	Number of steps		Condition code				
	Maximum	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
SHR (d, n)	–	4	●	●	●	●	↕

Command processing time (μs)				Remarks
Average		Maximum		
Maximum	Time	Maximum	Time	
d : Word	0.9	–	–	
d : Double word	1.5	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to shift								✓	✓	✓	✓		✓	✓	✓	
n	Number of bits to shift							✓	✓	✓	✓	✓					✓

Function

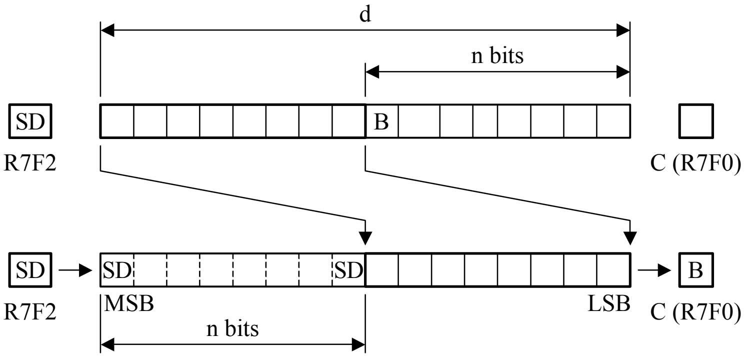
- The content of ‘d’ is shifted to the right (the lower direction) n bits.
- Data of SD(R7F2) is stored in n bits from the Most Significant Bit (MSB).
- The content of the n-th bit from the Least Significant Bit (LSB) is stored in C(R7F0).
- If ‘d’ is Word,

  Volume to shift is specified with the content (0 to 15) of the lower 4 bits (b3 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

  The value that can be specified to ‘n’ (a constant) is 0 to 15. (Decimal)
- If ‘d’ is Double word,

  Volume to shift is specified with the content (0 to 31) of the lower 5 bits (b4 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

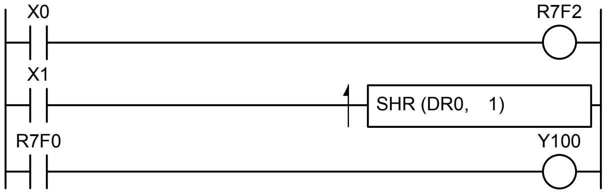
  The value that can be specified to ‘n’ (a constant) is 0 to 31. (Decimal)



Cautionary notes

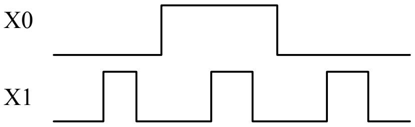
If n=0, it does not shift. C retains the previous state.

Program example

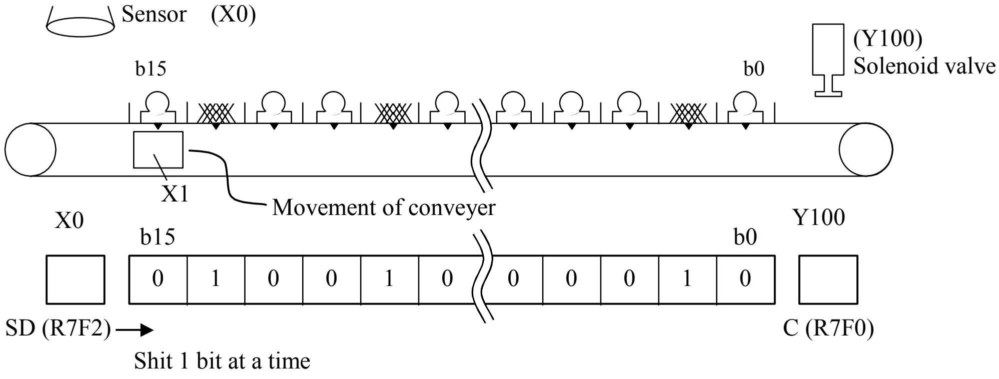


[Program description]

- There is a conveyer with 16 stands and it is moving to the right direction.
- Whenever a stand moves to the one right, one pulse input goes into X1.
- If inferior goods are put on the conveyer, X0 is turned ON because there is a sensor on the left end of the conveyer. The signal of X0 (a sensor input) and X1 (a conveyer move) is as follows.



- Data is also shifted 1 bit at a time with the movement to the right, and inferior goods are expelled out at the place (the right end of the conveyer) where data has come out to the carry because a solenoid valve (Y100) is turned ON.



Name	[Shift / Rotate] Shift left

Ladder format	Number of steps		Condition code				
	Maximum	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
SHL (d, n)	—	4	●	●	●	●	↕

Command processing time (μs)				Remarks
Average		Maximum		
Maximum	Time	Maximum	Time	
d : Word	0.9	—	—	
d : Double word	1.5	—	—	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to shift								✓	✓	✓	✓		✓	✓	✓	
n	Number of bits to shift							✓	✓	✓	✓	✓					✓

Function

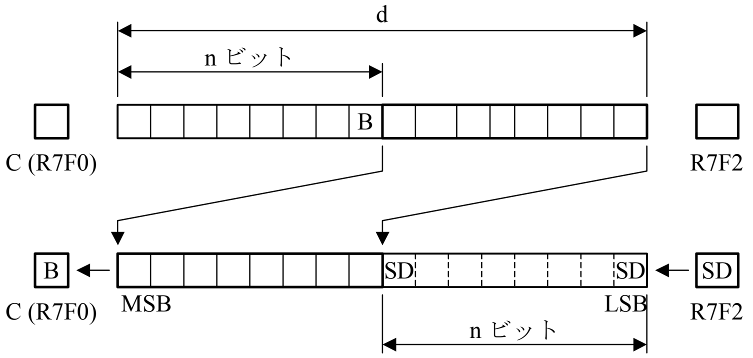
- The content of 'd' is shifted to the left (the upper direction) n bits.
- Data of SD(R7F2) is stored in n bits from the Least Significant Bit (LSB).
- The content of the n-th bit from MSB is stored in C(R7F0).
- If 'd' is Word,

  Volume to shift is specified with the content (0 to 15) of the lower 4 bits (b3 to b0) in 'n' (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be '0'.)

  The value that can be specified to 'n' (a constant) is 0 to 15. (Decimal)
- If 'd' is Double word,

  Volume to shift is specified with the content (0 to 31) of the lower 5 bits (b5 to b0) in 'n' (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be '0'.)

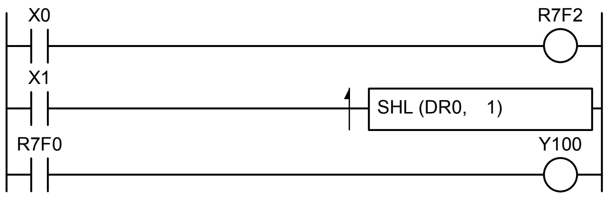
  The value that can be specified to 'n' (a constant) is 0 to 31. (Decimal)



Cautionary notes

If n=0, it does not shift. C retains the previous state.

Program example



[Program description]

- The value of R7F2 is determined by ON/OFF of X0.
- The content of DR0 is shifted to the left 1 bit at the rising of X1.
  In this case, the value of R7F2 is put into b0 and the value of b31 (b15 of WR1) is put into R7F0.
  Y100 is tuned ON/OFF from the value of b31 (b15 of WR1) of DR0 before the shift.

Name	[Shift / Rotate] Rotate right

Ladder format	Number of steps		Condition code				
	Maximum	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
ROR (d, n)	–	4	●	●	●	●	↕

Command processing time (μs)				Remarks
Average		Maximum		
Maximum	Time	Maximum	Time	
d : Word	0.9	–	–	
d : Double word	1.5	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to rotate								✓	✓	✓	✓		✓	✓	✓	
n	Number of bits to rotate							✓	✓	✓	✓	✓					✓

Function

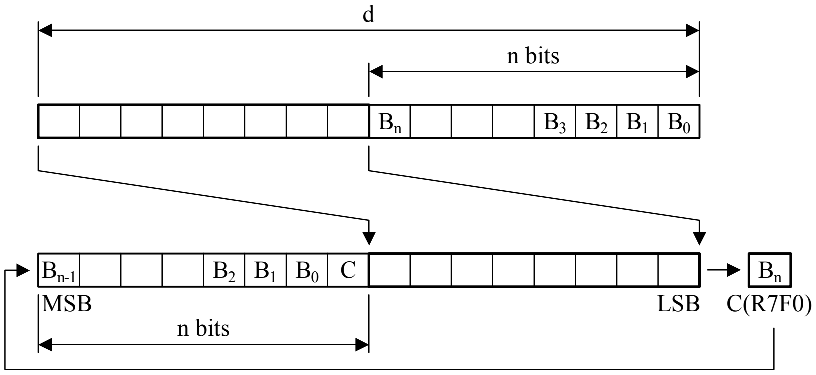
- The content of ‘d’ is rotated to the right (the lower direction) n bits.
- The content of C(R7F0) is put into MSB, at a same time, the content of LSB is stored in C(R7F2). This processing is repeated n times.
- The content of the n-th bit from LSB is stored in C(R7F0).
- If ‘d’ is Word,

  Volume to shift is specified with the content (0 to 15) of the lower 4 bits (b3 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

  The value that can be specified to ‘n’ (a constant) is 0 to 15. (Decimal)
- If ‘d’ is Double word,

  Volume to shift is specified with the content (0 to 31) of the lower 5 bits in ‘n7’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

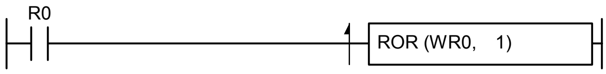
  The value that can be specified to ‘n’ (a constant) .



Cautionary notes

If n=0, it does not rotate. C retains the previous state.

Program example



[Program description]

- WR0 is shifted to the right 1 bit at the rising of R0.

  In this case, the value of LSB b0 is put into R7F0 and the value of R7F just before shifting is put into MSB b15.

Name	[Shift / Rotate] Rotate left

Ladder format	Number of steps		Condition code				
	Maximum	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
ROL (d, n)	—	4	●	●	●	●	↕

Command processing time (μs)				Remarks
Average		Maximum		
Maximum	Time	Maximum	Time	
d : Word	0.9	—	—	
d : Double word	1.5	—	—	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to rotate								✓	✓	✓	✓		✓	✓	✓	
n	Number of bits to rotate							✓	✓	✓	✓	✓					✓

Function

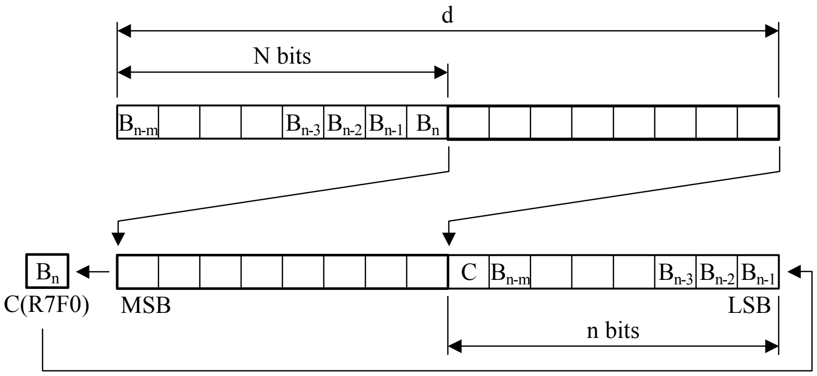
- The content of ‘d’ is rotated to the left (the upper direction) n bits.
- The content of C(R7F0) is put into LSB, and at a same time, the content of MSB is stored into C(R7F0). This processing is repeated n times.
- The content of the n-th bit from MSB can be stored in C(R7F0).
- If ‘d’ is Word,

  Volume to shift is specified with the content (0 to 15) of the lower 4 bits (b3 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

  The value that can be specified to ‘n’ (a constant) is 0 to 15. (Decimal)
- If ‘d’ is Double word,

  Volume to shift is specified with the content (0 to 31) of the lower 5 bits (b4 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

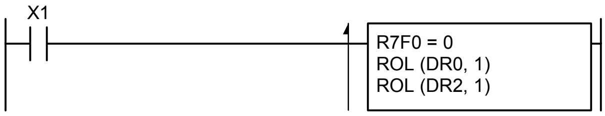
  The value that can be specified to ‘n’ (a constant) is o to 31. (Decimal)



Cautionary notes

If n=0, it does not rotate. C retains the previous state.

Program example

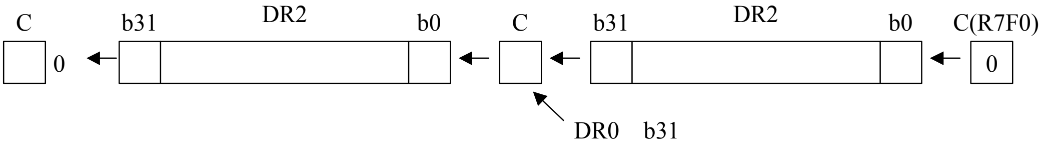


[Program description]

- 64-bit data is shifted 1 bit at a time at the rising of X1.

  ‘0’ is put into the opened space by shifting.

  The whole movement



Name	[Shift / Rotate] Logic shift right

Ladder format	Number of steps		Condition code				
	Maximum	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
LSR (d, n)	—	4	●	●	●	●	↕

Command processing time (µs)				Remarks
Average		Maximum		
Maximum	Time	Maximum	Time	
d : ワード	0.9	—	—	
d : ダブルワード	1.4	—	—	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to shift								✓	✓	✓	✓		✓	✓	✓	
n	Number of bits to shift							✓	✓	✓	✓	✓					✓

Function

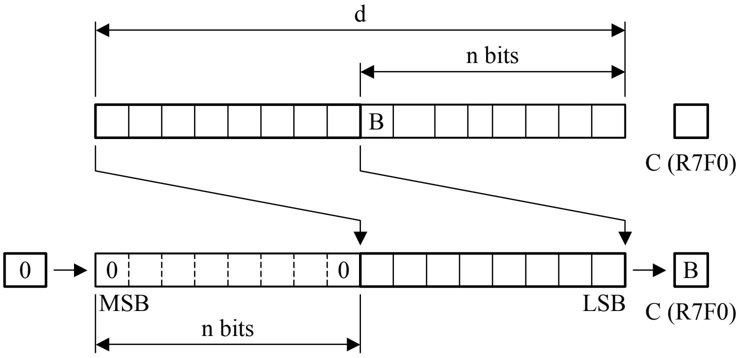
- The content of ‘d’ is shifted to the right (the lower direction) n bits.
- n bits from MSB store ‘0’ respectively.
- The content of the b-th bit from LSB is stored in C(R7F0).
- If ‘d’ is Word,

  Volume to shift is specified with the content (0 to 15) of the lower 4 bits (b3 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

  The value that an can be specified to ‘n’ (a constant) is 0 to 15. (Decimal)
- If ‘d’ is Double word,

  Volume to shift is specified with the content (0 to 31) of the lower 5 bits in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

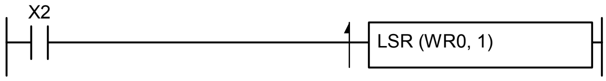
  The value that can be specified to ‘n’ (a constant) is 0 to 31. (Decimal)



Cautionary notes

If n=0, it does not shift. C retains the previous state.

Program example



[Program description]

- The value of WR is shifted to the right 1 bit at the rising of X2.

  In this case, ‘0’ is put into b15 and the value of b0 before a shift is put into R7F0.

Name	[Shift / Rotate] Logic shift left

Ladder format	Number of steps		Condition code				
	Maximum	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
LSL (d, n)	–	4	●	●	●	●	↕

Command processing time (μs)				Remarks
Average		Maximum		
Maximum	Time	Maximum	Time	
d : ワード	0.9	–	–	
d : ダブルワード	1.4	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to shift								✓	✓	✓	✓		✓	✓	✓	
n	Number of bits to shift							✓	✓	✓	✓	✓					✓

Function

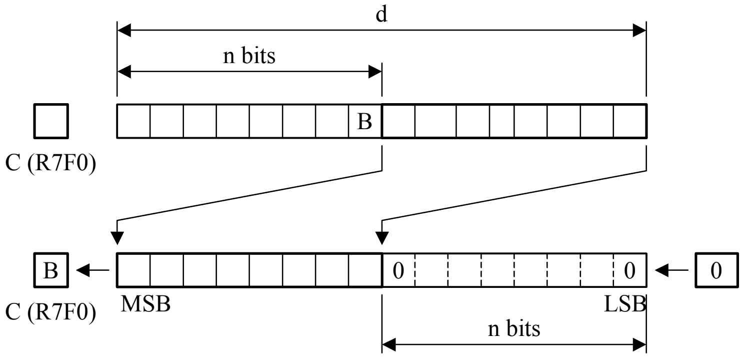
- The content of ‘d’ is shifted to the left (the upper direction) n bits.
- n bits from LSB store ‘0’ respectively.
- The content of the n-th bit from MSB is stored in C(R7F0).
- If ‘d’ is Word,

  Volume to shift is specified with the content (0 to 15) of the lower 4 bits (b3 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

  The value that can be specified to ‘n’ (a constant) is 0 to 15. (Decimal)
- If ‘d’ is Double word,

  Volume to shift is specified with the content (0 to 31) of the lower 5 bits (b4 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

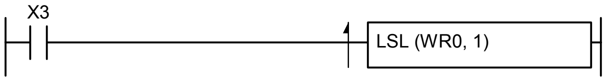
  The value that can be specified to ‘n’ (a constant) is 0 to 31. (Decimal)



Cautionary notes

If n=0, it does not shift. C retains the previous state.

Program example



[Program description]

- The value of WR0 is shifted to the left 1 bit.

  In this case, ‘0’ is put into b0 and the value of b15 before a shift is put into R7F0.

Name	[Shift / Rotate] BCD shift right

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
BSR (d, n)	—	4	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
d : Word	0.38	—	—	
d : Double word	1	—	—	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to shift								✓	✓	✓	✓		✓	✓	✓	
n	Number of digits to shift							✓	✓	✓	✓	✓					✓

Function

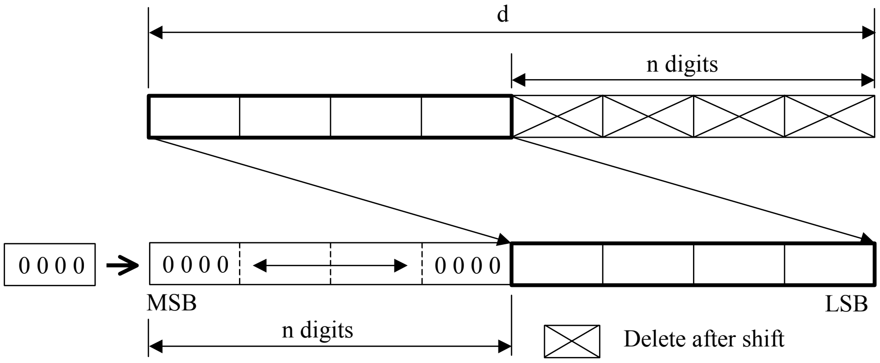
- The content of ‘d’ is shifted to the right (the lower direction) n digits. (One digit has 4 bits.)
- n digits from the most significant digits store ‘0’ respectively.
- n digits from the least significant digits are deleted.
- If ‘d’ is Word,

  Volume to shift is specified with the content (0 to 3) of the lower 2 bits (b1 and b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

  The value that can be specified to ‘n’ (a constant) is 0 to 3. (Decimal)
- If ‘d’ is Double word,

  Volume to shift is specified with the content (0 to 7) of the lower 3 bits (b2 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

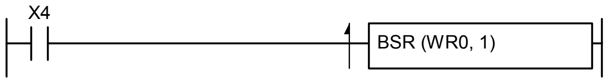
  The value that can be specified to ‘n’ (a constant) is 0 to 7. (Decimal)



Cautionary notes

If n=0, it does not shifted.

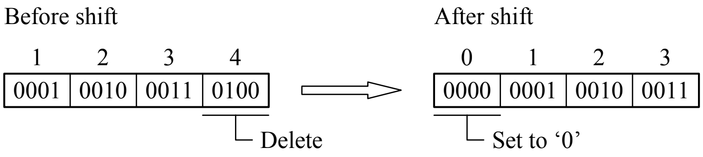
Program example



[Program description]

The content of WR0 which is considered to be BCD code is shifted to the right one digit (4 bits) at the rising of X4.

In this case, data in lower 4 bits (b3 to b0) is deleted and the upper 4 bits (b15 to b12) are set to ‘0000’.



Before shift

1	2	3	4
0001	0010	0011	0100

Delete

After shift

0	1	2	3
0000	0001	0010	0011

Set to ‘0’

Name	[Shift / Rotate] BCD Shift left

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
BSL (d, n)	—	4	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
d : Word	0.38	—	—	
d : Double word	1	—	—	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to shift								✓	✓	✓	✓		✓	✓	✓	
n	Number of digits to shift							✓	✓	✓	✓	✓					✓

Function

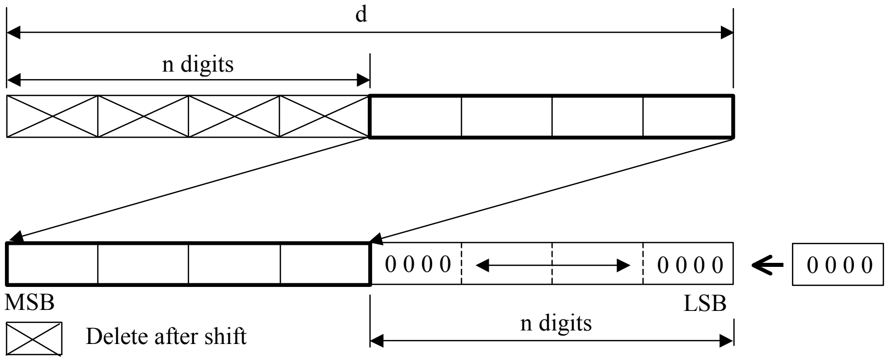
- The content of ‘d’ is shifted to the left (the upper direction) n digits. (One digit has 4 bits.)
- n digits from the least significant digit store ‘0’ respectively.
- n digits form the most significant digit are deleted.
- If ‘d’ is Word,

  Volume to shift is specified with the content (0 to 3) of lower 2 bits (b1 and b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

  The value that can be specified to ‘n’ (a constant) is 0 to 3. (Decimal)
- If ‘d’ is Double word,

  Volume to shift is specified with the content (0 to 7) of the lower 3 bits (b2 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

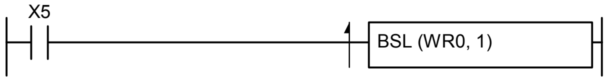
  The value that can be specified to ‘n’ (a constant) is 0 to 7. (Decimal)



Cautionary notes

If n=0, it does not shift.

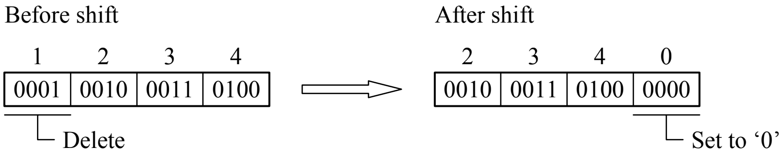
Program example



[Program description]

The content of WR0 which is considered to be BCD code is shifted to the left one digit (4 bits) at the rising of X5.

In this case, data in the upper 4 bits is deleted and the lower 4 bits are set to ‘0000’.



Name	[Shift / Rotate] Batch shift right

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
WSHR (d, n)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
d : Bit	2.09+0.01n	–	–	
d : Word	1.48+0.22n	–	–	

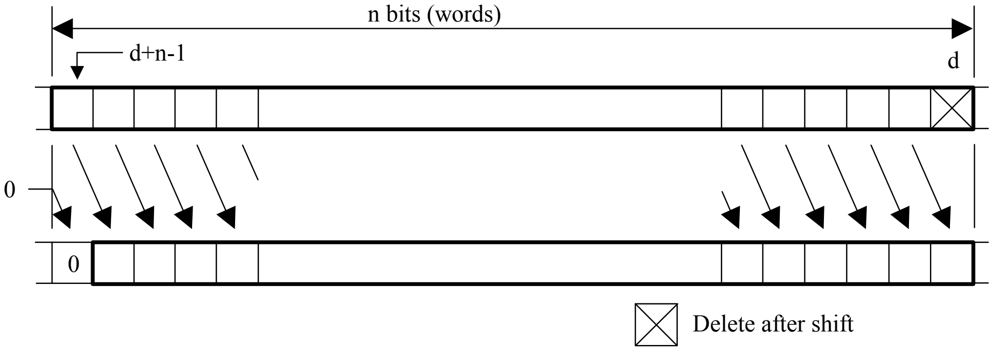
Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to shift				✓					✓	✓						
n	Number of bits to shift Number of words to shift							✓	✓	✓	✓	✓					✓

Function

- n bits (word) from ‘d’ is shifted to the right (the direction where I/O number is small) one bit (word).
- The content of the bit (word) specified by ‘d’ is deleted.
- ‘d+n-1’ which is in n bits (words) of ‘d’ stores ‘0’.
- If ‘d’ is Word,

  Bit (word) volume to shift is specified with the content (0 to 255) of lower 8 bits (b7 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)
- If ‘n’ is a constant,

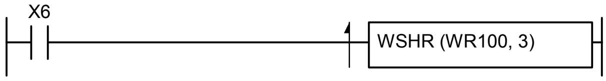
  Bit (word) volume to shift is specified. From 0 to 255 is a valid range.



Cautionary notes

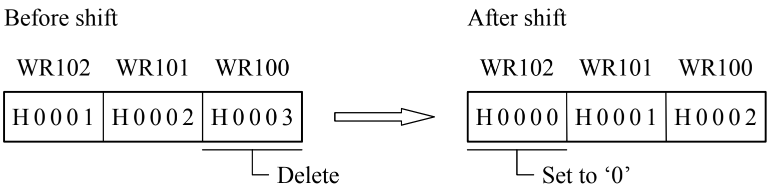
- If n=0, it does not shift. DER is set to ‘0’.
- ‘d+n−1’ should be used within the I/O range. If exceeded, DER=1 and it is shifted from ‘d’ to the maximum range.

Program example



[Program description]

The contents of WR100, WR101 and WR102 which is considered to be BCD code are shifted to the right one word at the rising of X6.



Before shift

WR102	WR101	WR100
H0001	H0002	H0003

After shift

WR102	WR101	WR100
H0000	H0001	H0002

Name	[Shift / Rotate] Batch shift left

Ladder format	Number of steps		Condition code				
	Condition	steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
WSHL (d, n)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
d : Bit	2.09+0.01n	–	–	
d : Word	1.38+0.22n	–	–	

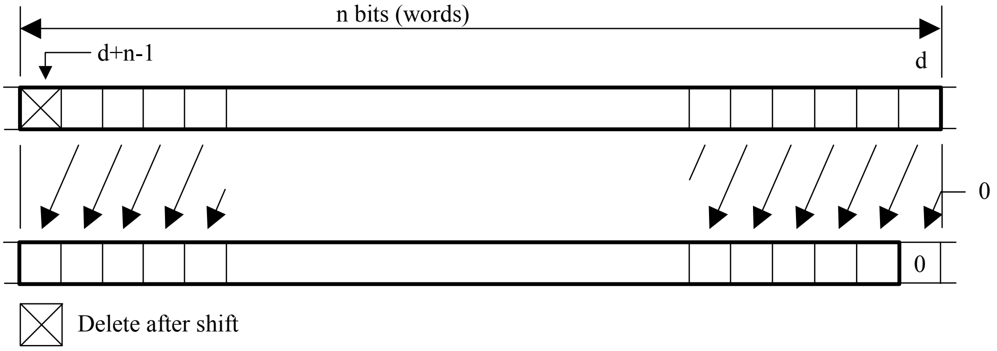
Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to shift				✓					✓	✓						
n	Number of bits to shift Number of words to shift							✓	✓	✓	✓	✓					✓

Function

- n bits (words) from 'd' are shifted to the left (the direction where I/O number is large) one bit (word).
- The bit (word) specified by 's' stores '0'.
- The content of 'd+n-1' which is in n bits ahead of 'd' is deleted.
- If 'n' is Word,

  Bit (word) volume to shift is specified the content (0 to 255) of lower 8 bits (b7 to b0) in 'n' (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be '0'.)
- If 'n' is a constant,

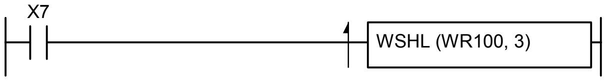
  Bit (word) volume to shift is specified. From 0 to 255 is a valid range.



Cautionary notes

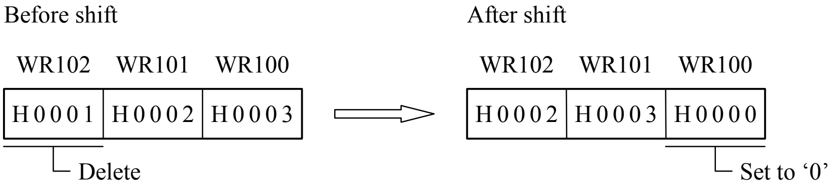
- If n=0, it does not shift and DER is set to '0'.
- 'd+n−1' should be used within the I/O range. If exceeded, DER=1 and it is shifted from 'd' to the maximum range.

Program example



[Program description]

The contents of WR100, WR101 and WR102 which is considered to be BCD code are shifted to the left one word at the rising of X7.



Before shift

WR102	WR101	WR100
H0001	H0002	H0003

Delete

After shift

WR102	WR101	WR100
H0002	H0003	H0000

Set to ‘0’

Name	[Shift / Rotate] BCD batch shift right

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
WBSR (d, n)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		A constant is specified with decimal.
Condition	Time	Condition	Time	
–	1.42+0.28n	–	–	

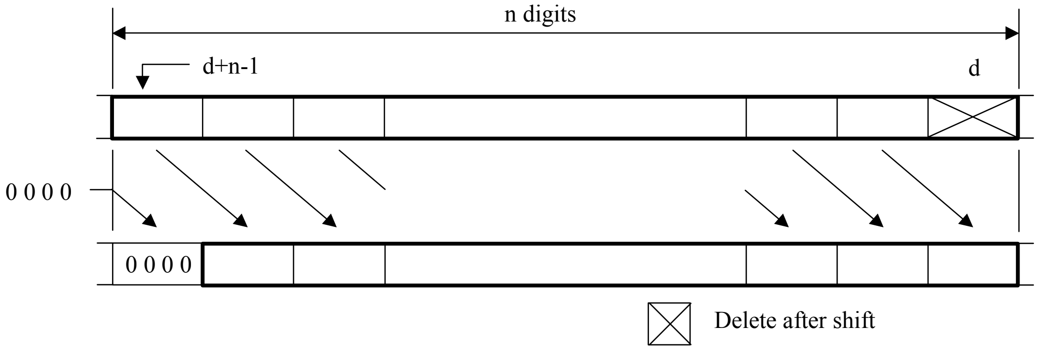
Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to shift									✓	✓						
n	Number of words intended for shift							✓	✓	✓	✓	✓					✓

Function

- n words from ‘d’ which is considered to be BCD data consisting of 4n digits is shifted to the right (the direction where I/O number is small) one digit. (One digit has 4 bits.)
- The content of the least significant digit which is in the specified words ahead of ‘d’ is deleted.
- The most significant digit (the upper 4 bits) which is in n words ahead of ‘d’ stores ‘0’.
- If ‘n’ is Word,

  The number of digits to shift is specified with the content (0 to 255) of lower 8 bits (b7 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)
- If ‘n’ is a constant,

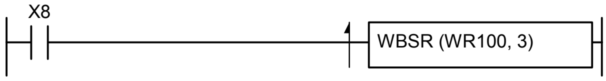
  The number of digits to shift is specified. From 0 to 255 is a valid range.



Cautionary notes

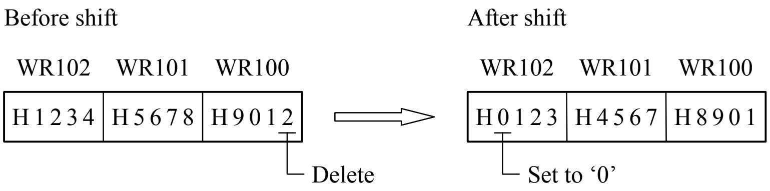
- If n=0, it does not shift, and DER is set to ‘0’.
- ‘d+n−1’ should be used within the I/O range. If exceeded, DER=1 and it is shifted from ‘d’ to the maximum range.

Program example



[Program description]

The contents of WR100, WR101 and WR102 which is considered to be BCD code are shifted to the right 4 bits at the rising of X8.



Name	[Shift / Rotate] BCD batch shift left

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
WBSL (d, n)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		A constant is specified with decimal.
Condition	Time	Condition	Time	
–	1.42+0.28n	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to shift										✓	✓						
n	Number of words intended for shift								✓	✓	✓	✓	✓					✓

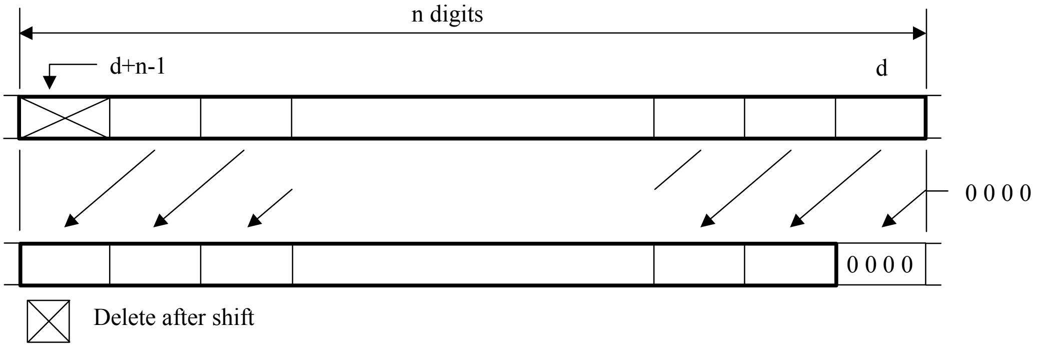
Function

- n words from ‘d’ which is considered to be BCD data consisting of 4n digits is shifted to the left (the direction where I/O number is large) one digit. (One digit has 4 bits.)
- The content of the least significant digit in words specified by ‘d’ stored ‘0’.
- The most significant digit (the upper 4 bits) which is in n words ahead of ‘d’ is deleted.
- If ‘n’ is Word,

The number of digits to shift is specified with the content (0 to 255) of the lower 8 bits (b7 to b0) in ‘n’ (WX, WEX, WY, WEY, WR, WL, WM, WN, TC). (The upper bits are ignored and it is considered to be ‘0’.)

- If ‘n’ is a constant,

The number of digits to shift is specified. From 0 to 255 is a valid range.



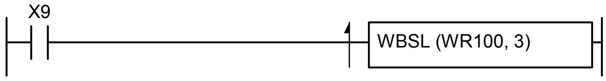
Cautionary notes

- If n=0, it does not shift, and DER is set to ‘0’.
- ‘d+n-1’ should be used within the I/O range. If exceeded, DER=1, and it is shifted from ‘d’ to the maximum range.

Application

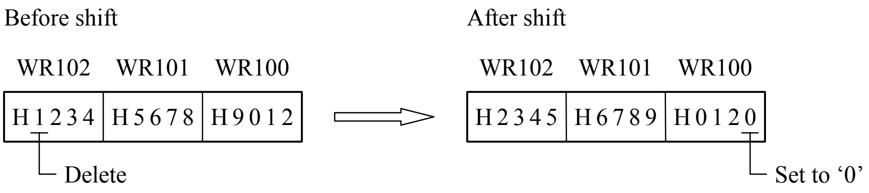
WBSL (d, n)

Program example



[Program description]

The contents of WR100, WR101 and WR102 which is considered to be BCD code are shifted to the left 4 bits at the rising of X9.



Name	[Shift / Rotate] Character data One byte right shift

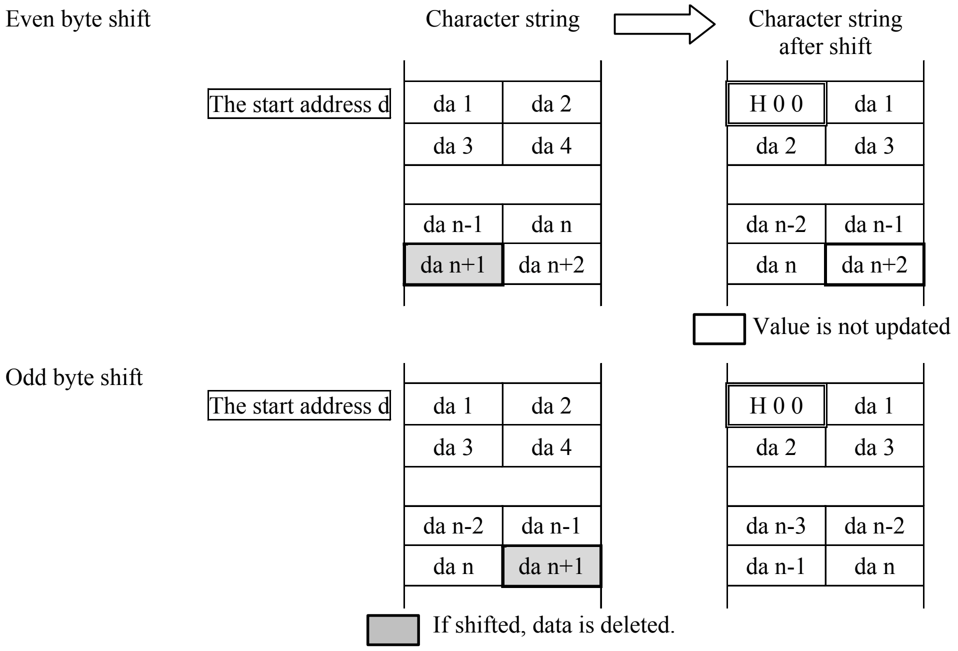
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
BSHR (d, n)	—	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		A constant is specified with decimal.
Condition	Time	Condition	Time	
—	4.85+0.15n	—	—	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	The start I/O of character string intended for shift											✓						
n	Number of bytes intended for shift								✓	✓	✓	✓	✓					✓

Function

- n bytes in a character string data whose the start is address d is shifted to the right one byte.

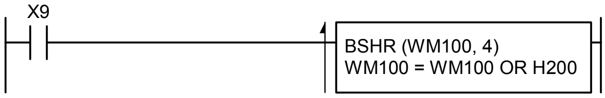


- An opened space after the shift stores H00.
- The next data to the specified number of bytes is deleted by the shift.

Cautionary notes

- The internal output to use for character string should be used within the I/O number. If exceeded the maximum of I/O number, DER=1 and the operation is not performed.
- If n=0, it does not shift, and DER=0.

Program example



[Program description]

Assume that 4 bytes of the sending data are stored at WM100 or after WM100.

- The data consisting 4 bytes whose the start is WM100 is shifted to the right one byte at the rising of X9.
- The communication control code STX (H01) is added to the head of data.

WM100	“ T ”	“ E ”
WM101	“ X ”	“ T ”
WM102		

⇨

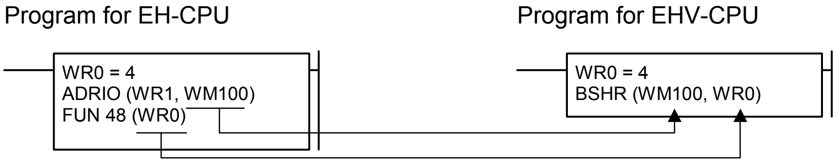
WM100	STX	“ T ”
WM101	“ E ”	“ X ”
WM102	“ T ”	

PRN ➔ PRJ

This command is equivalent to FUN 48 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 48 (s) for EHV, how to convert is as follows.

FUN 48 (s) ➔ BSHR ([I/O specified by ‘s+1’], s)



* This command is not convertible in a conversion tool. Please convert as mentioned above by users, yourselves

Name	[Shift / Rotate] Character data One byte left shift

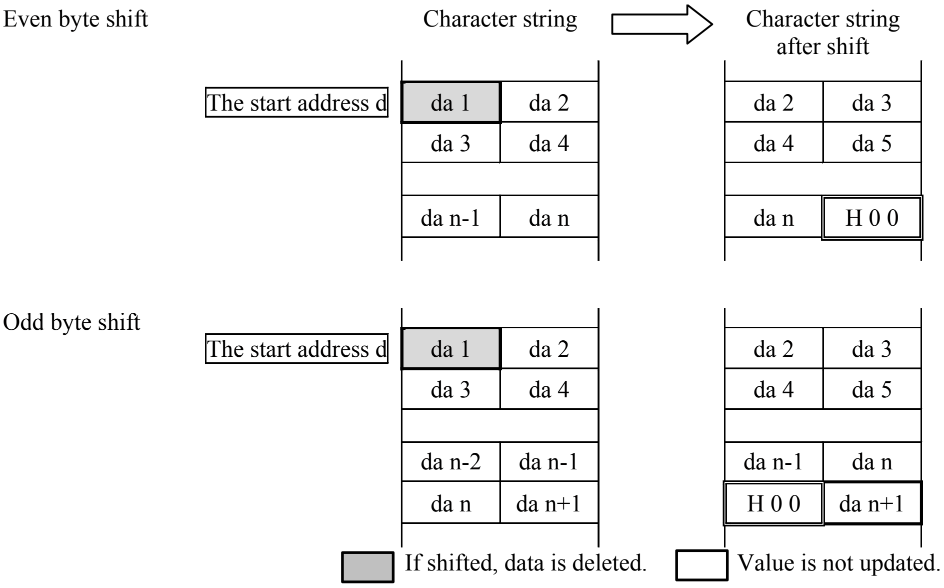
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
BSHL (d, n)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		A constant is specified with decimal
Condition1	Time	Condition	Time	
–	0.28+0.22n	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	The start I/O of character string intended for shift											✓						
n	Number of bytes intended for shift								✓	✓	✓	✓	✓					✓

Function

- n bytes in character string data whose the start is address d is shifted to the left one byte.



Even byte shift

Character string ⇒ Character string after shift

The start address d	Character string		Character string after shift	
	da 1	da 2	da 2	da 3
	da 3	da 4	da 4	da 5
	da n-1	da n	da n	H 0 0

Odd byte shift

The start address d	Character string		Character string after shift	
	da 1	da 2	da 2	da 3
	da 3	da 4	da 4	da 5
	da n-2	da n-1	da n-1	da n
	da n	da n+1	H 0 0	da n+1

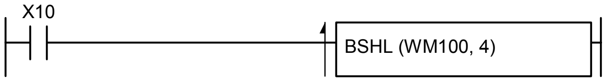
▨ If shifted, data is deleted. □ Value is not updated.

- An opened space after the shift stores H00.
- The head data in character string is deleted by the shift.

Cautionary notes

- The internal output to use for character string should be used within the I/O number. If exceeded the maximum of I/O number, DER=1 and the operation is not performed.
- If n=0 or n=1, it does not shift, and DER=0.

Program example



[Program description]

Assume that 5 bytes of the receiving data with a control code are stored at WM100 or after WM100.

- The data consisting of 5 bytes whose the start is WM100 is shifted to the left one byte at the rising of X10.
- Since the control code is deleted, only the receiving data remains

WM100	STX	“ T ”
WM101	“ E ”	“ X ”
WM102	“ T ”	

⇨

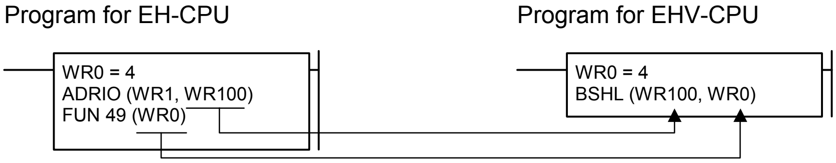
WM100	“ T ”	“ E ”
WM101	“ X ”	“ T ”
WM102	H00	

PRN ➔ PRJ

This command is equivalent to RUN 49 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 49 (s) for EHV, how to convert is as follows.

FUN 49 (s) ➔ BSHL ([I/O specified by ‘s+1’], s)



* This command is not convertible in a conversion tool. Please convert as mentioned above by users, yourselves.

Name	[Conversion of Character] Conversion Binary ➔ BCD

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
BCD (d, s)	Word	4	↕	●	●	●	●
	Double word	5					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	1.9	—	—	
Double word	5.5	—	—	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O (BCD) after conversion								✓	✓	✓			✓	✓	✓	
s	I/O (Binary) before conversion							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- The content of ‘s’ in binary is converted to BCD and the result is stored in ‘d’.
- If the result which converted ‘s’ exceeds the BCD digits, DER is ‘1’ and it is not performed.
  If ‘s’ is Word, H0000 ≤ s ≤ H270F (0～9,999).
  If ‘s’ is Double word, H00000000 ≤ s ≤ H5F5E0FF (0～99,999,999).
- Combinations of ‘d’ and ‘s’ are as follows.

d	s
Word	Word
Double word	Double word

s	0	0	1	0	0	1	1	0	1	0	0	1	0	1	0	0	H2694 (9,876)

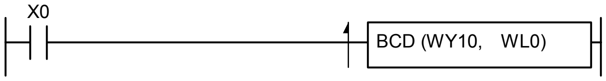
⇩

d	1	0	0	1	1	0	0	0	0	1	1	1	0	1	1	0	H9876
	15															0	
	9				8				7				6				

Cautionary notes

When the data error occurs, the content of ‘d’ remains unchanged as it was before the command was executed.

Program example



[Program description]

The content of WL0 in binary is converted to BCD at turning on X0 and the result is output to WY10.

WL0 | H1B4F (6991) | ⟹ WY10 | H6991 |

Name	[Conversion of Character] Conversion BCD ➔ Binary

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
BIN (d, s)	Word	4	↕	●	●	●	●
	Double word	5					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	1.8	–	–	
Double word	4.2	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O (Binary) after conversion									✓	✓	✓			✓	✓	✓	
s	I/O (BCD) before conversion								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- The content of ‘s’ in BCD is converted to binary and the result is stored in ‘d’.
- If the content of ‘s’ is not BCD data (if the content of ‘s’ contains symbols A – F), DER is ‘1’ and it is not performed.
- Combinations of ‘d’ and ‘s’ are as follows.

d	s
Word	Word
Double word	Double word

s	0	0	0	1	0	0	1	0	0	0	1	1	0	1	0	0	H1234

⇩

d	0	0	0	0	0	1	0	0	1	1	0	1	0	0	1	0	1,234 (H04D2)
	15															0	

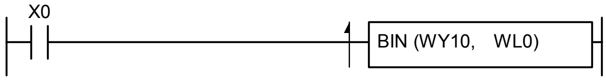
||

1,234 (Decimal)

Cautionary notes

When the data error occurs, the content of ‘d’ remains unchanged as it was before the command was executed.

Program example



[Program description]

The content of WL0 in BCD is converted to binary at turning on X0 and the result is output to WY10.

WL0 H6991 ⇒ WY10 H1B4F (6991)

Name	[Conversion of Character] Conversion 16-bit unsigned binary ➔ ASCII in decimal

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
BINDA (d, s)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		d can be use up to d+2.
Condition	Time	Condition	Time	
–	3.1	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O after conversion								✓	✓	✓			✓	✓	✓	
s	I/O before conversion							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- A 16-bit unsigned binary data is converted to a 5-digit ASCII in decimal code.
- A digit which does not have a number as a result of performing zero suppression to the converted result is set to H20 (a space). And surplus one byte after conversion to ASCII is set to NULL which means an end of a character string.

Parameter

d : The start I/O of a table to store ASCII data in decimal after conversion is specified.

s : The internal output which stores 16-bit unsigned binary data to convert or a constant are specified.

16-bit unsigned binary data

s	0 ～ 65,535

⇨

Decimal ASCII data

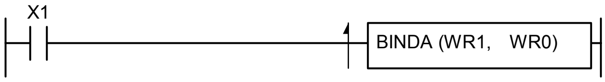
	15 8	7 0
d	10^4	10^3
d+1	10^2	10^1
d+2	10^0	NULL

10^n : ASCII code of 10^n positions

Cautionary notes

The internal output to use for d and s parameters should be specified within the range of I/O number.

Program example



[Program description]

A value (16-bit unsigned binary data) of WR0 is converted to ASCII data in decimal at the rising of X1, and the result is set from WR1 to WR3. 0

If WR0 = 12345, WR1=H3132, WR2=H3334, and WR3=H3500

PRN ➔ PRJ

This command is equivalent to FUN 30 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 30 (s) for EHV, how to convert is as follows.

FUN 30 (s) ➔ BINDA (s+1, s)

Example) FUN 30 (WR100) ➔ BINDA (WR101, WR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	[Conversion of Character] Conversion 32-bit singed binary ➔ ASCII in decimal

Ladder format	Number of steps		Condition code				
	Condition	Step	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
SBINDA (d, s.S)	—	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		d can be used up to d+5.
Condition	Time	Condition	Time	
—	10.9	—	—	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O after conversion									✓	✓	✓						
s.S	I/O before conversion													✓	✓	✓	✓	✓

Function

- 32-bit signed binary data is converted to a 10-digit ASCII code in decimal.
- A digit which does not have a number as a result of performing zero suppression to the converted result is set to H20 (a space). And surplus one byte after conversion to ASCII is set to NULL which means an end of a character string.

Parameter

d : The start I/O of a table to store ASCII data in decimal after conversion is specified.

s.S : The internal output stores 32-bit signed binary data or a constant is specified.

32-bit signed binary data

s -2,147,483,648 ～ 2,147,483,647

Specify by Double word

ASCII data in decimal

	15 8	7 0
d	Sign	10^9
d+1	10^8	10^7
d+2	10^6	10^5
d+3	10^4	10^3
d+4	10^2	10^1
d+5	10^0	NULL

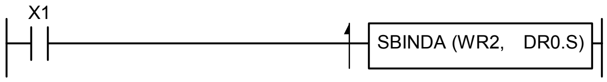
Sign Positive : H20 (space)
Negative : H2D ("-")

10^n : ASCII code pf 10^n position

Cautionary notes

The internal output to use for d and s parameters should be specified within the range of I/O number.

Program example



[Program description]

A value (32-bit signed binary data) of DR10.S is converted to ASCII data in decimal at the rising of X1, and the result is set from WR12 to WR17.

If DR10.S = −1234567, WR12=H2D20, WR13=H2020, WR14=H3132, WR15=H3334, WR16=H3536, and WR17=H3700.

PRN ➔ PRJ

This command is equivalent to FUN 31 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 31 (s) for EHV, how to convert is as follows.

FUN 31 (s) ➔ SBINDA (s+2, s.S) , provided that ‘s.S’ is double word.

Example) FUN 31 (WR100) ➔ SBINDA (WR102, DR100.S)

* If converted by a conversion tool, it is converted as mentioned above.

Application
SBINDA (d, s.S)

Name	[Conversion of Character] Conversion Unsigned binary ➔ ASCII in hexadecimal

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
BINHA (d, s)	Word	4	↕	●	●	●	●
	Double word	5					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	2.2	—	—	
Double word	3.3	—	—	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O after conversion								✓	✓	✓	✓					
s	I/O before conversion							✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- Unsigned binary data specified by ‘s’ is converted to ASCII code in hexadecimal.
  If ‘s’ is Word, 16-bit unsigned binary data is converted 4-digit ASCII code in hexadecimal.
  If ‘s’ is Double word, 32-bit unsigned binary data is converted to 8-digit ASCII code in hexadecimal.
- Zero suppression is not performed to the converted result. And NULL behind ASCII data means an end of a character string.

Parameter

d : The start I/O of a table to store ASCII data in hexadecimal after conversion is specified.

s : Case of Word: The word internal output which stores 16-bit unsigned binary data to convert or a constant is specified.

Case of Double word: The word internal output which stores 32-bit unsigned binary data to convert or a constant is specified.

16-bit unsigned binary data

s	H0000 ～ HFFFF

⇨

ASCII data in hexadecimal

d	16^3	16^2
d+1	16^1	16^0
d+2	NULL	

16^n : ASCII code of 16^n position

32-bit unsigned binary data

s	H00000000 ～ HFFFFFFFF

Specify by Double word

⇨

ASCII data in hexadecimal

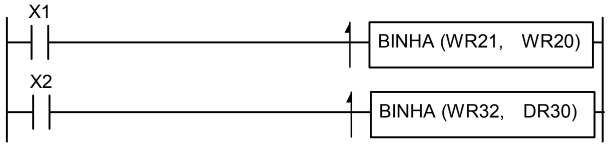
d	16^7	16^6
d+1	16^5	16^4
d+2	16^3	16^2
d+3	16^1	16^0
d+4	NULL	

16^n : ASCII code of 16^n position

Cautionary notes

- The internal output to use for d and s parameters should be specified within the range of I/O number.
- This command changes a size of d parameter by types of s parameter which is a target to convert.
  (If ‘s’ is word, it uses up to d+2. If ‘s’ is double word, it uses up to d+4.)

¥ Program example



[Program description]

- A value (16-bit unsigned binary data) of WR20 is converted to ASCII data in hexadecimal at the rising of X1, and the result is set from WR21 to WR22. (WR23 is set to NULL.)
  If WR20 = H1234, WR21=H3132, WR22=H3334, and WR13=H0000.
- A value (32-bit unsigned binary data) of DR30 is converted to ASCII data in hexadecimal at the rising of S2, and the result is set from WR32 to WR35. (WR36 is set to NULL.)
  If DR30 = H001289AB, WR32=H3030, WR33=H3132, WR34=H3839, WR35=H4142, and WR36=H0000.

PRN ➔ PRJ

This command is equivalent to FUN 32 (s) and FUN 33 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 32 (s) and FUN 33 (s) for EHV, how to convert is as follows.

FUN 32 (s) ➔ BINHA (s+1, s)

Example) FUN 32 (WR100) ➔ BINHA (WR101, WR100)

FUN 33 (s) ➔ BINHA (s+2, s), provided that ‘s’ is double word.

Example) FUN 33 (WR100) ➔ BINHA (WR102, DR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	[Conversion of Character] Conversion BCD ➔ ASCII in decimal

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
BCDDA (d, s)	Word	4	↕	●	●	●	●
	Double word	5					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	2.6	—	—	
Double word	3.8	—	—	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O after conversion									✓	✓	✓	✓					
s	I/O before conversion								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

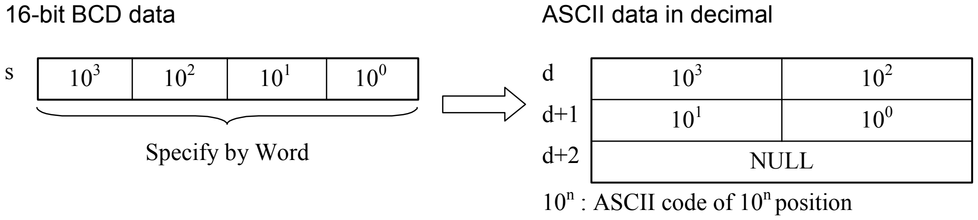
- BCD data is converted to ASCII code in decimal.
  If 's' is Words, 16-bit BCD data is converted to 4-digit ASCII code in decimal.
  If 's' is Double word, 32-bit BCD data is converted to 8-digit ASCII code in decimal.
- A digit which does not have a number as a result of performing zero suppression to the converted result is set to H20 (a space). And NULL behind ASCII data means an end of a character string.

Parameter

d : The start I/O of a table to store ASCII data in decimal after conversion is specified.

s : Case of Word: The internal output (word) which stores 16-bit BCD data to convert or a constant is specified.

Case of Double word: The internal output (double word) which stores 32-bit BCD data to convert or a constant is specified.



16-bit BCD data

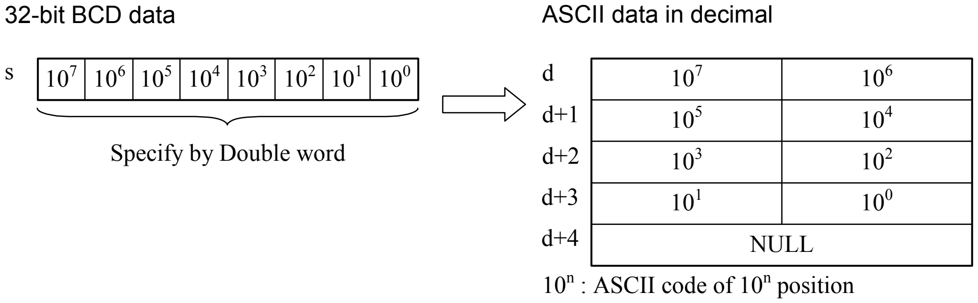
s	10^3	10^2	10^1	10^0

Specify by Word

ASCII data in decimal

d	10^3	10^2
d+1	10^1	10^0
d+2	NULL	

10^n : ASCII code of 10^n position



32-bit BCD data

s	10^7	10^6	10^5	10^4	10^3	10^2	10^1	10^0

Specify by Double word

ASCII data in decimal

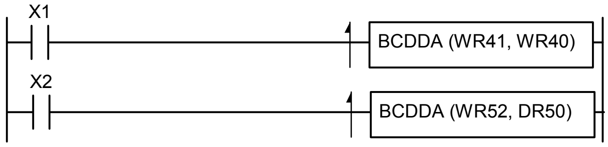
d	10^7	10^6
d+1	10^5	10^4
d+2	10^3	10^2
d+3	10^1	10^0
d+4	NULL	

10^n : ASCII code of 10^n position

Cautionary notes

- The internal output to use for d and s parameters should be specified within the range of I/O number.
- If data to convert specified by s parameter is data other than BCD data (if it is A to F), DER=1 and the operation is not performed.
- This command changes a size of d parameter by types of s parameter which is a target to convert.
  (If 's' is Word, it uses up to d+2. If 's' is Double word, it uses up to d+4.)

Program example



[Program description]

- A value (16-bit BCD data) of WR40 is converted to ASCII data in decimal at the rising of X1, and the result is set into WR41 to WR42. (WR43 is set to NULL.)
  If WR40 = H0123, WR41=H3031, WR42=H3233, and WR43=H0000.
- A value (32-bit BCD data) of DR50 is converted to ASCII data in decimal at the rising of X2, and the result is set into WR52 to WR55. (WR56 is set to NULL.)
  If DR50 = H00120567, WR52=H3030, WR53=H3132, WR54=H3035, WR55=H3637, and WR56=H0000.

PRN ➔ PRJ

This command is equivalent to FUN 34 (s) and FUN 35 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 34 (s) and FUN 35 (s) for EHV, how to convert is as follows.

FUN 34 (s) ➔ BCDDA (s+1, s)

Example) FUN 34 (WR100) ➔ BCDDA (WR101, WR100)

FUN 35 (s) ➔ BCDDA (s+2, s), provided that 's' is Double word.

Example) FUN 35 (WR100) ➔ BCDDA (WR102, DR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	[Conversion of Character] Conversion 5-digit unsigned ASCII in decimal ➔ 16-bit binary

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
DABIN (d, s)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		s can used up to s+2.
Condition	Time	Condition	Time	
–	6.2	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O after conversion									✓	✓	✓	✓					
s	I/O before conversion								✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Function

- 5-digit unsigned ASCII data in decimal is converted to 16-bit binary data.
  H00 and H20 (NULL and space) in upper digits are dealt with as H30 (“0”). (digit for zero suppression)

Parameter

d : The internal output to store 16-bit binary data after conversion is specified.

s : The start I/O of a table which stores unsigned ASCII data in decimal to convert is specified.

Unsigned ASCII data in decimal

	15 8	7 0
s	10^4	10^3
s+1	10^2	10^1
s+2	10^0	H00

⇨

16-bit binary data

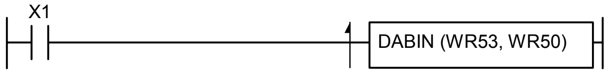
d	0 ～ 65,535

10^n : ASCII code of 10^n position

Cautionary notes

- The internal output to use for d and s parameters should be specified within the range of I/O number.
- If 5-digi ASCII code specified by s parameter is data other than from H30 to H39 (0 to 9), DER=1, and the operation is not performed.
- If the operation result becomes 65,535 or more, DER=1 and the operation is not performed.

Program example



[Program description]

Value (unsigned ASCII data in decimal) of WR50 to WR52 is converted to 16-bit binary data and the result is set into WR53.

If WR50=H3132, WR51=H3334 and WR52=H3500, WR53=12345.

PRN ➔ PRJ

This command is equivalent to FUN 36 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 36 (s) for EHV, how to convert is as follows.

FUN 36 (s) ➔ DABIN (s+3, s)

Example) FUN 36 (WR100) ➔ DABIN (WR103, WR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	[Conversion of Character] Conversion 10-digit signed ASCII in decimal ➔ 32-bit binary

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
SDABIN (d, s)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		s can be used up to s+5.
Condition	Time	Condition	Time	
–	10.1	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O after conversion													✓	✓	✓	
s	I/O before conversion							✓	✓	✓	✓	✓					✓

Function

- 10-digit signed ASCII data in decimal is converted to 32-bit binary data.
- Argument should be combined among H00, H20 (NULL and space), H30 to H39 and H2D (“-“).
- H00 and H20 (NULL and space) in upper digits are dealt with as H30 (“0”). (digit for zero suppression)

Parameter

d : The internal output to store 32-bit binary data after conversion is specified.

s : The start I/O of a table which stores signed ASCII data in decimal to convert or a constant is specified.

Signed ASCII data in decimal

	15 8	7 0
s	符号	10^9
s+1	10^8	10^7
s+2	10^6	10^5
s+3	10^4	10^3
s+4	10^2	10^1
s+5	10^0	H00

⇨

32-bit signed binary data

d	−2,147,483,648 ～ 2,147,483,647

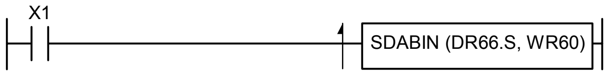
Sign Positive : H20 (space)
Negative : H2D(“−“)

10^n : ASCII code of 10^n position

Cautionary notes

- The internal output to use for d and s parameters should be specified with the range of I/O number.
- If 10-difit ASCII code specified by s parameter is data other than from H30 to H39 (0 to 9) and the sign is data other than H20 and H2D, DER=1 and the operation is not performed, but this is not true of H00 and H20 (NULL and space) in digits with which zero suppression is executed.
- If the operation result becomes value other than from −2,147,483,648 to 2,147,483,647, DER=1 and the operation is not performed.

Program example



[Program description]

Value (signed ASCII data in decimal) of WR60 to WR65 is converted to 32-bit signed binary data at the rising of X1 and the result is set into DR66.S.

If WR60=H2D32, WR61=H3134, WR62=H3734, WR63=H3833, WR64=H3634 and WR65=H3800, DR66.S=−21474383648.

PRN ➔ PRJ

This command is equivalent to FUN 37 (s) in the program (PRN file) of the EH-CPU.

When changing the program which has used FUN 37 (s) for EHV, how to convert is as follows.

FUN 37 (s) ➔ SDABIN ([s+6].S, s), provided that ‘s+6’ is double word.

Example) FUN 37 (WR100) ➔ SDABIN (DR106.S, WR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	[Conversion of Character] Conversion ASCII in hexadecimal ➔ 16-bit binary

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
HABIN (d, s)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	5.2	–	–	
Double word	7.6	–	–	

Usable I / O		Bit							Word1					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O after conversion									✓	✓	✓	✓		✓	✓	✓	
s	I/O before conversion								✓	✓	✓	✓	✓					✓

Function

- ASCII data in hexadecimal is converted to binary data.
  If 'd' is Word, 4-digit ASCII data in hexadecimal is converted to 16-bit binary data.
  If 'd' is Double word, 8-digit ASCII data in hexadecimal is converted to 32-bit binary data.
- H00and H20 (NULL and space) in upper digits are dealt with as H30 ("0"). (Digit for zero suppression)

Parameter

d : The internal output (word) to store 16-bit binary or the internal output (double word) to store 32-bit binary data after conversion is specified.

s : The start I/O of a table which stores ASCII data in hexadecimal to convert is specified.

ASCII data in hexadecimal

	15 8	7 0
s	16^3	16^2
s+1	16^1	16^0

16^n : ASCII code of 16^n position

⇨

16 ビットバイナリデータ

d H0000 ～ HFFFF

ASCII data in hexadecimal

	15 8	7 0
s	16^7	16^6
s+1	16^5	16^4
s+2	16^3	16^2
s+3	16^1	16^0

16^n : ASCII code of 16^n position

⇨

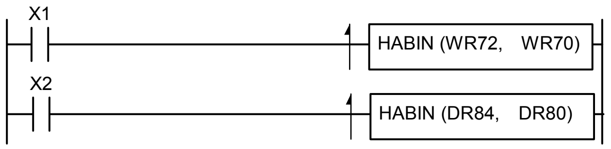
32-bit binary data

d H00000000 ～ HFFFFFFFF

Cautionary notes

- The internal output to use d and s parameters should be specified within the range of I/O number.
- If ASCII code specified by s parameter is data other than from H30 to H39 (0 to 9) and from H41 to H46 (A to F), DER=1 and the operation is not performed. But this is not true of H00 and H20 (NULL and space) in digits with which zero suppression is performed.
- This command changes a size of s parameter by types of d parameter to store the operation result. (If 'd' is Word, ASCII code up to s+1 is converted. If 'd' is Double word, ASCII code up to s+3 is converted.)

Program example



[Program description]

- Value (ASCII data in hexadecimal) of WR70 and WR71 is converted to 16-bit binary data at the rising of X1 and the result is set into WR72.

  If WR70 = H3132 and WR71=H4142, WR72=H12AB.
- Value (ASCII data in hexadecimal) of WR80 to WR83 is converted to 16-bit binary data at the rising of X2 and the result is set into DR84.

  If WR80=H4645, WR81=H4443, WR82=H4241 and WR83=H3938, DR80 is set to HFEDCBA98.

PRN ➔ PRJ

This command is equivalent to Fun 38 (s) and FUN 39 (s) in the program (PRN file) of EH-CPU.

When changing the program whish has used FUN 38 (s) and FUN 39 (s) for EHV, how to convert is as follows.

FUN 38 (s) ➔ HABIN (s+2, s)

Example) FUN 38 (WR100) ➔ HABIN (WR102, WR100)

FUN 39 (s) ➔ HABIN (s+4, s), provided that 's+4' is double word.

Example) FUN 39 (WR100) ➔ HABIN (DR104, WR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	[Conversion of Character] Conversion ASCII in decimal ➔ 16-bit BCD

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
DABCD (d, s)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	4.7	–	–	
Double word	6.6	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O after conversion									✓	✓	✓	✓		✓	✓	✓	
s	I/O before conversion								✓	✓	✓	✓	✓					✓

Function

- ASCII data in decimal is converted to 16-bit BCD data.
  If ‘d’ is Word, 4-digit ASCII data in decimal is converted to 16-bit BD data.
  If ‘d’ is Double word, 8-digit ASCII data in decimal is converted to 32-bit BBD data.
- H00 and H20 (NULL and space) in upper digits are dealt with as H30 (“0”). (Digit for zero suppression)

Parameter

d : The internal output (word) to store 16-bit BCD data or the internal output (double word) to store 32-bit BCD data after conversion is specified.

s : The start I/O of a table which stores 4-digit ASCII data in decimal to convert is specified.

ASCII data in decimal

	15 8	7 0
s	10^3	10^2
s+1	10^1	10^0

10^n : ASCII code of 10^n position

⇨

16 ビット BCD データ

	15	8 7		0
d	10^3	10^2	10^1	10^0

10^m : 10^m の位の BCD コード

ASCII data in decimal

	15 8	7 0
s	10^7	10^6
s+1	10^5	10^4
s+2	10^3	10^2
s+3	10^1	10^0

10^n : ASCII code of 10^n position

⇨

32-bit BCD data

	31	24 23		16 15		8 7		0
d	10^7	10^6	10^5	10^4	10^3	10^2	10^1	10^0

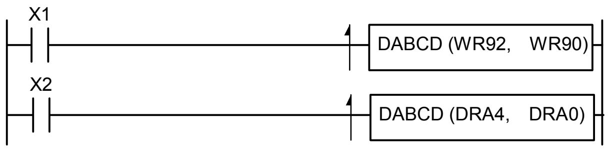
Specify by Double word

10^m : BCD code of 10^m position

Cautionary notes

- The internal output to use for d and s parameters should be specified within the range of I/O number.
- If ASCII code specified by s parameter is data other than from H30 to H39 (0 to 9), DER=1 and the operation is not performed.
- This command changes a size of s parameter by types of d parameter to store the operation result.
  (If ‘d’ is Word, ASCII code up to s+1 is converted. If ‘d’ is Double word, ASCII code up to s+3 is converted.)

Program example



[Program description]

- Value (ASCII data in decimal) of WR90 and WR91 is converted to 16-bit BCD data at the rising of X1and the result is set into WR92.
  If WR90 = H2020 and WR91=H3031, WR92=H0001.
- Value (ASCII data in decimal) of from WRA0 to WRA3 is converted to 16-bit BCD data at the rising of X2 and the result is set into DRA4.
  If WRA0=H3938, WRA1=H3736, WRA2=H3534 and WRA3=H3332, DRA4 is set to H98765432.

PRN ➔ PRJ

This command is equivalent to FUN 40 (s) and FUN 41 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 40 (s) and FUN 41 (s) for EHV, how to convert is as follows.

FUN 40 (s) ➔ DABCD (s+2, s)

Example) FUN 40 (WR100) ➔ DABCD (WR102, WR100)

FUN 41 (s) ➔ DABCD (s+4, s), provided that ‘s+4’ is Double word.

Example) FUN 41 (WR100) ➔ DABCD (DR104, WR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	[Conversion of Character] Conversion Binary character string ➔ ASCII character string in hexadecimal

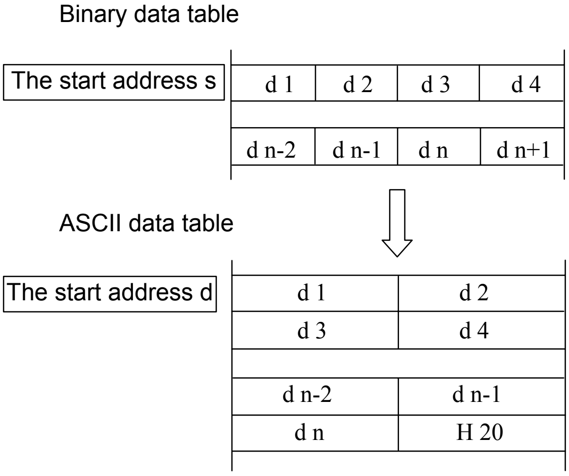
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
ASC (d, s, n)	—	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	7.69+0.31n	—	—	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	The start I/O of ASCII data table										✓	✓					
s	The start I/O of binary data table							✓	✓	✓	✓	✓					
n	Number of conversions							✓	✓	✓	✓	✓					✓

Function

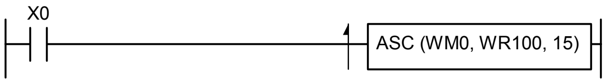
The number of characters specified by ‘n’ from the start in the binary data table specified by ‘s’ is converted to ASCII code in hexadecimal and converted data is stored in sequence from the internal output specified by ‘d’.



Cautionary notes

- The internal output used for binary data table and ASCII data table should be specified within the range of I/O No. And if areas of binary data table and ASCII data table are overlapping, the operation is not performed because of DER=1.
- The converted ASCII data are stored in sequence from the start in the word-unit. If the number of characters to convert is odd numbers, an end of the table stores H20 (space).

Program example



[Program description]

15 characters from WR100 of binary data in hexadecimal are converted to ASCII data in hexadecimal at the rising of X0. The converted result is stored in sequence from WM0.

WR100	H1 2	3 4
WR101	H5 6	7 8
WR102	H9 A	B C
WR103	HD E	F 0

⇨

WM0	H3 1	3 2
WM1	H3 3	3 4
WM2	H3 5	3 6
WM3	H3 7	3 8
WM4	H3 9	4 1
WM5	H4 2	4 3
WM6	H4 4	4 5
WM7	H4 6	2 0

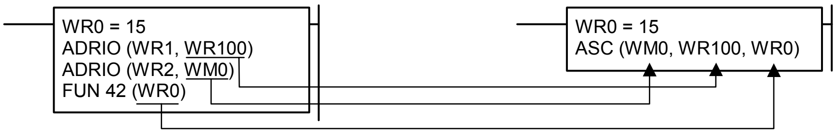
PRN ➔ PRJ

This command is equivalent to FUN 42 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 42 (s) for EHV, how to convert is as follows.

FUN 42 (s) ➔ ASC ([I/O specified by s+2], [I/O specified by s+1], s)

Program for EH-CPU Program for EHV-CPU



```
WR0 = 15
ADRIO (WR1, WR100)
ADRIO (WR2, WM0)
FUN 42 (WR0)
```

```
WR0 = 15
ASC (WM0, WR100, WR0)
```

* This command is not convertible by a conversion tool. Please convert as mentioned above by users, yourselves.

Name	[Conversion of character] Conversion ASCII character string in hexadecimal ➔ Binary character string

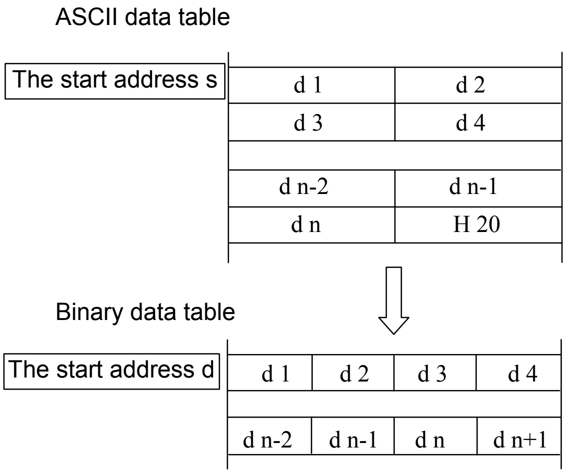
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
HEX (d, s, n)	–	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	8.4+0.6n	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	The start I/O in binary data table											✓	✓					
s	The start I/O in ASCII data table								✓	✓	✓	✓	✓					
n	Number of conversions								✓	✓	✓	✓	✓					✓

Function

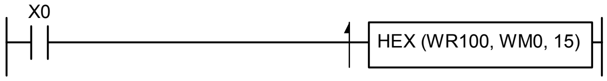
The number of characters specified by ‘n’ from the start in the ASCII code table in hexadecimal specified by ‘s’ is converted to binary data and converted data is stored in sequence from the internal output specified by ‘d’.



Cautionary notes

- The internal output used for ASCII data table and binary data table should be specified within the range of I/O No. And if areas of ASCII data table and binary data table are overlapping, the operation is not performed because of DER=1.
- Converted binary data is stored in sequence from the start in word units. If the number of characters to convert is not a multiple of 4, a data part less than 1 word stores 0.

Program example



[Program description]

ASCII data in hexadecimal for 15 characters from WM0 is converted to binary data in hexadecimal at the rising of X0 and the converted data is stored in sequence from WR100.

WM0	H3 1	3 2
WM1	H3 3	3 4
WM2	H3 5	3 6
WM3	H3 7	3 8
WM4	H3 9	4 1
WM5	H4 2	4 3
WM6	H4 4	4 5
WM7	H4 6	2 0

⇨

WR100	H1 2	3 4
WR101	H5 6	7 8
WR102	H9 A	B C
WR103	HD E	F 0

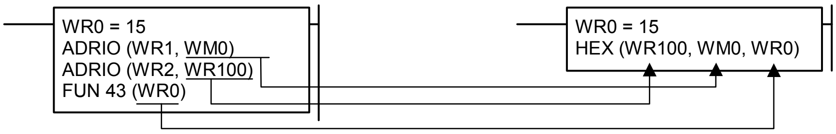
PRN ➔ PRJ

This command is equivalent to FUN 43 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 43 (s) for EHV, how to convert is as follows.

FUN 43 (s) ➔ HEX ([I/O specified by s+2], [I/O specified by s+1], s)

Program for EH-CPU Program for EHV-CPU



```
WR0 = 15
ADRIO (WR1, WM0)
ADRIO (WR2, WR100)
FUN 43 (WR0)
```

```
WR0 = 15
HEX (WR100, WM0, WR0)
```

* This command is not convertible by a conversion tool. Please convert as mentioned above by users, yourselves.

Name	[Data operation] Conversion Word units ➔ Byte units

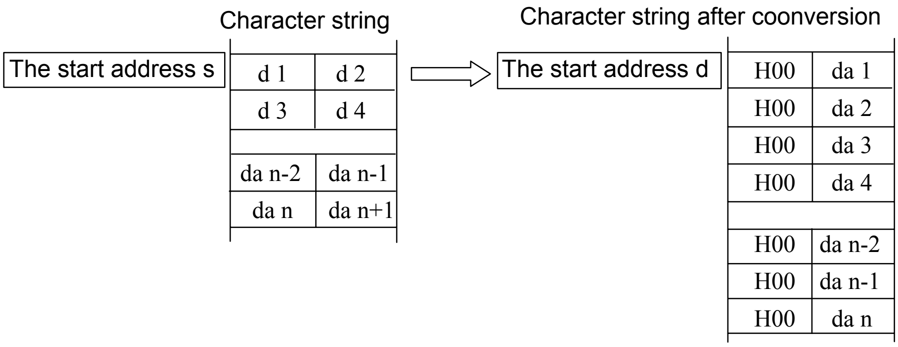
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
WTOB (d, s, n)	–	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	6.77+0.23n	–	–	

Usable / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	The start I/O of character string after conversion										✓						
s	The start I/O of character string before conversion										✓						
n	Number of conversion bytes							✓	✓	✓	✓	✓					✓

Function

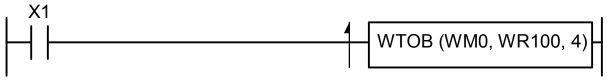
n-byte data is picked out from the character data of which the start is address s, and the data picked out is stored in sequence from I/O specified by d as 1 byte per 1 word.



Cautionary notes

- The internal output used for the character string and the character string after conversion should be specified within the range of I/O No.
- If areas of the character string and the character string after conversion are overlapping, the operation is not performed because of DER=1.
- If n＝0, it is not converted and DER=0.

Program example



[Program description]

Data for 4 bytes from WR100 is picked out on order of the upper byte then the lower byte at the rising of X1, and data picked out is set in sequence from WM0.

WR100	H1 2	3 4
WR101	H5 6	7 8

⇨

WM0	H0 0	1 2
WM1	H0 0	3 4
WM2	H0 0	5 6
WM3	H0 0	7 8

PRN ➔ PRJ

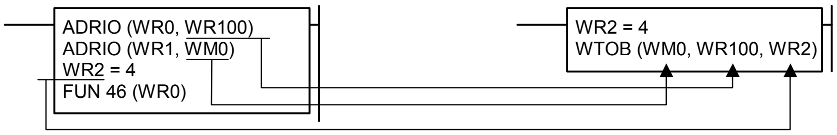
This command is equivalent to FUN 46 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 46 (s) for EHV, how to convert is as follows.

FUN 46 (s) ➔ WTOB ([I/O specified by s+1], [I/O specified by s], s+2)

Program for EH-CPU

Program for EHV-CPU



```
ADRIO (WR0, WR100)
ADRIO (WR1, WM0)
WR2 = 4
FUN 46 (WR0)
```

```
WR2 = 4
WTOB (WM0, WR100, WR2)
```

* This command is not convertible by a conversion tool. Pleas convert as mentioned above by users, yourselves.

Name	[Data operation] Conversion Byte units ➔ Word units

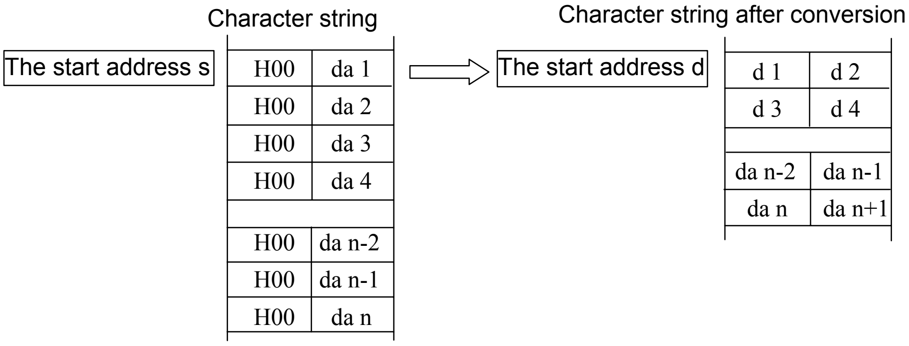
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
BTOW (d, s, n)	–	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	7.79+0.21n	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	The start I/O of character string after conversion										✓						
s	The start I/O of character string before conversion										✓						
n	Number of conversion bytes							✓	✓	✓	✓	✓					✓

Function

The lower byte (n bytes) are picked out from the character string data of which the start is address s, and the bytes picked out are stored in sequence from I/O specified by d as 2 bytes per 1 word.



Character string

The start address s	
H00	da 1
H00	da 2
H00	da 3
H00	da 4
H00	da n-2
H00	da n-1
H00	da n

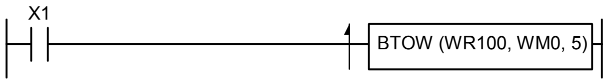
Character string after conversion

The start address d	
d 1	d 2
d 3	d 4
da n-2	da n-1
da n	da n+1

Cautionary notes

- The internal output used for the character string and the character string after conversion should be specified within the range of I/O No.
- If area of the character string and the character string after conversion are overlapping, the operation is not performed because of DER=1.
- If n＝0, it is not converted and DER=0.
- If the number of conversion bytes is odd number, the lower 8 bits at the end of the place for output is H00.

Program example



[Program description]

The lower bytes for 5 words from WM0 are picked out at the rising of X1, and the bytes picked out are set in order of the upper byte, a next lower byte, from WR100.

WM0	Any	H1 2
WM1	Any	H3 4
WM2	Any	H5 6
WM3	Any	H7 8
WM4	Any	H9 A

⇨

WR100	H1 2	3 4
WR101	H5 6	7 8
WR102	H9 A	0 0

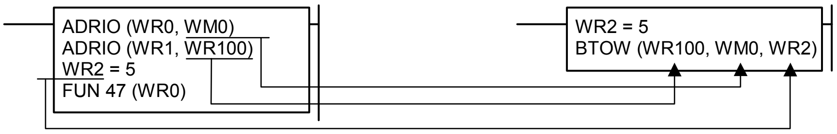
PRN ➔ PRJ

This command is equivalent to FUN 47 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 47 (s) for EHV, how to convert is as follows.

FUN 47 (s) ➔ BTOW ([I/O specified by s+1], [I/O specified by s], s+2)

Program for EH-CPU Program for EHV-CPU



```
ADRIO (WR0, WM0)
ADRIO (WR1, WR100)
WR2 = 5
FUN 47 (WR0)
```

```
WR2 = 5
BTOW (WR100, WM0, WR2)
```

* This command is not convertible by a conversion tool. Please convert as mentioned above by users, yourselves.

Name	[Data operation] Reverse

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
NOT (d, s)	Bit	4	●	●	●	●	●
	Word	4					
	Double word	5					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Bit	0.22	—	—	
Word	0.24	—	—	
Double word	0.34	—	—	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O after reverse		✓		✓				✓	✓	✓			✓		✓	
s	I/O to reverse	✓	✓		✓			✓	✓	✓	✓	✓	✓	✓		✓	

Function

- The contents of s are reversed to store in d.
- The combinations of d and s are as follows.

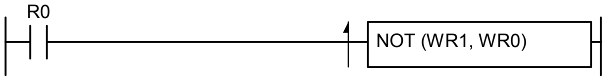
d	s
Bit	Bit
Word	Word
Double word	Double word

1	1	1	1	0	0	0	0	1	1	0	0	1	0	1	0
⇩	⇩	⇩	⇩	⇩	⇩	⇩	⇩	⇩	⇩	⇩	⇩	⇩	⇩	⇩	⇩
0	0	0	0	1	1	1	1	0	0	1	1	0	1	0	1

Cautionary notes

Please set a startup condition of this command to the edge trigger.

Program example



[Program description]

The contents of WR0 are reversed at the rising of R0 and the result is stored in WR1.

Ex.) If WR0 is H1234, WR1=HEDCB after execution of the command.

PRN → PRJ

This command is equivalent to NOT (d) in the program (PRN file) of EH-CPU.

When changing the program which has used NOT (d) for EHV, how to convert is as follows.

NOT (d) → NOT (d, s) Both d and s are the same I/O.

* If converted by a conversion tool, it is converted as mentioned above.

Name	[Data operation] Combination

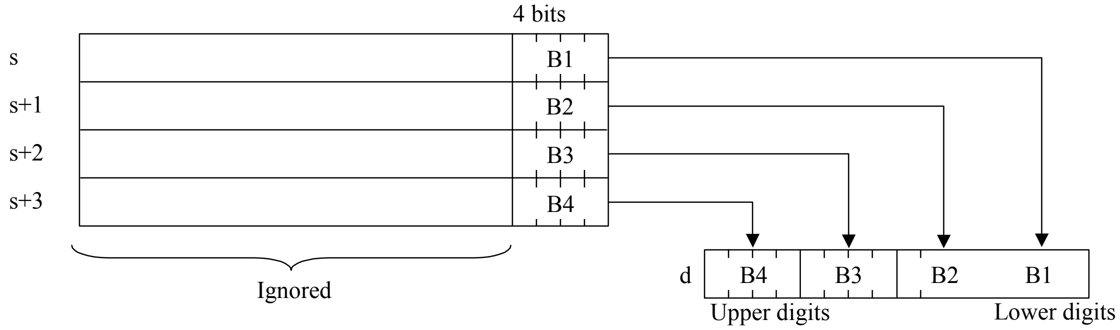
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
UNIT (d, s, n)	―	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		n is from 0 to 4.
Condition	Time	Condition	Time	
―	2.8	―	―	

Usable I / O		Bit						Word					Double word				Constant
		K	/	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN .m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O for writing in the united result								✓	✓	✓						
s	The start I/O to unite							✓	✓	✓	✓						
n	Number of words to unite																✓

Function

- The value of the lower 4 bits in n (1 to 4) words from s is set to each4 bits from the lower in d.
- If n is from 1 to 3, the bit not to be set to d is 0.
- As for data from s to s+n-1, value does not change even if this command is executed.

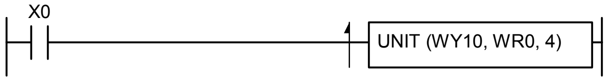


n=0 B4 to B1are 0.
n=1 B4 to B2 are 0.
n=2 B4 to B3 are 0.
n=3 B4 is 0.

Cautionary notes

- s+n−1 should be used within the I/O range.
- If n=0, I/O for writing in is set to 0 because of DER=0.
- If n≧5, it is not executed.

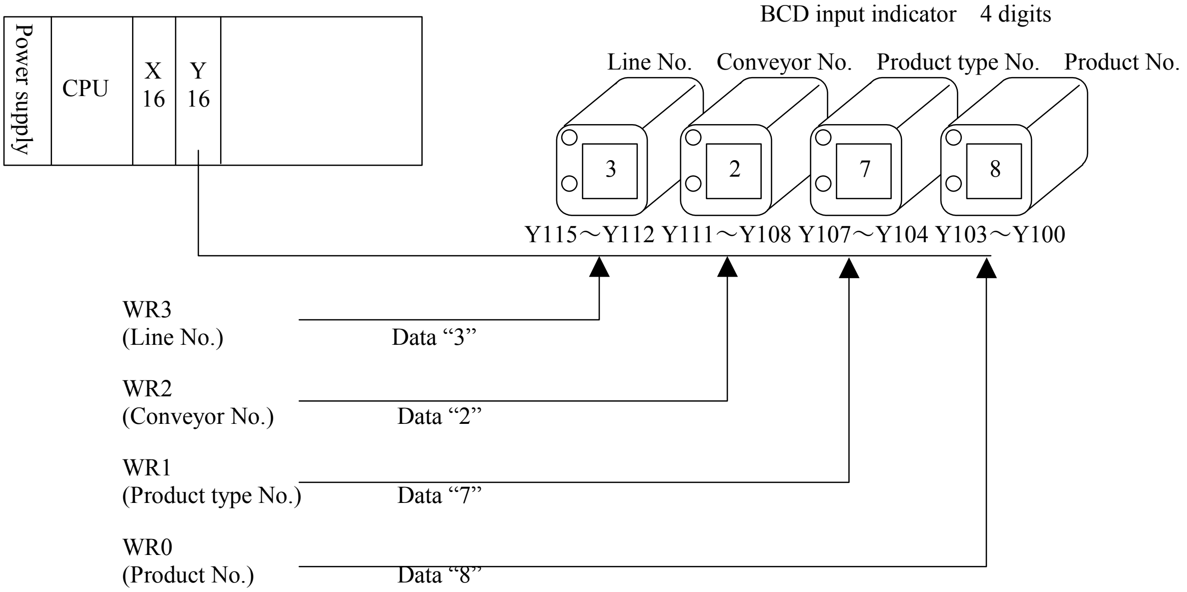
Program example



[Program description]

The 4-digit BCD input indicator is connected to WY10, and individual data from WR0 to WR3 is displayed to each digit.

(As for WR0 to WR3, only data in the lower 4 bits are valid.)



Name	[Data operation] Distribution

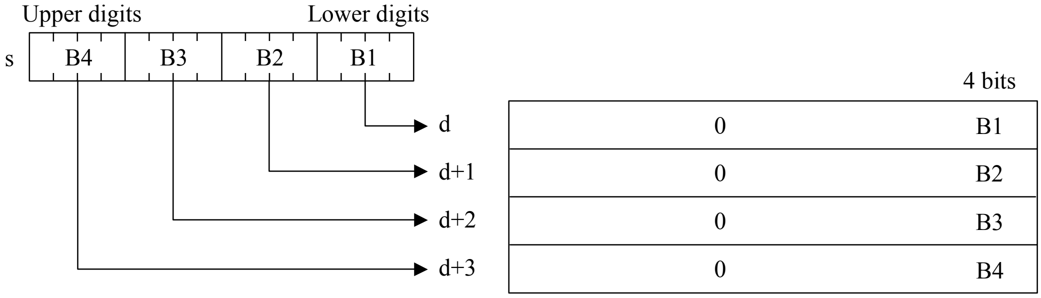
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
DIST (d, s, n)	–	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		n is from 0 to 4.
Condition	Time	Condition	Time	
–	2.6	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O for writing in the distributed result								✓	✓	✓						
s	The start I/O to distribute							✓	✓	✓	✓						
n	Number of words to distribute																✓

Function

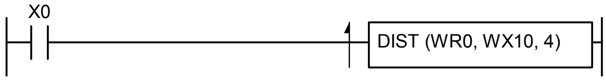
- s is distributed into each 4 bits, and the distributed value is set to the lower 4 bits in n words in sequence from d.
- The each upper 12 bits from d to d+n-1 is set to 0.
- The value of s does not change even if this command is executed.



Cautionary notes

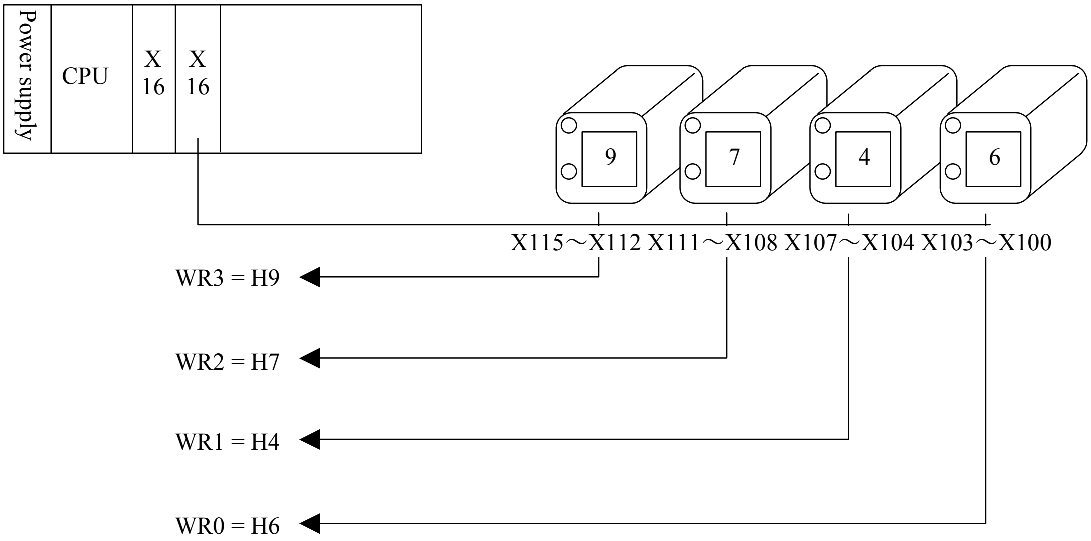
- d+n−1 should be used within the I/O range.
- If n=0, I/O for writing in is set to 0 because of DER=0.
- If n≧5, it is not executed.

Program example



[Program description]

The input of the 4-bit 4-digit Digit switch is connected to WX10 and data in each digit is stored in from WR0 to WR3 as an independent data.



Name	[Data operation] Combination of character data

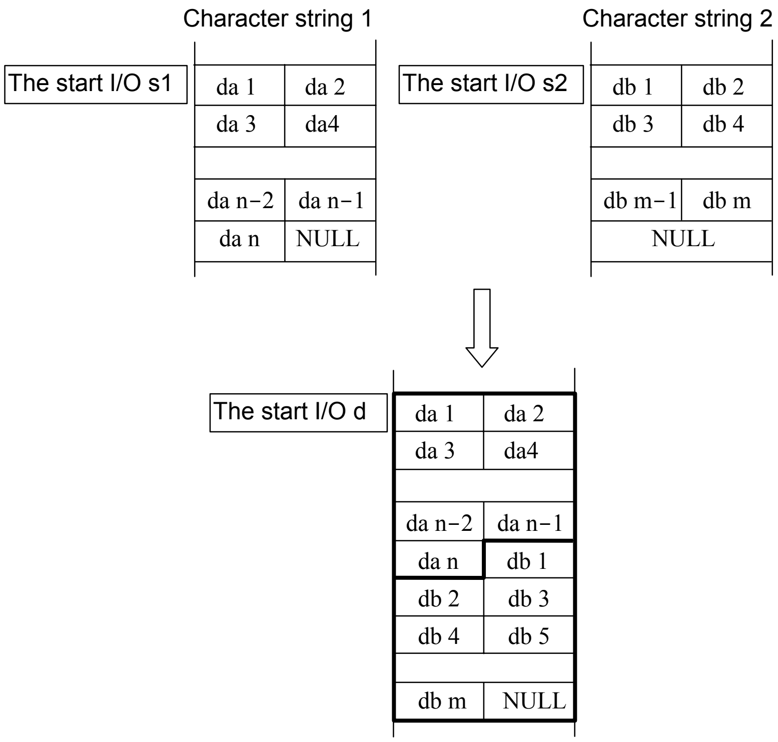
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
SADD (d, s1, s2)	—	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	13.35+0.65n	—	—	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	The start I/O for writing in character string after uniting											✓						
s1	Character string 1 The start I/O											✓						
s2	Character string 2 The start I/O											✓						

Function

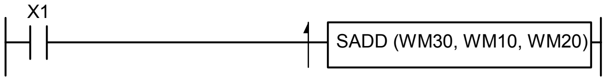
Two different tables are united to make one table.



Cautionary notes

- The internal outputs used for character string 1 and 2 should be specified within the range of I/O No.
- If areas of character string 1 and 2 are overlapping, the operation is not performed because of DER=1.
- NULL (H00) judges the end of data in character string 1 and 2 both. And the rear of character string after uniting is set to NULL.

Program example



[Program description]

At the rising of X1, data between WM10 and NULL(H00) and data between WM20 and NULL are united to be set into and after WM30.

WM10	H4 8	4 9
WM11	H5 4	4 1
WM12	H4 3	4 8
WM13	H4 9	0 0

WM20	H4 E	4 8
WM21	H5 3	4 E
WM22	H5 2	4 9
WM23	H4 E	5 3
WM24	H0 0	0 0

⇨

WM30	H4 8	4 9
WM31	H5 4	4 1
WM32	H4 3	4 8
WM33	H4 9	4 E
WM34	H4 8	5 3
WM35	H4 E	5 2
WM36	H4 9	4 E
WM37	H5 3	0 0

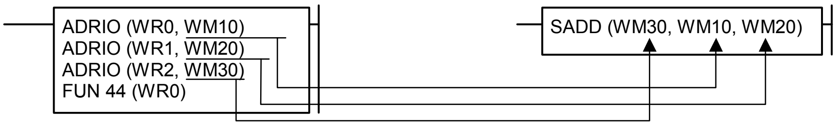
PRN ➔ PRJ

This command is equivalent to FUN 44 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 44 (s) for EHV, how to convert is as follows.

FUN 44 (s) ➔ SADD ([I/O specified by s+2], [I/O specified by s], [I/O specified by s+1])

Program for EH-CPU Program for EHV-CPU



```
ADRIO (WR0, WM10)
ADRIO (WR1, WM20)
ADRIO (WR2, WM30)
FUN 44 (WR0)
```

```
SADD (WM30, WM10, WM20)
```

* This command is not convertible by a conversion tool. Please convert as mentioned above by users, yourselves.

Name	[Data operation] Comparison of character data

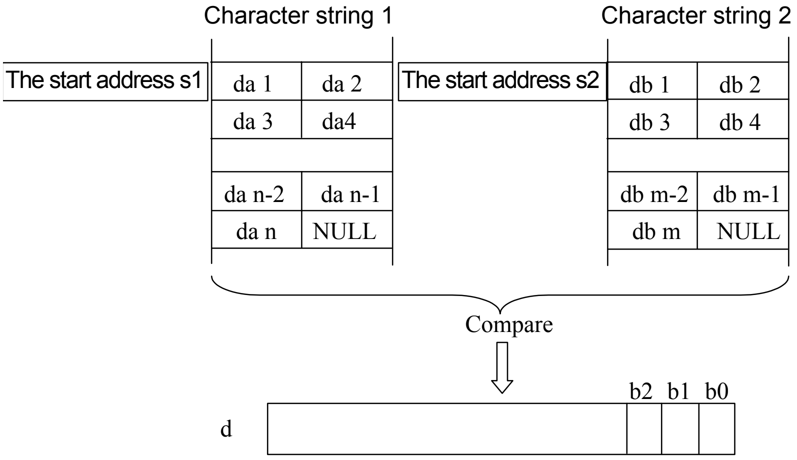
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
SCMP (d, s1, s2)	—	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	9.43+0.57n	—	—	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	The start I/O for writing in character string after uniting											✓						
s1	Character string 1 The start I/O								✓	✓	✓	✓						
s2	Character string 2 The start I/O								✓	✓	✓	✓						

Function

Each data from the start to NULL(H00) in two different tables of which the start is the specified I/O are collated. The number of characters is compared first, then the character string is compared. If the number of characters is not matched, the character string is not compared because the comparison is terminated.

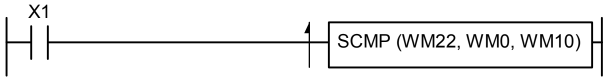


	b2	b1	b0
Number of character is not matched	1	0	0
Character string is not matched	0	1	0
Character string is matched	0	0	1

Cautionary notes

- The internal outputs used for the character string 1 and 2 should be specified within the range of I/O No.
- If areas of the character string 1 and 2 are overlapping, the operation is not performed because of DER=1.

Program example



```
SCMP (WM22, WM0, WM10)
```

[Program description]

Data in and after WM0 and data in and after WM10 are compared and the result is set to WM22 at the rising of X1.

PRN ➔ PRJ

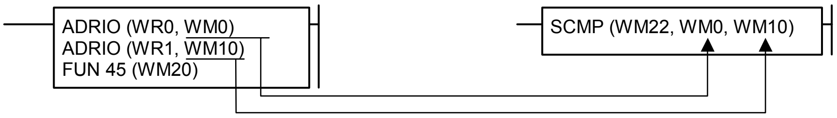
This command is equivalent to FUN 45 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 45 (s) for EHV, how to convert is as follows.

FUN 45 (s) ➔ SCMP (s+2, [I/O specified by s], [I/O specified by s+1])

Program for EH-CPU

Program for EHV-CPU



```
ADRIO (WR0, WM0)
ADRIO (WR1, WM10)
FUN 45 (WM20)
```

```
SCMP (WM22, WM0, WM10)
```

* This command is not convertible by a conversion tool. Please convert as mentioned above by users, yourselves.

Name	[Data operation] Conversion Bit units ➔ Word unites

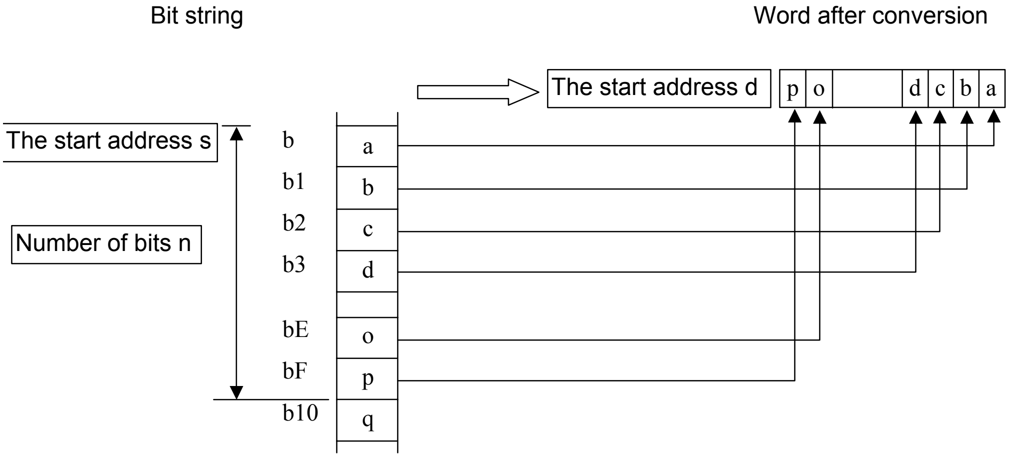
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
BITTOW (d, s, n)	—	5	↕	●	●	●	●

Command processing time (µs)				Remarks
Average		Maximum		n is from 0 to 16.
Condition	Time	Condition	Time	
—	8.92+2.08n	—	—	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Word I/O to store bit data										✓	✓						
s	The start bit I/O to store in word data	✓	✓	✓	✓													
n	Number of conversion bits								✓	✓	✓	✓	✓					✓

Function

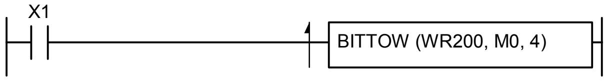
n bits from bit I/O specified by s is set in sequence from the lower bit into word I/O specified by d. If the number of bits specified is less than 16 (if the number of bits specified is from 1 to 15), the upper bits in word I/O are set to 0.



Cautionary notes

- The internal outputs used for the bit string and the word after conversion should be specified within the range of I/O No. If the bit I/O address exceeds the maximum of I/O No., data is developed within the range of I/O's specification, but DER=1.
- If areas of the bit string and the word after conversion are overlapping, the operation is not performed because of DER=1.
- If the number of bits exceeds 16, it is not processed because of DER=1.
- If the number of bits is 0, it is not processed, and DER=0.

Program example



[Program description]

Bit data from M0 to M3 (4 bits) is set in sequence from the lower bit of WR200 and other bits are set to 0 at the rising of X1.

PRN ➔ PRJ

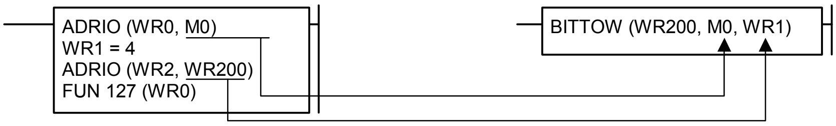
This command is equivalent to FUN 127 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 127 (s) for EHV, how to convert is as follows.

FUN 127 (s) ➔ BITTOW ([I/O specified by s+2], [I/O specified by s], s+1)

Program for EH-CPU

Program for EHV-CPU



* This command is not convertible by a conversion tool. Please convert as mentioned above by users, yourselves.

Name	[Data operation] Conversion Word units ➔ Bit unites

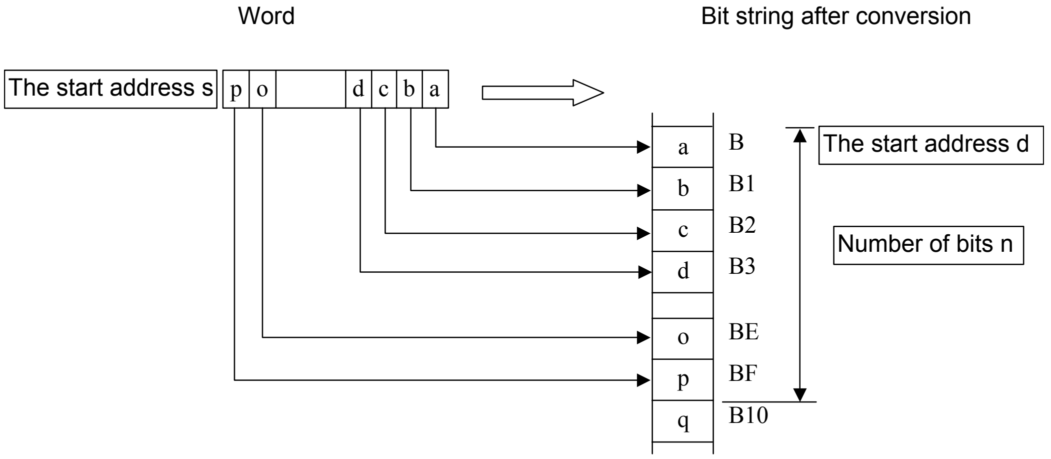
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
WTOBIT (d, s, n)	—	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		n is from 0 to 16.
Condition	Time	Condition	Time	
—	9.01+2.29n	—	—	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	The start I/O in bit string after conversion	✓	✓	✓	✓													
s	Word I/O developed to bit data											✓						
n	Number of conversion bits								✓	✓	✓	✓	✓					✓

Function

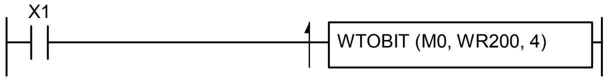
n bits from the 0th bit in word I/O specified by s are developed, of which the start is bit I/O specified by d.



Cautionary notes

- The internal outputs used for word data and the bit string after conversion should be specified within the range of I/O No. If the bit I/O address exceeds the maximum of I/O No., data is developed within the range of I/O's specification, but DER=1.
- If areas of word data and the bit string after conversion are overlapping, the operation is not performed because of DER=1.
- If the number of bits exceeds 16, it is not processed because of DER=1.
- If the number of bits is 0, it is not processed, and DER=0.

Program example



[Program description]

The lower 4 bits of WR200 are picked out to set from M0 to M3 at the rising of X1. (The least significant bit of WR200 is stored in M0.)

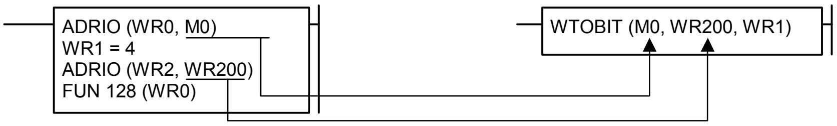
PRN ➔ PRJ

This command is equivalent to FUN 128 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 128 (s), how to convert is as follows.

FUN 129 (s) ➔ WTOBIT ([I/O specified by s], [I/O specified by s+2], s+1)

Program for EH-CPU Program for EHV-CPU



* This command is not convertible by a conversion tool. Please convert as mentioned above by users, yourselves.

Name	[Data operation] Linear interpolation

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
INTPL (s)	–	3	↕	●	●	●	●

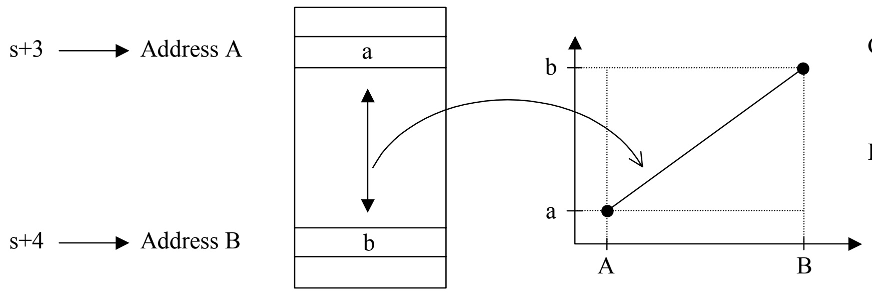
Command processing time (µs)				Remarks
Average		Maximum		s can be used up to s+4.
Condition	Time	Condition	Time	
–	78.8+0.2n	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	The top I/O in table for operation											✓						✓

Function

- The stored value is extracted from I/O of address specified by s+3 and s+4, and the linear expression is found from two extracted values.
  Data between the address specified by s+3 and the address by s+4 is computed in the found linear expression, and the result is stored.
- If s+3 > s+4, data which is stored in the address specified by s+3 is stored in from s+3+1 to the end of the table.
  And data which is stored in the address specified by s+4 is stored in from the beginning of the table to s+4-1.
- If s+3 = 0 and s+4 > 0, data which is stored in the address specified by s+4 is stored in from the beginning of the table to s+4-1.

Ordinary

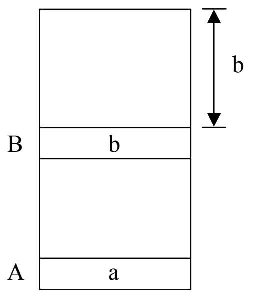


Gradient $\frac{b-a}{B-A}$

Intercept $\frac{Ab-Ba}{B-A}$

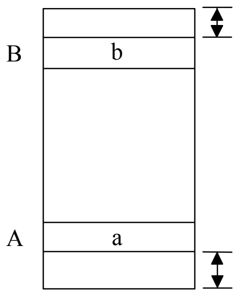
Each cell from the address A+1 to B-1 is calculated from gradient and intercept to put the value on.

Case: a=number of factors in table



Puts data "b" in from the beginning of the table to the address B-1.

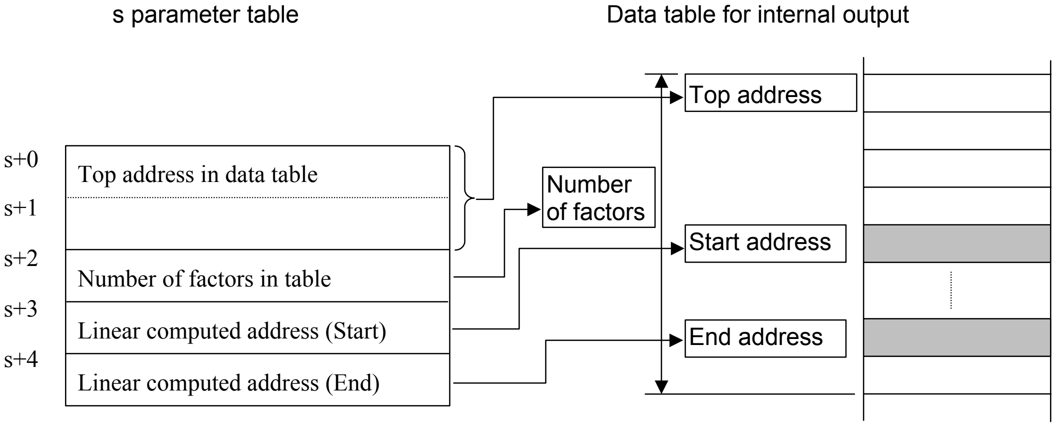
Case: Address A>B



Puts data "b" in from the beginning of the table to the address B-1.
Puts data "a" in from the address A+1 to the end of the table.

Parameter

5 words counting from the beginning is the word address specified by s are used.

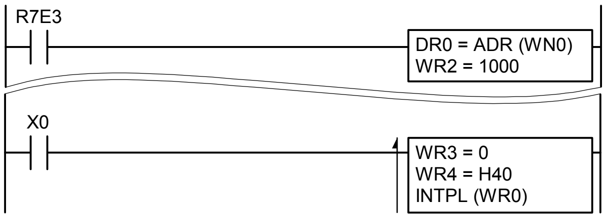

s parameter table
Data table for internal output
s+0
s+1
s+2
s+3
s+4
Top address in data table
Number of factors in table
Linear computed address (Start)
Linear computed address (End)
Number of factors
Top address
Start address
End address

Parameter	Description	Details
s, s+1	The top address in data table	Specify the top address in a table in which data to interpolate to a linear expression is stored. * I/O address coding command is used for specifying the address.
s+2	Number of factors in table	Specify all number of factors (word) in data table.
s+3	Start address	Specify the top address in a table interpolated in a linear expression. (Please specify the offset from the top address specified by s and s+1.)
s+4	End address	Specify the end address in a table interpolated in a linear expression. (Please specify the offset from the top address specified by s and s+1.)

Cautionary notes

- The address to which the number of factors of a table counting from the top address in the table is added should not exceed the I/O range.
- The start address (s+3) of the table should not exceed the number of factors of the table.
- The end address (s+4) of the table should not exceed the number of factors of the table.

Program example



```
DR0 = ADR (WN0)
WR2 = 1000
```

```
WR3 = 0
WR4 = H40
INTPL (WR0)
```

[Program description]

The values from WN1 to WN3F are computed according to a liner expression based on the values of WN0 and WN40 at the rising of X0, and the result is stored.

Name	[Data operation] Increment

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
INC (d)	–	3	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	0.24	–	–	
Double word	0.3	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Incremental I/O									✓	✓	✓	✓		✓	✓	✓	

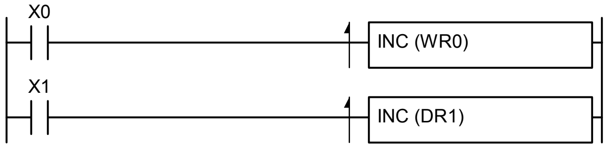
Function

The value of the internal output specified by d parameter increases by 1 whenever the command is executed.

Cautionary notes

- If the internal output specified to d parameter is HFFFF in word, it becomes H0 by adding 1.
- If the internal output specified to d parameter is HFFFFFFFF in double word, it becomes H0 by adding 1.

Program example



[Program description]

- 1 is added to WR at the rising of X0.
- 1 is added to DR1 at the rising of X1.

PRN → PRJ

This command is equivalent to FUN 123 (s) / FUN 124 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 123 (s) / FUN 124 (s) for EHV, how to convert is as follows.

FUN 123 (s) → INC (s)

Example) FUN 123 (WR100) → INC (WR100)

FUN 124 (s) → INC (s), provided that s is double word.

Example) FUN 124 (WR100) → INC (DR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	[Data operation] Decrement

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
DEC (d)	–	3	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	0.24	–	–	
Double word	0.3	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Incremental I/O									✓	✓	✓	✓		✓	✓	✓	

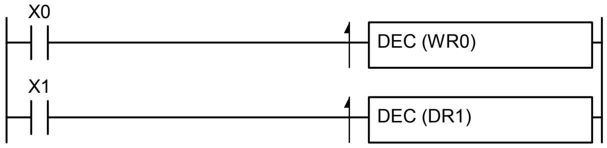
Function

The value of the internal output specified to d parameter decreases by 1 whenever the command is executed.

Cautionary notes

- If the internal output specified to d parameter is H0 in word, it becomes HFFFF by subtracting 1.
- If the internal output specified to d parameter is H0 in double word, it becomes HFFFFFFFF by subtracting 1.

Program example



[Program description]

- 1 is subtracted from WR at the rising of X0.
- 1 is subtracted from DR1 at the rising of X1.

PRN ➔ PRJ

This command is equivalent to FUN 125 (s) / FUN 126 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 125 (s) / FUN 126 (s) for EHV, how to convert is as follows.

FUN 125 (s) ➔ DEC (s)

Example) FUN 125 (WR100) ➔ DEC (WR100)

FUN 126 (s) ➔ DEC (s), provided that s is double word.

Example) FUN 126 (WR100) ➔ DEC (DR100)

* If converted by a conversion tool, it is converted as mentioned above.

Name	[Data search] Search for word data

Ladder format	Number of steps		Condition code				
	Conversion	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
DSRCH (d, s1, s2, n)	—	6	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		d can be use up to d+1.
Condition	Time	Condition	Time	
—	7.6+0.3n	—	—	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Search result								✓	✓	✓	✓					
s1	Data to be searched							✓	✓	✓	✓	✓					✓
s2	The top I/O in search area							✓	✓	✓	✓	✓					
n	Number of search data							✓	✓	✓	✓						✓

Function

A data group within the specified range is searched for specified data. The first searched position and the number of data searched from the specified range are output to from d to d+1.

Parameter

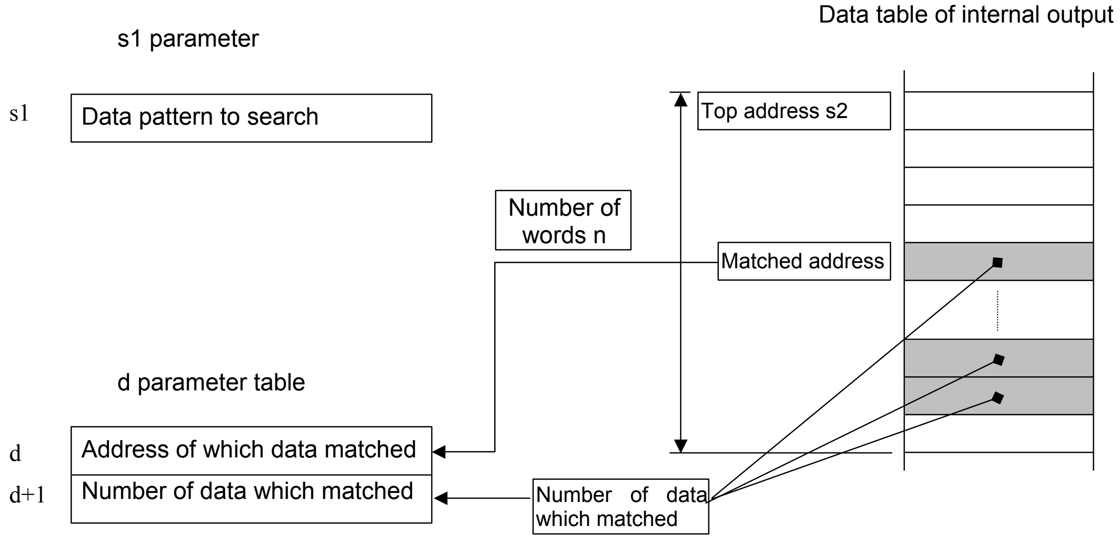
d : The data position (a relative position from the top I/O) searched first is stored.

d+1 : The number matched to data to be searched in the specified area is stored.

s1 : Data to search or the internal output in which data is stored is specified.

s2 : The top I/O in the area to search is specified.

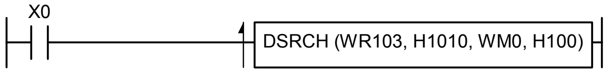
n : The number of words in the are to search is specified.


Data table of internal output
s1 parameter
s1
Data pattern to search
Top address s2
Number of words n
Matched address
d parameter table
d
Address of which data matched
d+1
Number of data which matched
Number of data which matched

Cautionary notes

- The internal output used for s1, s2, n, and d and the search area should be specified within the range of I/O number.
- Take care that the search area does not overlap with s1, s2, and d parameters.
  If the area overlaps, the command is not executed because of DER=1.

Program example



[Program description]

256 words (H100 word) counting from WM0 is searched for data which is H1010 at the rising of X0.

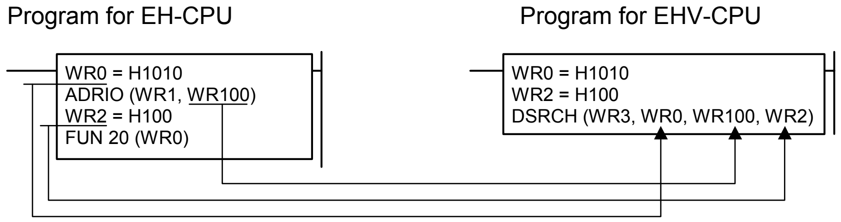
The search result is set into WR103 (data position) and WR104 (number of data).

PRN ➔ PRJ

This command is equivalent to FUN 20 (s) in the program (PRN file) of EH-CPU.

When changing the program whish has used FUN 20 (s) for EHV, how to convert is as follows.

FUN 20 (s) ➔ DSRCH (s+3, s, [I/O specified by s+1], s+2)



* This command is not convertible in a conversion tool. Please convert as mentioned as above by users, yourselves.

Name	[Data search] Extract data table

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
TSRCH (d, s, n1, n2)	–	6	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		d can be used up to d+1.
Condition	Time	Condition	Time	
–	19.1	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	The top I/O to store in extract table								✓	✓	✓	✓					
s	The top I/O in data table							✓	✓	✓	✓	✓					✓
n1	Sized of 1 block							✓	✓	✓	✓	✓					
n2	Block No. to extract							✓	✓	✓	✓						✓

Function

The data with the specified block number is extracted from the data block group within the specified range. The extracted data is copied to the specified drawing area.

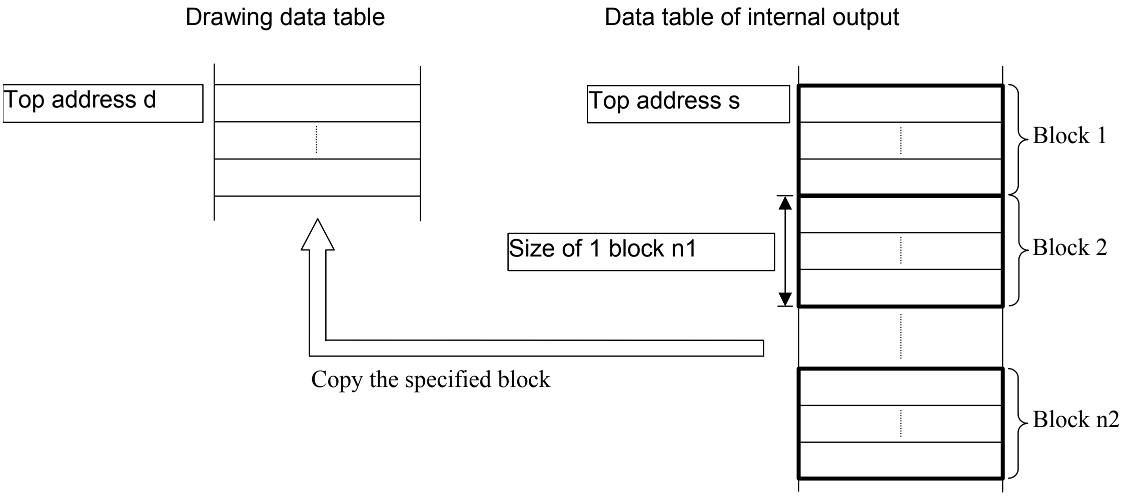
Parameter

d : The internal output to store the extracted table is specified.

s : The top I/O in the table to extract data is specified.

n1 : The size of data block (number of words) is specified.

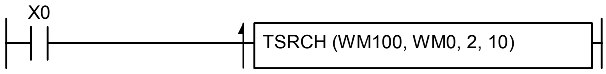
n2: The number of data blocks is specified.



Cautionary notes

- The internal output used for d and s parameters, the data table, and the drawing data table should be specified within the range of I/O number.
- Take case that all kinds of table area do not overlap with d and s parameters. If the are overlaps, the command is not executed because of DER=1.

Program example



[Program description]

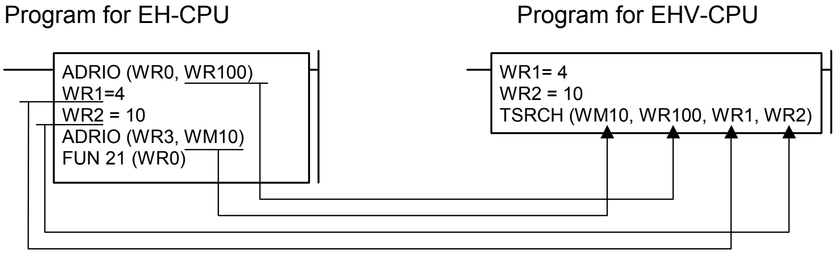
The 10th data block counting from WM0 is drawn from the data table consisting of 1 block with 2 words is set into WM100 and after WM100.

PRN ➔ PRJ

This command is equivalent to FUN 21 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 21 (s) for EHV, how to convert is as follows.

FUN 21 (s) ➔ TSRCH ([I/O specified by s+3], [I/O specified by s], s+1, s+2)



* This command is not convertible by a conversion tool. Please convert as mentioned above by users, yourselves.

Name	[Data search] Search for Maximum / Minimum / Average

Ladder format	Number of steps		Condition code				
VSRCH (d, s1, s2, n)	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	―	6	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		d can be used up to d+2 / d+5.
Condition	Time	Condition	Time	
―	21.84+16.16n	―	―	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Search result									✓	✓	✓	✓					
s	Search types								✓	✓	✓	✓	✓					✓
n1	The top I/O in area to be searched								✓	✓	✓	✓	✓					
n2	Number of data to be searched								✓	✓	✓	✓						✓

Function

The average, minimum and maximum are computed from the specified data table. (Selecting the classification from integer [word / double word] and real number to search is possible by specifying s parameter.)

Parameter

d : Search result is stored .

s1 : The classification of the number to search is specified.

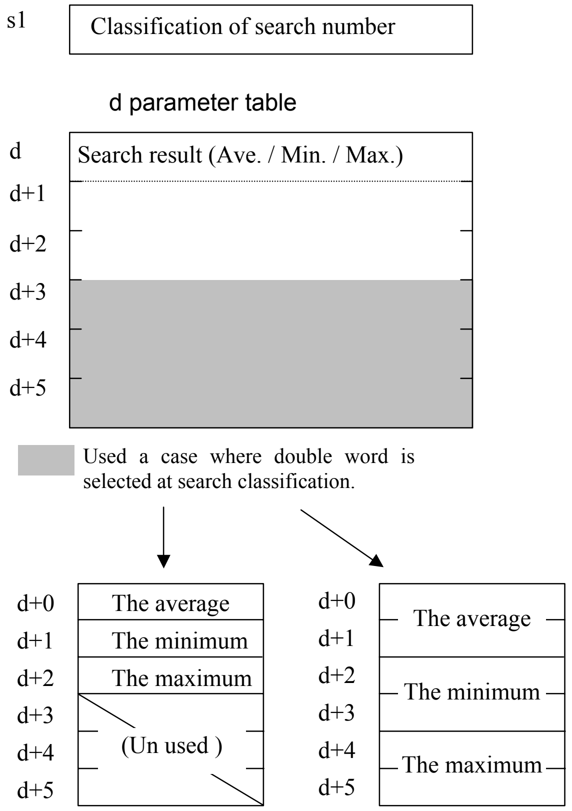
Classification of the number		Set value
Integer data (At word)	Unsigned	H0001
	Signed	H0002
Integer data (At double word)	Unsigned	H0004
	Signed	H0008
Floating point data		H000F

s2 : The top I/O in the area to be searched

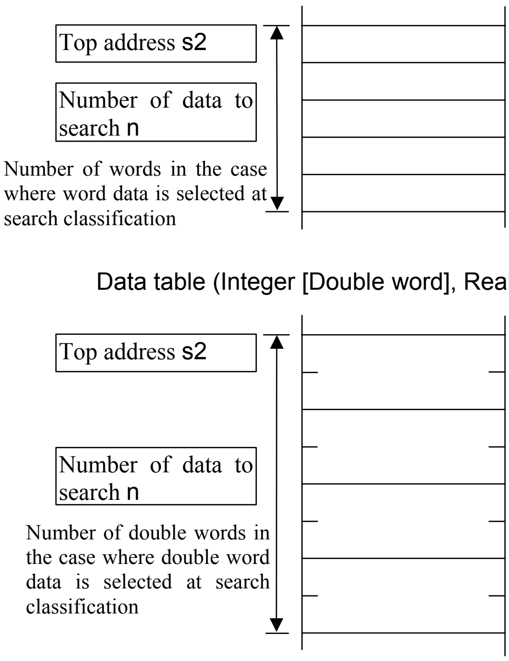
n : The number of data in the area to be searched is specified. The valid ranges of n are as follows.

At specified word: 1 to 65,535 (in decimal), H0001 to HFFFF (in hexadecimal)

At specified double word and floating point: 1 to 32,767 (in decimal), H0001 to H7FFF (in hexadecimal)



Data table (Integer [Word])

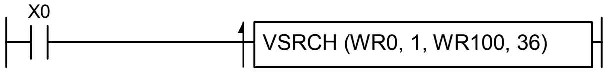

Top address s2
Number of data to search n
Number of words in the case where word data is selected at search classification
Data table (Integer [Double word], Real number)
Top address s2
Number of data to search n
Number of double words in the case where double word data is selected at search classification

Cautionary notes

- When the specified value of search classification is abnormal, the operation is performed because of DER=1.
- When the integer (word) is specified at classification of search number, only d to d+2 of the calculation result are stored.

  (As for d+3 to d+5, a precious value to the command execution is retained.)
- When the integer is specified at classification of search number, the value of which a fraction to a decimal point is rounded down is the average.
- The internal output used for the parameter table and the data table should be specified within the range of I/O number.
- If the area of the data table overlaps with the area of s parameter, the operation is not performed because of DER=1.
- If the number of data to be searched 8n) is 0, the operation is not performed because of DER=1.
- If the result is without the range from –1e+37 to 1e+37 at the operation of floating decimal data, the result is not output because of DER=1.
- If the value of s parameter or the data table is changed during executing this command, a correct operation result cannot be obtained.

Program example



```
X0
VSRCH (WR0, 1, WR100, 36)
```

[Program description]

An average value, a minimum value, and a maximum value of the 36-word unsigned integer data counting from WR100 which are computed at the rising of X0 are set into WR0 (average), WR1 (minimum), and WR2 (maximum) respectively.

PRN ➔ PRJ

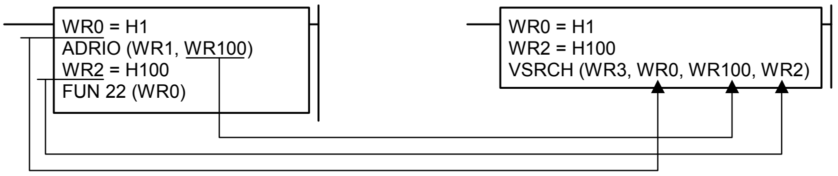
This command is equivalent to FUN 63 (s) in the program (PRN file) of EH=CPU.

When changing the program which has used FUN 63 (s) for EHV, how to convert is as follows.

FUN 63 (s) ➔ VSRCH (s+3, s, [I/O specified by s+1], s+2)

Program for EH-CPU

Program for EHV-CPU



```
WR0 = H1
ADRIO (WR1, WR100)
WR2 = H100
FUN 22 (WR0)
```

```
WR0 = H1
WR2 = H100
VSRCH (WR3, WR0, WR100, WR2)
```

* This command is not convertible by a conversion tool. Please convert as mentioned above by users, yourselves.

Name	[Data exchange] Exchange

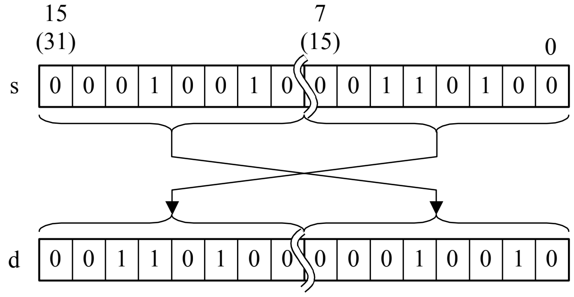
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
SWAP (d, s)	Word	4	●	●	●	●	●
	Double word	5					

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Word	0.32	—	—	
Double word	0.34	—	—	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O after exchange									✓	✓	✓	✓		✓	✓	✓	
s	I/O to exchange								✓	✓	✓	✓	✓	✓	✓	✓	✓	

Function

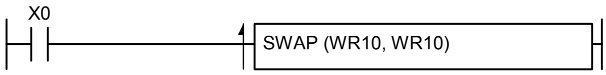
- When s is word, the upper 8 bits is exchanged for the lower 8 bits in the content of s to store in d.
- When s id double world, the upper words is exchanged for the lower word in the content of s to store in d.



Cautionary notes

A topup of this command should be set to the edge trigger.

Program example



[Program description]

The upper 8 bits and the lower 8 bits in WR10 are exchanged at the rising of X0 and stored in d.

WR10 $\boxed{\text{H 1 2 3 4}}$ ⟹ WR10 $\boxed{\text{H 3 4 1 2}}$

* Since this command is executed at every scan if it is the edge trigger, the upper and the lower in WR10 are exchange at every scan.

PRN ➔ PRJ

This command is equivalent to SWAP (d) in the program (PRN file) of EH-CPU.

When changing the program which has used SWAP (d) for EHV, how to convert is as follows.

SWAP (d) ➔ SWAP (d, d) the same I/O is set to d and s both.

* If converted by a conversion tool, it is converted as mentioned above.

Name	[Data exchange] Block exchange

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
XCG (d1, d2, n)	–	5	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Bit	4.04+0.26n	–	–	
Word	4.25+0.25n	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d1	The top I/O to be exchanged				✓						✓						
d2	The top I/O to exchange				✓						✓						
n	Number of bits to exchange Number of words to exchange							✓	✓	✓	✓	✓					✓

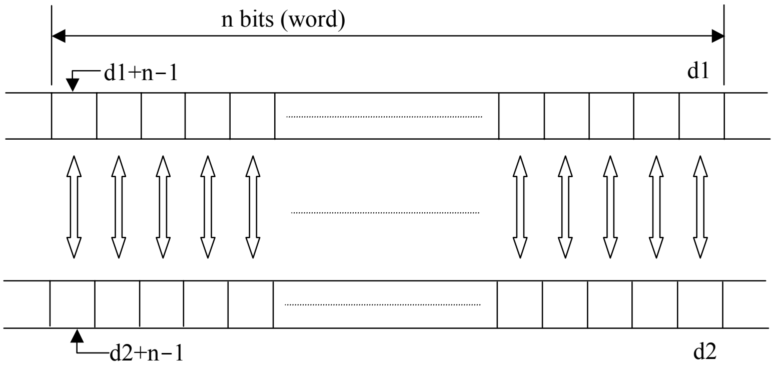
Function

- The content from d1 to n bits (words) and the content from d2 to n bits (words) are exchanged.
- When n is words,

  the number of bits (words) exchanged the content (0 to 255) of the lower 8 bits (b7 to b0) in n (WX, WEX, WY, WEY, WR, WL, WM, WN, TC) is specified. (The upper bits are ignored and it is considered to be '0'.)

  n (a constant) can specify from 0 to 255. (Decimal)
- The combination of d1 and d2 are as follows.

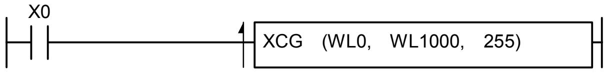
d1	d2
Bit	Bit
Word	Word



Cautionary notes

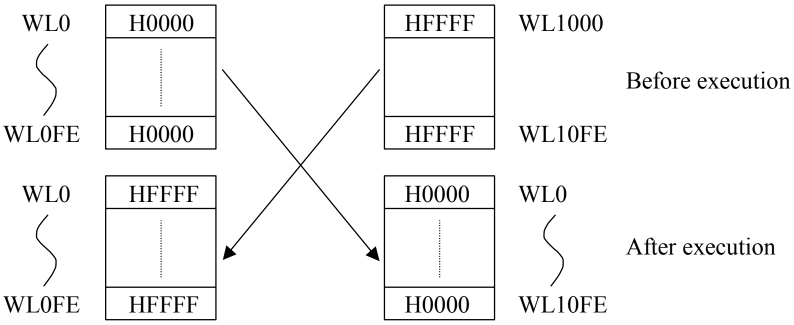
- If n=0, the batch exchange is not performed. DER becomes 0.
- d1+n−1 and d2+n−1 should be used within the I/O range. If exceeded, it exchanges up to the maximum range of the number of bits (words) of smaller one of the number of bits (words) specified to d1 and d2 because of DER=1.

Program example



[Program description]

The content from WL0 to WL0FE and the content from WL1000 to WL10FE are exchanged at the rising of X1.



Name	[Data transfer] Block transfer

Ladder format	Number of steps			Condition code				
	Condition		Steps	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s		DER	ERR	SD	V	C
MOV (d, s, n)	I/O	I/O / C	5	↕	●	●	●	●
	I/O	Array (I/O / C)	6					
	Array (I/O)	I/O / C	6					
	Array (I/O)	Array (I/O / C)	7					
	Array (C)	I/O / C	6					
	Array (C)	Array (I/O / C)	7					

Command processing time (μs)		Remarks
Condition	Time	・C means a constant. ・n means the number of words. ・The upper part means bit and the lower means word in the command processing time.
d : I/O, s : I/O / C	6.8 + 0.2n	
	6.82 + 0.18n	
d : I/O, s : Array (I/O / C)	6.8 + 0.2n	
	6.82 + 0.18n	
d : Array (I/O / C), s : I/O / C	6.8 + 0.2n	
	6.82 + 0.18n	
d : Array (I/O / C), s : Array (I/O / C)	6.8 + 0.2n	
	6.82 + 0.18n	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	The top I/O for transferring destination		✓	✓	✓				✓	✓	✓	✓					
s	The top I/O for transferring source	✓	✓	✓	✓			✓	✓	✓	✓	✓					
n	Number of bits (words) to transfer							✓	✓	✓	✓	✓					✓

Function

- n bits (or words) of I/O data specified by s are transferred to the I/O specified by d.

  The combinations of d, s, and n are shown below.

d	s	n	Remarks
Bit	Bit	A constant	n is from 0 to 1023.
		Word I/O	n is data from b0 to b9.
Word	Word	A constant	n is from 0 to 1023.
		Word I/O	n is data from b0 to b9.

- The value form s to s+n-1 is retained.
- If ranges of a transferring source and a transferring destination overlap, it changes to a transferred value.
- An array constant can be used for d and s parameters.

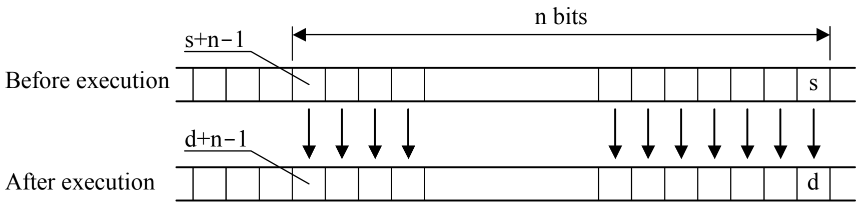
  Example) MOV (WR100(WR0), WN1000(WR0), 32)

  If WR0 = H10, the 32-word data from WR110 is transferred to the 32-word from WN1010.

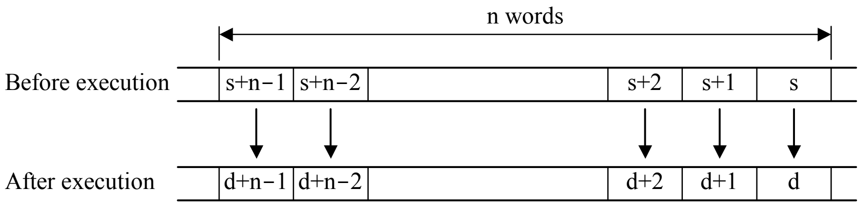
  * In EHV, there is no command which is equivalent to FUN 120 (Index setting / Argument d), FUN 121 (Index setting / Argument s) and FUN122 (Index canceling) in EH-CPU.

- Bit ➔ Bit

Data is transferred from the bit I/O to the bit I/O.



- Word ➔ Word

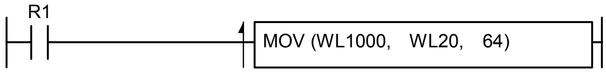


Cautionary notes

- d+n−1 and s+n−1 should be used within the range of I/O for EHV-CPU. If exceeded, it is transferred up to the maximum range because of DER(R7F4)=1.
- If n=0, the batch transfer is not performed. DER becomes 0.

  If usable maximum I/O number is exceeded when the array is used, it is transferred up to the maximum range because of DER=1.

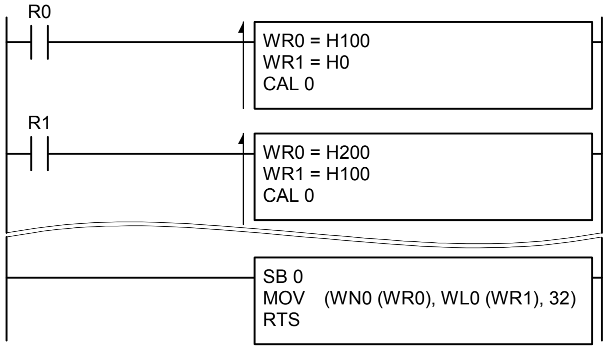
Program example



```
MOV (WL1000,   WL20,   64)
```

[Program description]

The 64 word data is transferred from the link system 1 in the 1st link to the link system 2 in the 2nd link at the rising of R1. The transfer area is set to from WL20 to WL5F and from WL1000 to WL103F, respectively.



```
WR0 = H100
WR1 = H0
CAL 0
```

```
WR0 = H200
WR1 = H100
CAL 0
```

```
SB 0
MOV   (WN0 (WR0), WL0 (WR1), 32)
RTS
```

[Program description]

- SB 0 is called at the rising of R0, and 32 words from WL0 is transferred to 32 words from WN100.

•SB 0 is called at the rising of R1, and32 words from WL100 is transferred to 32 words from WN200.

Name	[Data transfer] Bit block transfer

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
BMOV (d, s, n1, n2)	–	6	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		n1 is from 1 to 16.
Condition	Time	Condition	Time	
d : Bit / s : Bit	14.8+3.2n	–	–	
d : Bit / s : Word	20.43+7.57n	–	–	
d : Word / s : Bit	20.38+7.62n	–	–	
d : Word / s : Word	20.34+11.66n	–	–	

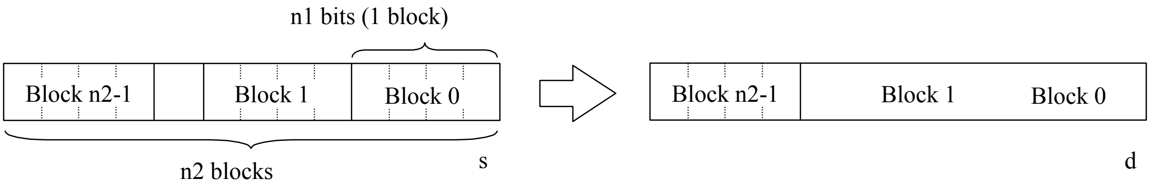
Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	The top I/O for transferring destination		✓	✓	✓				✓	✓	✓	✓					
s	The top I/O for transferring source	✓	✓	✓	✓			✓	✓	✓	✓	✓					
n1	Number of bits for 1 block							✓	✓	✓	✓	✓					✓
n2	Number of blocks to be transferred							✓	✓	✓	✓	✓					✓

Function

This is a command to transfer the specified number of bits to another area, considering some bits to be one block.

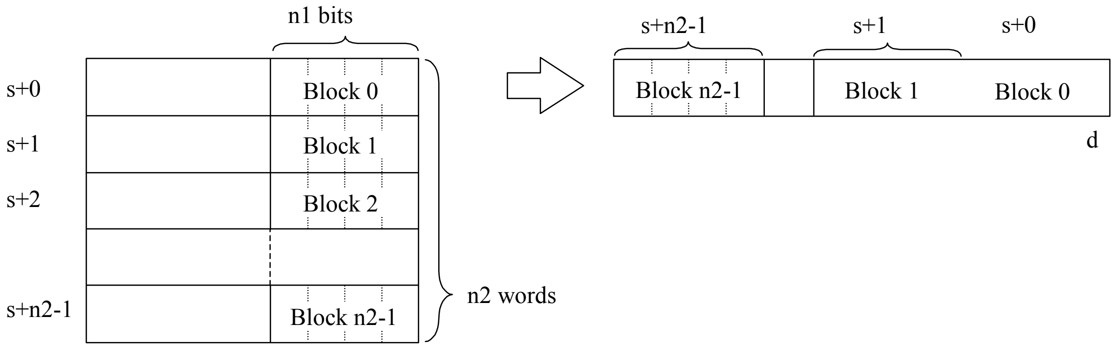
(1) When both d and s are specified to the bit I/O.

Considering n1 bits to be one block, n2 blocks are transferred from the bit I/O specified by s, putting the bit I/O specified by d to the top.



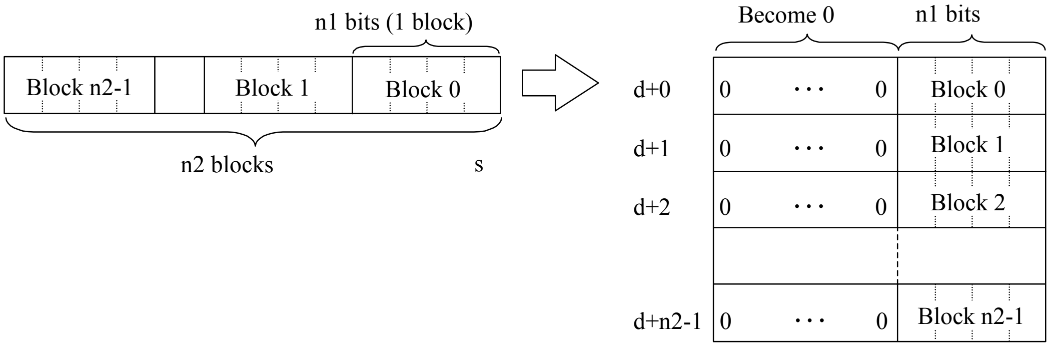
(2) When d is specified to the bit I/O and s is specified to the word I/O.

The lower n1 bits in n2 words are transferred from the word I/O specified by s, putting the bit I/O specified by d to the top.



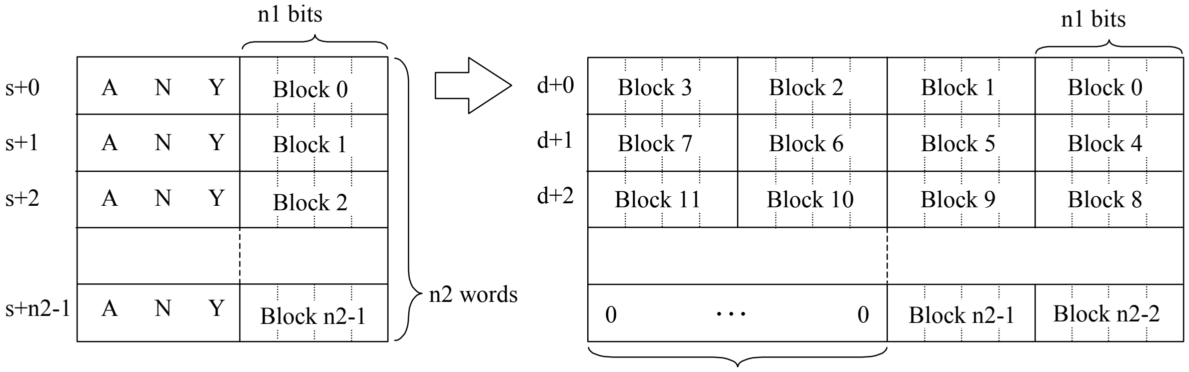
(3) When d is specified to the word I/O and s is specified to the bit I/O.

Considering n1 bits from the bit I/O specified by s to be one block, n2 blocks is transferred to the lower in the word I/O, putting the word I/O specified by d to the top.



(4) When both d and s are specified to the word I/O.

Considering the lower n1 bits specified by s to be one block, n2 blocks is transferred putting the word I/O specified by d to the top. (Each block is stored consecutively.)

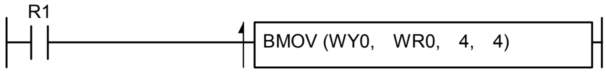


If the transfer result is less than 1 word, the upper part stores 0.

Cautionary notes

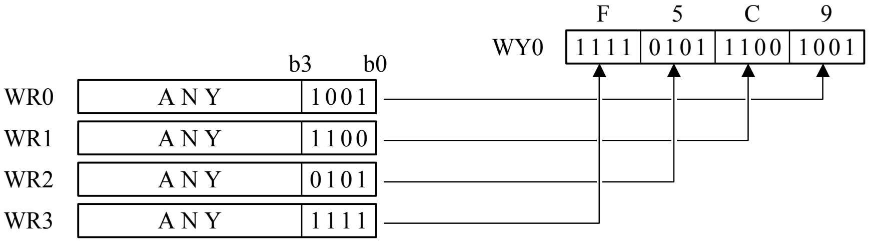
- The value of s which is a transfer source is retained. But if overlapping a transfer destination with a transfer source is specified, it changes to the transferred value.
- When d is the word I/O and s is the bit I/O, the upper bits in d after transferring becomes 0.
- I/Os of a transfer source and a transfer destination should be used within the I/O range of EHV-CPU. If exceeded, it is transferred up to the maximum range because of DER(R7F4) =1.
- n1 is valid from 0 to 16. If a value outside the valid range is set, the command is not performed because of DER=1.
- If n1=0 or n2=0, the batch transfer is not performed. DER become 0.

Program example



[Program description]

4 blocks considering 4 bits from the lower from WR to WR3 to be 1 block is transferred to WY0 at the rising of R1.



Name	[Data transfer] Copy

Ladder format	Number of steps			Condition code				
	Condition		Steps	R7F4	R7F3	R7F2	R7F1	R7F0
	d	s		DER	ERR	SD	V	C
COPY (d, s, n)	I/O	I/O / C	5	↕	●	●	●	●
	I/O	Array (I/O / C)	6					
	Array (I/O)	I/O / C	6					
	Array (I/O)	Array (I/O / C)	7					
	Array (C)	I/O / C	6					
	Array (C)	Array (I/O / C)	7					

Command processing time (μs)		Remarks
Condition	Time	・C means a constant. ・n means the number of words. ・The upper column means bit and the lower means word in the command processing time.
d : I/O, s : I/O / C	4.3 + 0.01n	
	3.8 + 0.18n	
d : I/O, s : Array (I/O / C)	5.8 + 0.01n	
	4.8 + 0.18n	
d : Array (I/O / C), s : I/O / C	4.4 + 0.01n	
	3.8 + 0.18n	
d : Array (I/O / C), s : Array (I/O / C)	5.8 + 0.01n	
	4.8 + 0.18n	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Top I/O for transferring destination		✓	✓	✓					✓	✓	✓	✓					
s	Top I/O for transferring source/O	✓	✓	✓	✓				✓	✓	✓	✓	✓					
n	Number of bits (words) to transfer								✓	✓	✓	✓	✓					✓

Function

- n bits (or words) of I/O data specified by s is copied to I/O specified by d.

  The combinations of d, s, and n are shown below.

d	s	n	Remarks
Bit	Bit	A constant	n is from 0 to 1023.
		Word I/O	n is data from b0 to b9.
Word	Word	A constant	n is from 0 to 1023.
		Word I/O	n is data from b0 to b9.

- The value of s is retained.
- If ranges of a transfer source and a transfer destination overlap, it changes to the copied value.
- An array variable can be used for d and s parameters.

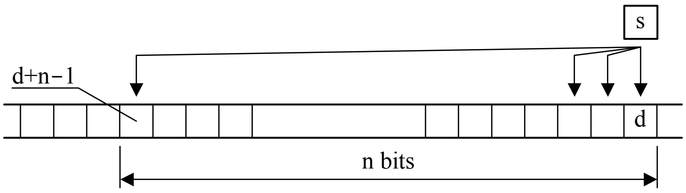
  Example) COPY (R100(WR10), M0(WR10), 16)

  If WR10 = H20, data in M20 is copied to 16 bits (from R120 to R12F) putting R120 to the top.

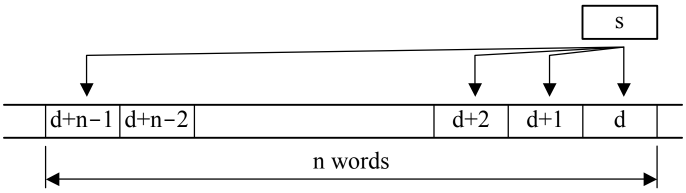
  * In EHV, there is no command which is equivalent to FUN 120 (Index setting / Argument d), FUN 121 (Index setting / Argument s), and FUN122 (Index canceling) in EH-CPU.

- Bit ➔ Bit

  Data is transferred from the bit I/O to the bit I/O.



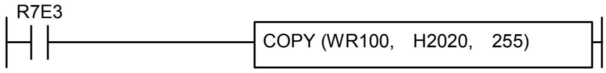
- Word ➔ Word



Cautionary notes

- d+n-1 and s+n-1 should be used within the I/O range of EHV-CPU. If exceeded, it is transferred up to the maximum range because of DER=1.
- If n=0, the batch transfer is not performed. DER(R7F4) becomes 0.
- If usable maximum I/O number is exceeded when the array is used, it is transferred up to the maximum range because of DER =1.

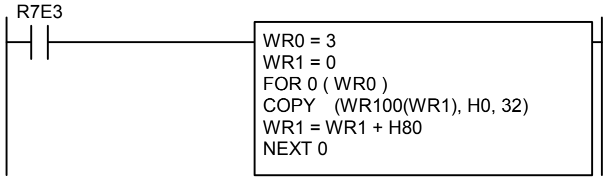
Program example



```
COPY (WR100,   H2020,   255)
```

[Program description]

A communication area which is considered to be from WR100 to WR1FE is covered with a space code (H20) as a default value at the 1st scan after the beginning of RUN.



```
WR0 = 3
WR1 = 0
FOR 0 ( WR0 )
COPY   (WR100(WR1), H0, 32)
WR1 = WR1 + H80
NEXT 0
```

[Program description]

Three areas are set to 0 by using the array.

At the 1st scan after the beginning of RUN, from WR100 to WR11F, from WR180 to WR19F, and from WR200 to WR21F are set to 0.

Name	[Data transfer] Bit block copy

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
BCOPY (d, s, n1, n2)	—	6	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		n1 is from 1 to 16.
Condition	Time	Condition	Time	
d : Bit / s : Bit	17.93+7.07n	—	—	
d : Bit / s : Word	12.94+7.06n	—	—	
d : Word / s : Bit	11.78+0.22n	—	—	
d : Word / s : Word	13.09+11.91n	—	—	

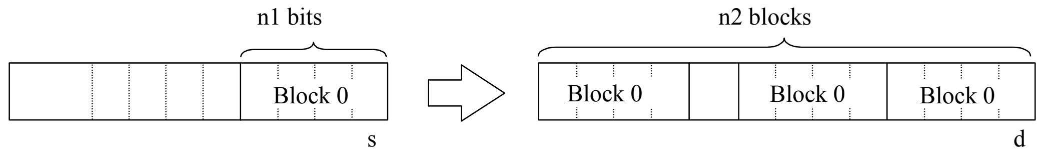
Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Top I/O for transferring destination		✓	✓	✓				✓	✓	✓	✓					
s	Top I/O for transferring source	✓	✓	✓	✓			✓	✓	✓	✓	✓					
n1	Number of bits in 1 block							✓	✓	✓	✓	✓					✓
n2	Number of copy blocks							✓	✓	✓	✓	✓					✓

Function

This is a command to copy a specified block to another area considering some bits to be a block.

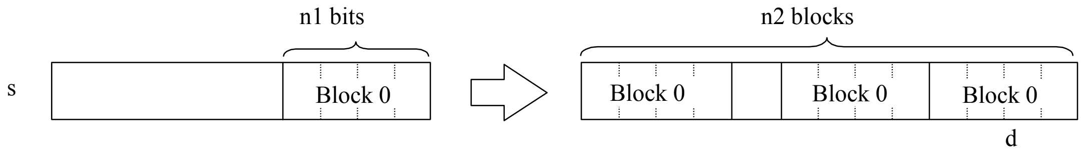
(1) When both d and s are specified to the bit I/O.

Considering n1 bits from the bit I/O specified by s to be 1 block, the same block is copied n2 times, putting the bit I/O specified by d to the top.



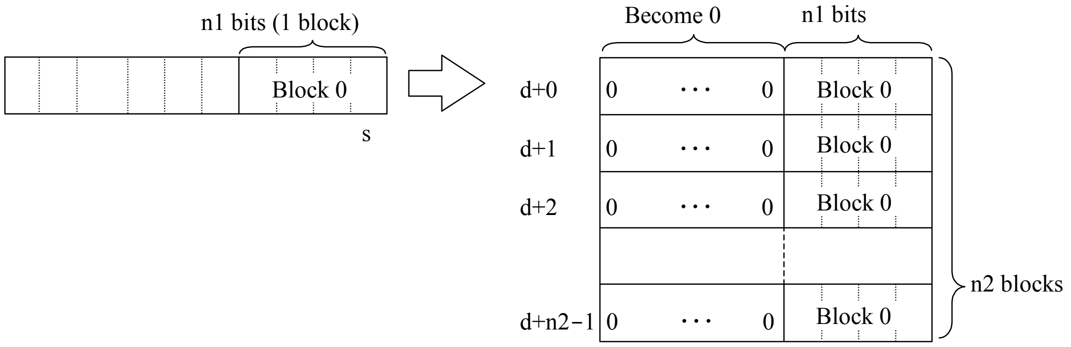
(2) When d is specified to the bit I/O and s is specified to the word I/O.

Considering the lower n1 bits in the word I/O specified by s to be 1 block, the same block is copies n2 times, putting the bit I/O specified by d to the top.



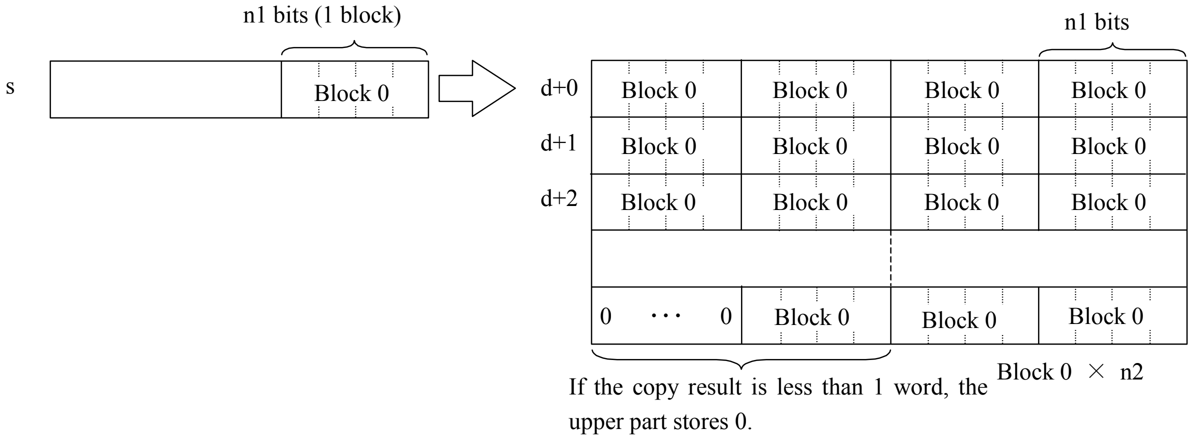
(3) When d is specified to the word I/O and s is specified to the bit I/O.

Considering n1 bits from the bit I/O specified by s to be 1 block, the same block is copies to the lower part in the word I/O n2 times, putting the word I/O specified by d to the top.



(4) When both d and s are specified to the word I/O.

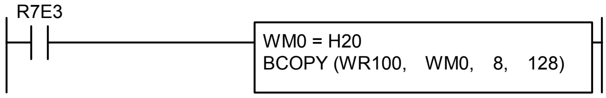
Considering the lower n1 bits in the word I/O specified by s to be 1 block, the same block is copied n2 times, putting the word I/O specified by d to the top. (Each block is stored consecutively.)


n1 bits (1 block)
s
Block 0
n1 bits
d+0
d+1
d+2
0 ・・・ 0
If the copy result is less than 1 word, the upper part stores 0.
Block 0 × n2

Cautionary notes

- The value of s which is a transfer source is retained. But if overlapping a transfer destination with a transfer source is specified, it changes to the transferred value.
- When d is the word I/O and s is the bit I/O, the upper bits in d after transferring becomes 0.
- I/Os of a transfer source and a transfer destination should be used within the I/O range of EHV-CPU. If exceeded, it is transferred up to the maximum range because of DER(R7F4)=1.
- n1 is valid from 0 to 16. If a value outside the valid range is set, the command is not performed because of DER=1.
- If n1=0 or n2=0, the batch transfer is not performed. DER becomes 0.

Program example



```
WM0 = H20
BCOPY (WR100,   WM0,   8,   128)
```

[Program description]

A communication area which is considered to be Wr100 to WR17F is covered with the space code (H20) as a default value at the 1st scan after the beginning of RUN. (128 blocks of 1 byte data are copied.)

Name	[Decode / Encode] Decode

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
DECO (d, s, n)	—	5	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
d: Bit / n : 1 – 4	0.7+0.13*(2^n)	—	—	
d: Bit / n : 5 – 16	3.5+0.023*(2^n)			
d: Word / n : 1 – 4	0.7+0.13*(2^n)			
d: Word / n : 5 - 16	3.5+0.023*(2^n)	—	—	

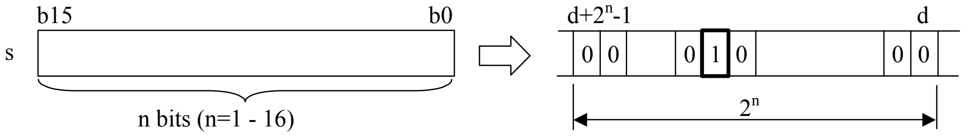
Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Top I/O for a decoding destination				✓				✓	✓	✓	✓					
s	I/O to decode							✓	✓	✓	✓	✓					✓
n	Number of bits to decode																✓

Function

(1) When I/O for a decoding destination is a bit area.

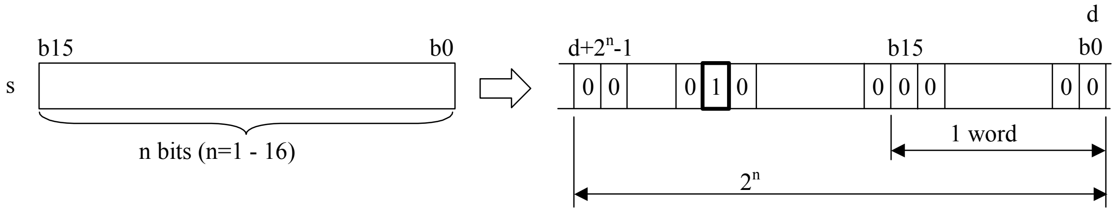
The lower n bits is decoded to 2^n and 1 is output to the decoded bit of the bit string from d to d+2^n-1.

(n= 1 - 16)



(2) When I.O for a decoding destination is a word area.

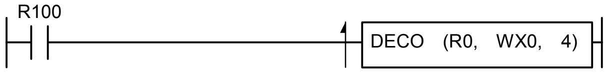
The lower n bits is decoded to 2^n and 1 is output to the decoded bit of the bit string from the 0th bit in d to d+2^n-1.



Cautionary notes

- d+2n−1 should be used within the I/O range.
- When n is 0, the command is not performed. The content from d to d+sn-1 retains the original value.
- n should be specified between 1 and 16.

Program example



[Program description]

If WX0＝HFFFF, RF, which is the 15 bit from R in bits indicated with the value of the lower 4 bits in WX, is set to 1 at the rising of R100.

Name	[Decode / Encode] Encode

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
ENCO (d, s, n)	–	5	●	●	●	●	↕

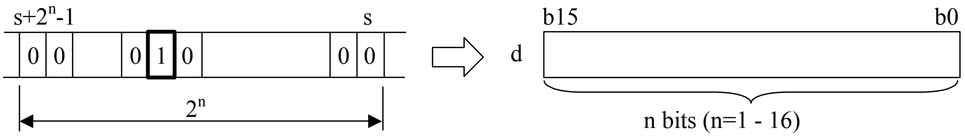
Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
s: Bit / n : 1 – 4	2.25+0.3*(2^n)	–	–	
s: Bit / n : 5 – 16	2.5+0.015*(2^n)			
s: Word / n : 1 – 4	2.25+0.3*(2^n)			
s: Word / n: 5 - 16	2.7+0.01*(2^n)	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Top I/O for an encoding destination				✓				✓	✓	✓	✓					
s	Top I/O in the bit string to encode							✓	✓	✓	✓	✓					✓
n	Number of bits to encode																✓

Function

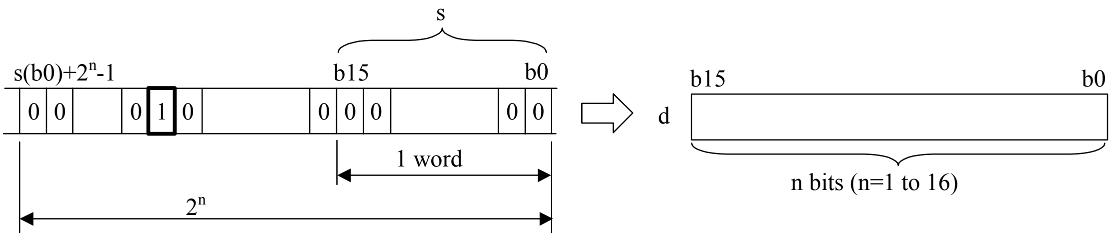
(1) When I/O for an encoding destination is a bit area.

The bit position 2n is encoded to n bits between s and s+2n-1 and the result is output to d. (n=1 to 16)



(2) When I/O for an encoding destination is a word area.

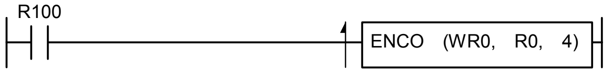
The bit position 2n is encoded to n bits between 0 bit and 2n-1 in s and the result is output to d. (n=1 to 16)



Cautionary notes

- s+2n−1 should be used within the I/O range.
- If there is several ‘1’ between s and s+2n-1 or 0 bit and 2n-1 bits in s, the larger bit position is encoded.
- If n is 0, the command is not performed. d retains the original value.
- n should be specified between 1 and 16.
- If all bits between s and s+2n-1 are 0, 0 is output to d and C(R7F0) is set to 1. In other cases, C(R7F0) is set to 0.

Program example



[Program description]

At the rising of R100, the most significant bit to which "1" is set is detected from the bit string which is from R0 to R00F (2^4-1=15 bits) and the number of binaries of 4 bits is set to the word I/O of d.

Example) If the 7th bits and 6th bits between R0 and RF are set to 1, WR0 is set to H0007 .

Name	[Decode / Encode] 7 Segment code

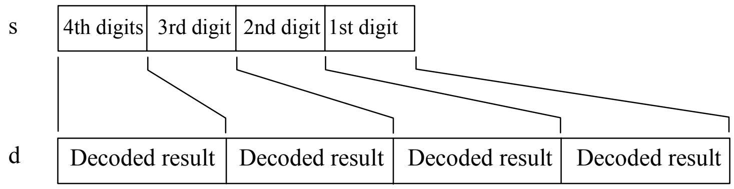
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
SEG (d, s)	—	4	●	●	●	●	●

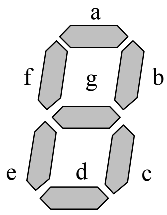
Command processing time (μs)				Remarks
Average		Maximum		n is from 1 to 16 (in decimal).
Condition	Time	Condition	Time	
—	0.74	—	—	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	Top I/O for a decoding destination														✓	✓	
s	Content of decoding							✓	✓	✓	✓	✓					✓

Function

The result which converted the content of s to the display code of 7 segment consisting of 4 digits of which 1 digit has 4 bits is output to d.



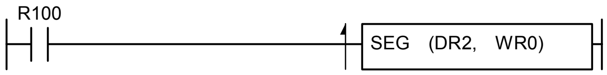


Input data 4 bits	Output data								7SEG display
	Res.	g	f	e	d	c	b	a	
0	0	0	1	1	1	1	1	1	0
1	0	0	0	0	0	1	1	0	1
2	0	1	0	1	1	0	1	1	2
3	0	1	0	0	1	1	1	1	3
4	0	1	1	0	0	1	1	0	4
5	0	1	1	0	1	1	0	1	5
6	0	1	1	1	1	1	0	1	6
7	0	0	1	0	0	1	1	1	7
8	0	1	1	1	1	1	1	1	8
9	0	1	1	0	1	1	1	1	9
A	0	1	1	1	0	1	1	1	A
B	0	1	1	1	1	1	0	0	B
C	0	0	1	1	1	0	0	1	C
D	0	1	0	1	1	1	1	0	D
E	0	1	1	1	1	0	0	1	E
F	0	1	1	1	0	0	0	1	F

Cautionary notes

- 7 segment in front of CPU module is controlled by another command (SEGCTL).
- The upper 1 bit (b7, b15, b23, b31) of each digit in d always becomes 0.

Program example



[Program description]

The content of WR0 is exchanged to the display data of 7 segment LED consisting of 4 digits of which 1 digit has 8 bits at the rising of X0.

(The upper 1 bit in each digit consisting of 8 bits always becomes 0.)

Name	[Information storage / Display] Data storage (Initial setting)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
RECSET (s, n)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		n is from 1 to 32 (in decimal). s can be used up to s+3.
Condition	Time	Condition	Time	
–	26.1	–	–	

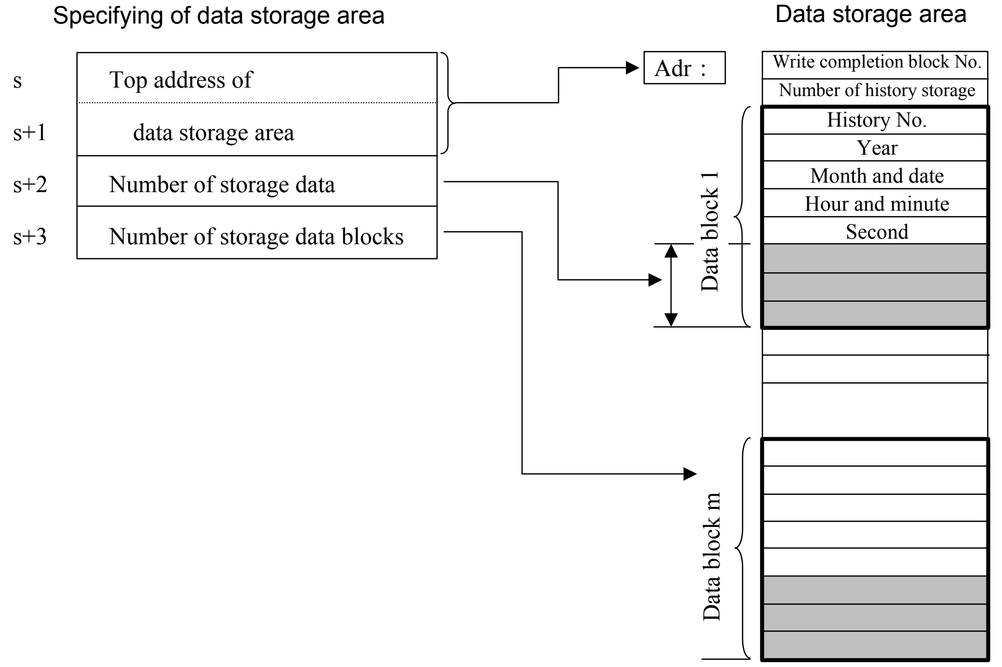
Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	Top I/O in parameter table										✓						
n	Control number																✓

Function

- The history number, the date and the time at a time of execution, and data specified by users (hereafter called data block) can be stored in the specified internal output by combining RECEXE (s, n) with this command. This is a command to perform an initial setting to memorize data block.
- The history storage can be controlled dividing to the maximum 32 sections, such as the history for event A and the history for event B.
- The user can specify the number of data to memorize. The number of words specified by the user and 5 words (the history number, the date and the time at a time of execution) are memorized at a time.
- If this command is executed, the write completion block number in the specified data storage area and the area for the number of history storage are cleared to set 0.

Parameter

Top I/O number of the parameter table to specify data storage area by s is specified.



Top address of data storage area is specified by I/O address coding command.

Cautionary notes

- The internal output used for s parameter and data storage area should be specified within the I/O number.
- If s parameter and data storage to the same control number area overlap, the operation is not performed because of DER=1.But the overlap with data storage area with different control number is not checked.
- If the number of storage data is 0, only history number and time data are memorized.
- The history storage cannot be executed until RECSET(s, n) to the control number of RECEXE(s, n) is executed.

Program example

See description pages of ”RECEXE (s)”.

Name	[Information storage / Display] Data storage (Execution)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
RECEXE (s, n)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		s uses the number of data to memorize. n is from 1 to 32 (in decimal).
Condition	Time	Condition	Time	
–	21.3	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	Data to memorize											✓						
n	Control number																	✓

Function

- The history number, the date and the time, and data specified by user area stored in data storage area specified by RECSET(s, n), when this command is executed, and the write completion block number is updated. Also the number of history storage is added.
- The address to memorize data block is computed on the system. And since data storage area is the ring buffer, if the number of data blocks specified is stored, the next data block is overwritten from the top of data storage area.

  (The number of history storage is added.)
- n parameter relates this command and RECSET. This runs according to the initial setting of RECSET(s, n) which is executed by the same value as n parameter of this command.

  Example) When there are RECSET(WR0, 1) and RECSET(WR4, 2)

  If RECEXE(WL0, 1) is written, data in WL0 and after WL0 are memorized according to the initial setting by RECSET(WR0, 1).

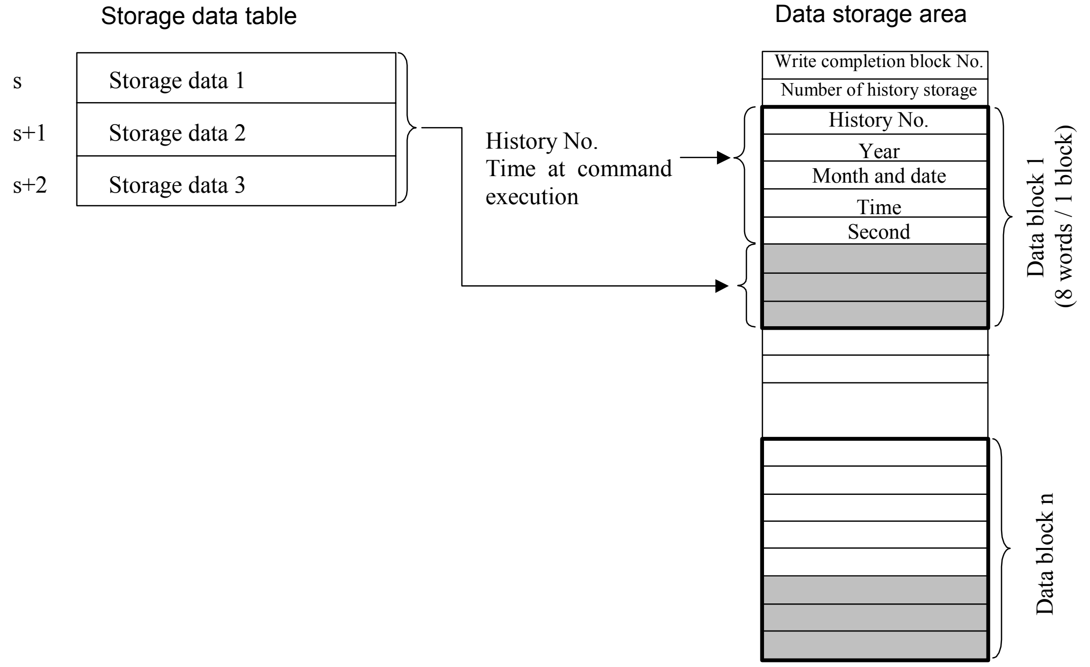
  If RECEXE(WL0, 2) is written, data in WL0 and after WL0 are memorized according the initial setting by RECSER(WR4, 2).

Parameter

Top I/O number in the table stored data to memorize by s is specified.

The number of storage data specified by RECSET(s, n) determines the size of s. (If the number of data storage is 0, the internal output of dummy should be allocated.)

Example) If the number of storage data is set to 3 by RECSET(s, n), s can be used up to s+2.



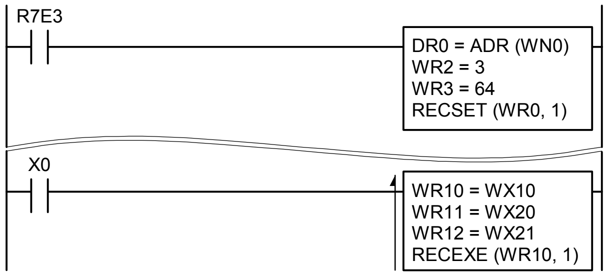
Cautionary notes

- The internal output used for s parameter should be specified within the I/O number.
- This command specifies the top I/O of the internal output in which data to store in the data storage area by s parameter is stored. Care should be taken since a purpose is different from s parameter of RECSET(s, n).
- Please program as RECSET(s, n) is executed before this command is executed. Even if this command is executed before RECSET(s, n) is executed, the operation is not performed because of DER=1.
- Although the number of times to memorize the history is added even if data is overwritten from the top of the area because the data storage area was filled, the write completion block number is back to 1 when the data was overwritten.

  Example) When the write block is 3.

The number of times to memorize the history	1	→	2	→	3	→	4	→	5	→	6	→	7	···
The write completion block number	1	→	2	→	3	→	1	→	2	→	3	→	1	···

Program example



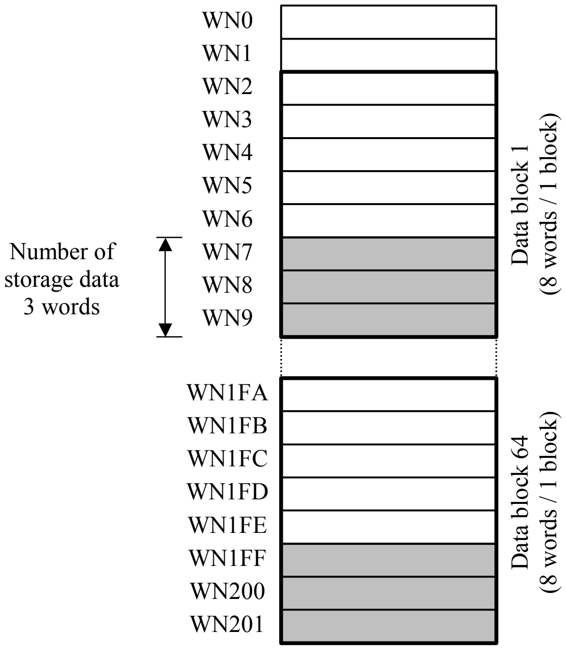
```
DR0 = ADR (WN0)
WR2 = 3
WR3 = 64
RECSET (WR0, 1)
```

```
WR10 = WX10
WR11 = WX20
WR12 = WX21
RECEXE (WR10, 1)
```

[Program description]

- Data storage area of the control No. 1 is registered at the 1st scan after RUN.

  Such data storage area as the following is set in this program.


WN0
WN1
WN2
WN3
WN4
WN5
WN6
WN7
WN8
WN9
Number of storage data 3 words
Data block 1 (8 words / 1 block)
WN1FA
WN1FB
WN1FC
WN1FD
WN1FE
WN1FF
WN200
WN201
Data block 64 (8 words / 1 block)

- Whenever X0 is turned ON, the date and the time at that time and values of WX10, WX20, and WX21 are stored in the next data block.
- If data is stored until data block 64, the next data overwrites the data block 1.

Name	[Information storage / Display] 7 segment control for CPU

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
SEGCTL (s)	–	3	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		s is valid from H00 to HFF.
Condition	Time	Condition	Time	
–	0.24	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	Value to be displayed to 7 segment								✓	✓	✓	✓	✓					✓

Function

The value of 7 segment in front of CPU module is switched to the specified value. (The value which can be specified is from H00 to HFF.) However, if this command is executed under error occurrence, the indication remains unchanged.

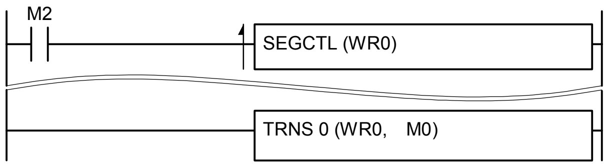
Parameter

s :The value displayed to 7 segment LED in front of CPU module is specified.
From H00 to HFF is valid.

Cautionary notes

If the value other than from H00 to HFF is specified of s parameter or a constant, the indication of 7 segment remains unchanged because of DER=1.

Program example



[Program description]

If error occurs in TRNS 0 command, the return code is displayed to 7 segment LED in front of CPU module.

Name	[I/O Refresh] All points refresh

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
ALREF	—	2	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	738	—	—	・Processing time for external input and output = Number of words × 1.4 μ sec ・CPU link module is about 1ms per stand. ・FL-NET module is about 15ms per stand.

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
—	(No argument)																

Function

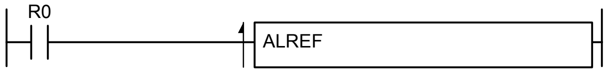
Whole external input and output and the link area are refreshed.

When this command is completed, update to a state that the input (X, EX) and the link area (L, WL, and WR at using FL-net module) are executed is performed. The output (Y, EY) is the output value before this command is executed.

Cautionary notes

If you want to perform the refresh partially, please use IOREF and SLREF.

Program example



[Program description]

All I/O are refreshed at the rising of R0.

PRN ➔ PRJ

This command is equivalent to FUN 80(s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 80 (s) for EHV, how to convert is as follows.

FUN 80 (s) ➔ ALREF s parameter is not used.

* If converted by a conversion tool, it is converted as mentioned above.

Name	[I/O Refresh] Refresh to specify classification

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
IOREF (s)	—	3	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	370	—	—	・Processing time for external input and output = Number of words × 1.4 μ sec ・CPU link module is about 1ms per stand. ・FL-NET module is about 15ms per stand.

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	I/O classification to refresh											✓						✓

Function

- The classification specified by s parameter is refreshed.
- If the input (E, EX) or the link area (L, WL, WR at using FL-net module) are used, update to a state that this command is executed is performed.
- If the output (Y, EY) is specified, the output is the output value which is specified before this command is executed.

Parameter

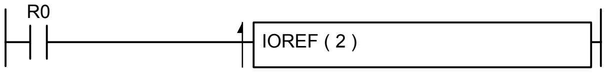
I/O classification to refresh in the word internal output specified by s is specified.

Parameter	Description	Details
s	Refresh I/O classification	H0000 ･･･ Input refresh (including remote) H0001 ･･･ Output refresh (including remote) H0002 ･･･ Link refresh

Cautionary notes

- The refresh is performed in slot units according to I/O allocation.
- If the classification of input and output is specified of other than H0000, H0001, and H0002, this command is not executed because of DER=1.

Program example



[Program description]

・The link area is refreshed at the rising of R0.

A state of input and output is not undated because of the specifying of the link area only.

PRN ➔ PRJ

This command is equivalent to FUN 81 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 81 (s) for EHV, how to convert is as follows.

FUN 81 (s) ➔ IOREF (s)

* If converted by a conversion tool, it is converted as mentioned above.

Name	[I/O Refresh] Refresh to specify slot

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
SLREF (s)	–	3	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	58	–	–	s is used the number of slots to refresh. ・Processing time for external input and output = Number of words × 1.4 μ sec ・CPU link module is about 1ms per stand. ・FL-NET module is about 15ms per stand.

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)		WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	Top I/O in table for refresh											✓						

Function

The module specified by s parameter is refreshed.

Up to the maximum of 128 slots can be specified.

Parameter

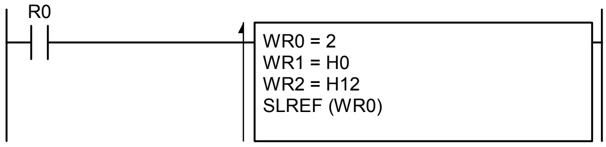
I/O classification to refresh in the word internal output specified by s is specified.

Parameter	Description	Details
s	Number of slots to refresh	The number of slots (the number of modules) to refresh is specified.
s+1～	Slot position to refresh	The number of slots of the slot position to refresh is specified. The constitution of words is represent with numbers. b15 b11 b12 b8 b7 b4 b3 b0 H0 fixed | Remote No. | Unit No. | Slot No. Remote number: 0 - 4 Unit number: 0 - 5 Slot number: 0 - A

Cautionary notes

- The internal output used for s parameter should be specified within the I/O number.
- When specifying inexistent position (position without I/O allocation) in the specifying of the slot position to refresh, the slot is not processed because of DER=1.
- If a points exceeds 128 slots, an excess over 128 slots is not processed because of DER=1.
  (The 128 slots are refreshed.)
- The area s of EX and EY which are the specified slot are also refreshed.

Program example



```
WR0 = 2
WR1 = H0
WR2 = H12
SLREF (WR0)
```

[Program description]

The 0th slot in the basic unit (Unit 0) and the 2nd slot in the 1st expansion unit (Unit 1) are refreshed at the rising of R0. (I/O data of other modules is updated at the scan END.)

PRN ➔ PRJ

This command is equivalent to FUN 82 (s) in the program (PRN file) of EH-CPU.

When converting the program which has used FUN 82 (s) for EHV, how to convert is as follows.

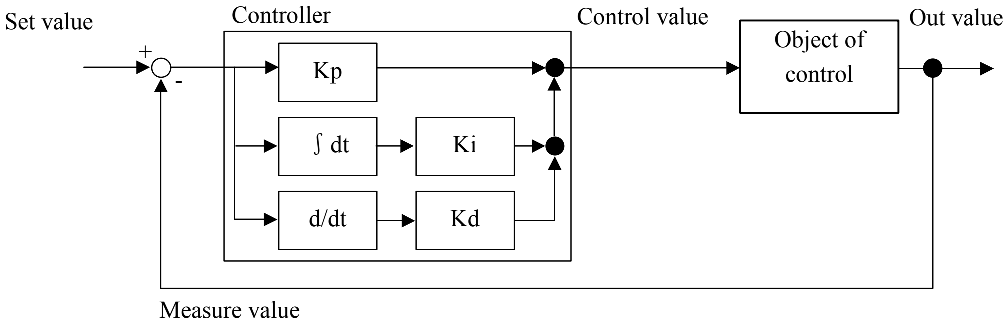
FUN 82 (s) ➔ SLREF (s)

* If converted by a conversion tool, it is converted as mentioned above.

Name	PID control

■ What is PID control

PIC control refers to the feedback control aiming at stabilizing at the set value after an out value to be controlled has become a set value speedily. The block chart of PID control is shown below.



PID stands for Proportional, Integral, and Derivative. And each item has a feedback gain.

Kp ··· Proportional gain: Kp is multiplied by the difference (deviation) between the out value and the set value. The larger Kp is, the time (response) the out value reaches the set value is. However, if Kp is too much large, the deviation cannot be eliminated and the out value oscillates near the set value.

Ki ··· Integral gain: The deviation is integrated overt a period time, and then Ki is multiplied by the total sum. Integral control can remove the deviation which cannot be eliminated by Proportional control since this acts on the past deviation.

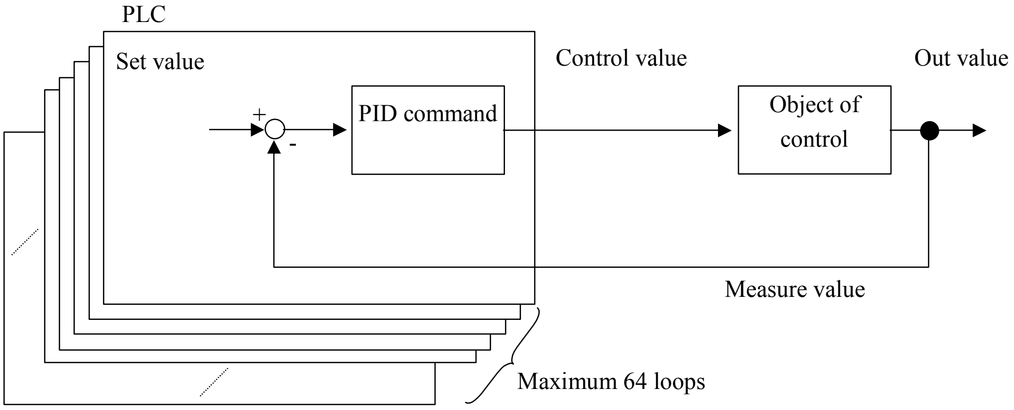
Control by P and I actions is called PI control.

Kd ··· Derivative gain: Kd is multiplied by the quantity of change. Derivative control is effective to the cause of fluctuation after the out value becomes stable at the set value since this acts on the quantity of change. However, if Kd is too large, the system may become unstable since the out value responds to a little quantity of change quickly.

■PID control in PLCZ

(1) Feature

EHV-CPU can control the feedback group of PID up to the maximum of 64 loops.



(2) Area used for PID command

PID command consists of three commands.

[1]PIDIT A table used for PID command is initialized.

[2]PIDOP A loop for PID operation is determined.

[3]PIDCL PID operation is executed.

PIDOP and PIDCL are used together and written inside the periodic scan. (PID control requires that a sampling scan is kept constant. That is why written to the periodic scan.)

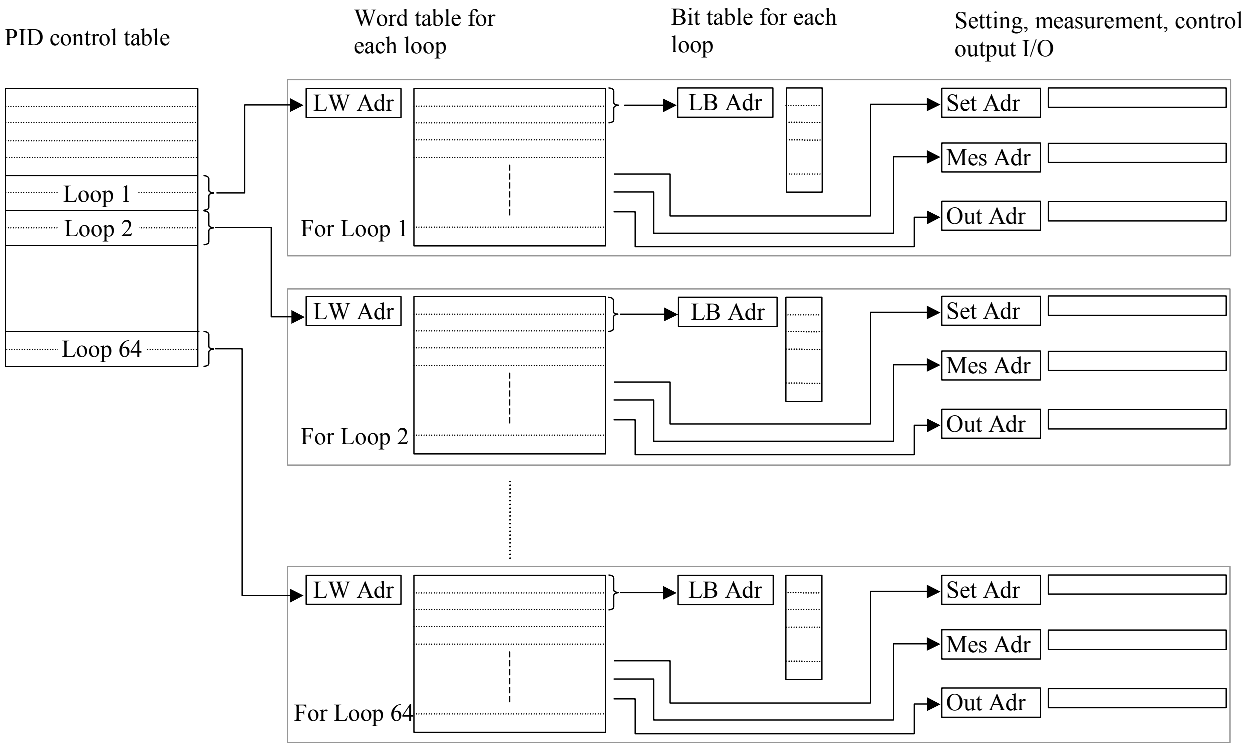
(3) Area used for PID command

The following word table and bit table are required for PID command.

[1]PID control table For all loops to control by PID.
5 + Number of loops × 2 words is required. (Number of loops is 10; 25 words)

[2]Word table for each loop A necessary table for each loop. The parameter is set.
It is performed setting the parameter in table.
Number of loops × 52 words is required.

[3]Bit table for each loop A necessary table for each loop. Used as control bit and for displaying a state of loop.



Name	[PID control] Initializing of PID

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
PIDIT (s)	–	3	●	●	●	●	●

Command processing time (µs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	93.7+5.14n	–	–	

Usable / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	Top I/O in PID control table										✓						

Function

Executing this command initializes a necessary area for PID operation.

Parameter

s : Top I/O number in PID control table.

Cautionary notes

- If the content shown in PID control table is incomplete, it is not initialized.
  (Error code is set to the area of "error code 0" in PID control table.)
- If this command is executed again after the initializing was complete normally (the area of "Indication of execution result of initializing" in PID control table is H0001), error occurs.

Name	[PID control] PID execution control

Ladder format	Number of steps		Condition code				
PIDOP (s)	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	–	3	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	22.8	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	Top I/O in PID control table											✓						

Function

Executing this command determines a loop to perform PID operation.

(A loop is determined by taking in PID execution flag and PID constant change flag from the bit table area for each loop.)

Parameter

s : Top I/O number in PID control table is specified.

Cautionary notes

- Please program so that this command is executed only once during a periodic scan.
  And a period to execute PIDOP (s) should be set to 20ms.
- If parameter s is specified of other than the top number in PID control table, error occurs.
  (Error code is set to areas of “error code 0” and “error code 1” in PID control table and this command is not executed.)

Name	[PID control] Calculation of PID

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
PIDCL (s)	–	3	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	58.8	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	PID loop word table										✓						

Function

PID calculation is performed adjusting to the sampling time set to the word table for each loop.

If PID calculation is executed, PIC calculation flag for the loop to be calculated is turned ON.

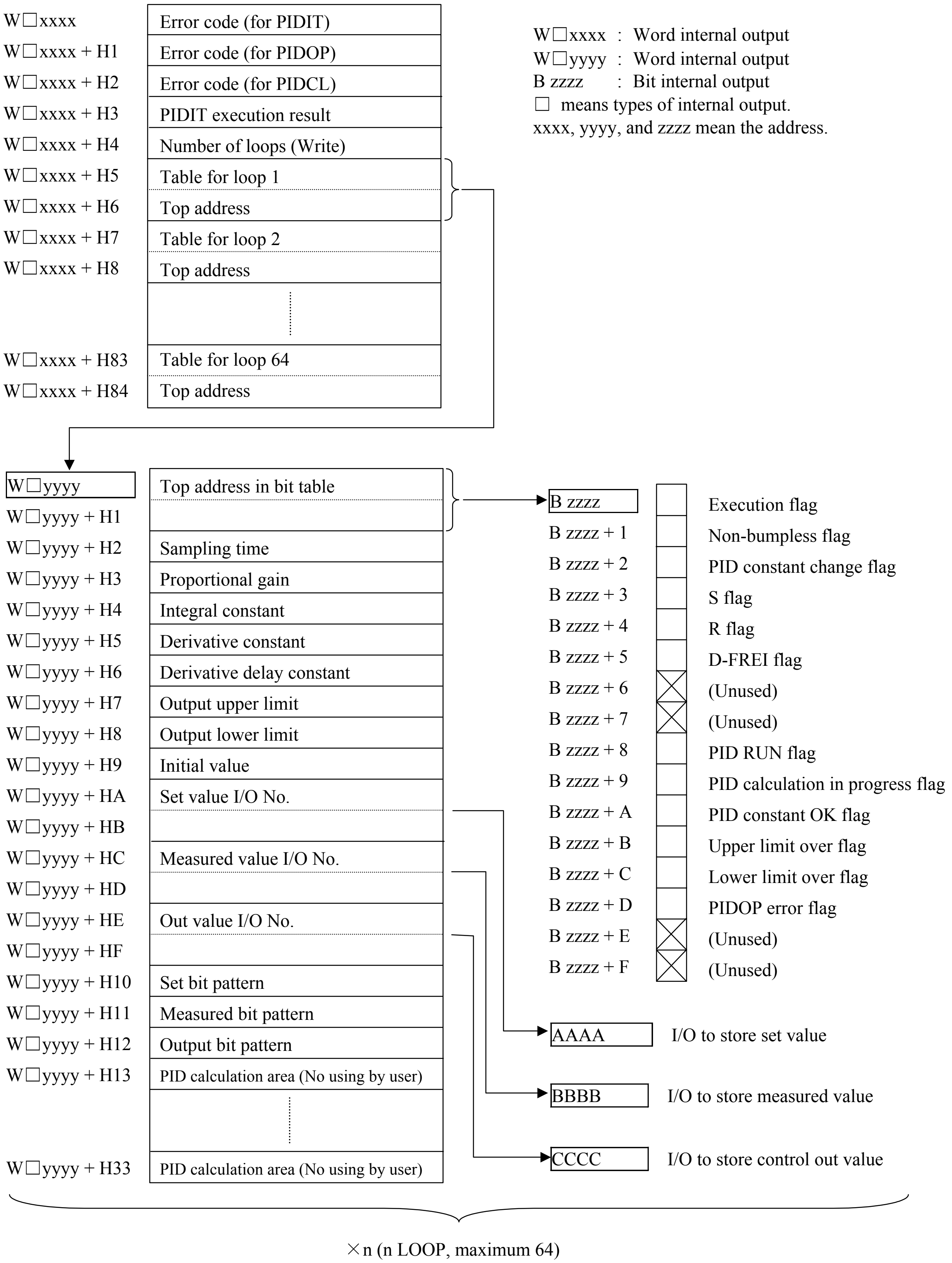
Parameter

s : The top I/O number in the word table for each loop is specified.

Cautionary notes

- Please program so that this command is executed only once during a periodic scan.
  And a period to execute PIDCL(s) should be set to 20ms.
- All top addresses in PID word table should be set before this command is executed.
- This command checks the ranges of the output maximum, the minimum, the set value bit pattern, and the out value bit pattern for each loop. If error is found, PIDCL execution flag in the bit table for each loop is turned ON, and error code is set to the are of “error code 2” in PID control table.
  (PIDCL is executed even if error is found.)

■ Details of table for PID command



W□xxxx	Error code (for PIDIT)
W□xxxx + H1	Error code (for PIDOP)
W□xxxx + H2	Error code (for PIDCL)
W□xxxx + H3	PIDIT execution result
W□xxxx + H4	Number of loops (Write)
W□xxxx + H5	Table for loop 1
W□xxxx + H6	Top address
W□xxxx + H7	Table for loop 2
W□xxxx + H8	Top address
	⋮
W□xxxx + H83	Table for loop 64
W□xxxx + H84	Top address

W□xxxx : Word internal output
W□yyyy : Word internal output
B zzzz : Bit internal output
□ means types of internal output.
xxxx, yyyy, and zzzz mean the address.

W□yyyy	Top address in bit table
W□yyyy + H1	
W□yyyy + H2	Sampling time
W□yyyy + H3	Proportional gain
W□yyyy + H4	Integral constant
W□yyyy + H5	Derivative constant
W□yyyy + H6	Derivative delay constant
W□yyyy + H7	Output upper limit
W□yyyy + H8	Output lower limit
W□yyyy + H9	Initial value
W□yyyy + HA	Set value I/O No.
W□yyyy + HB	
W□yyyy + HC	Measured value I/O No.
W□yyyy + HD	
W□yyyy + HE	Out value I/O No.
W□yyyy + HF	
W□yyyy + H10	Set bit pattern
W□yyyy + H11	Measured bit pattern
W□yyyy + H12	Output bit pattern
W□yyyy + H13	PID calculation area (No using by user)
	⋮
W□yyyy + H33	PID calculation area (No using by user)

B zzzz	□	Execution flag
B zzzz + 1	□	Non-bumpless flag
B zzzz + 2	□	PID constant change flag
B zzzz + 3	□	S flag
B zzzz + 4	□	R flag
B zzzz + 5	□	D-FREI flag
B zzzz + 6	☒	(Unused)
B zzzz + 7	☒	(Unused)
B zzzz + 8	□	PID RUN flag
B zzzz + 9	□	PID calculation in progress flag
B zzzz + A	□	PID constant OK flag
B zzzz + B	□	Upper limit over flag
B zzzz + C	□	Lower limit over flag
B zzzz + D	□	PIDOP error flag
B zzzz + E	☒	(Unused)
B zzzz + F	☒	(Unused)

AAAA	I/O to store set value
BBBB	I/O to store measured value
CCCC	I/O to store control out value

×n (n LOOP, maximum 64)

(1) Composition of PID control table

PID control table consists of [2], [3], [4], and [5]. Although the size of the table increases from the number of loops[3], it should not exceed the maximum No. of the word internal output. If exceeded, error code H0004 is written to error code 0[2].

Address	Description	Details	Remarks
xxxx	Error code 0※1 [Read]	・Error code which occurred on PIDIT process and a part of PIDOP process is set. ・If there is no error, the previous state is retained.	[2]
xxxx + 1	Error code 1※1 [Read]	・Error code which occurred on PIDOP process is set. ・If there is no error, the previous state is retained.	
xxxx + 2	Error code 2※1 [Read]	・Error code which occurred on PIDCL process is set. ・If there is error, the previous state is retained.	
xxxx + 3	PIDIT normal completion 1 [Read]	・H0001 is set when PIDIT [Initialization of PID] is executed normally. ・If there is error, H0000 is set into the error code 0	[5]
xxxx + 4	Number of loops※2 [Write]	・The number of loops used can be set from 1 to 64. ・If it is 0, PID process is not performed because H0002 is written to the error code 0.(PID process is not performed even if PIDOP and PIDCL have been programmed.)	[3]
xxxx + 5 xxxx + 6	Top address of WR in word table for loop 1※2 [Write]	A PID constant input and an internal output for PID internal calculation use 52 words per loop. If the maximum No. of the internal output is exceeded, error code XX05 is written to the error code 0.	[4]
xxxx + 6 xxxx + 7	Top address of WR in word table for loop 2※2 [Write]	A PID constant input and an internal output for PID internal calculation use 52 words per loop. If the maximum of the internal output is exceeded, error code XX05 is written to the error code 0.	
xxxx + 8 xxxx + 9	Top address of WR in word table for loop 3※2 [Write]	A PID constant input and an internal output for PID internal calculation use 52 words per loop. If the maximum of the internal output is exceeded, error code XX05 is written to the error code 0.	
...	...	...	
xxxx + 83 xxxx + 84	Top address of WR in word table for loop 64※2 [Write]	A PID constant input and an internal output for PID internal output use 52 words per loop. If the maximum of the internal output is exceeded, error code XX05 is written to the error code 0.	

*1 The error code is represented by 4 digits in hexadecimal. See the error code details for details.
*2 [Write] in the table indicates a parameter which users input on the program. (Read is also possible.)

(2) Composition of Word table for each loop

Word table is specified in the area of [5] of (1) PID control table.

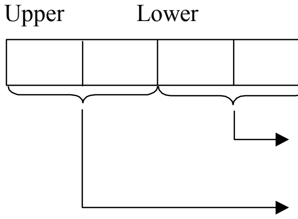
Address	Description	Specification	Details	Remarks
yyyy yyyy + 1	Top No. in bit table	The top address in bit table is set using I/O address coding command.	16 bits is used per loop. Please se a top No. within the bit internal output.	[11]
yyyy + 2	Sampling time TZ	1 to 200 (× a periodic cycle) An analog input/output is mounted on the basic base or the expansion base.	A multiple of the minimum set value is set. The minimum set value is a value set to the loop number [3].	[12]
yyyy + 3	Proportional gain KP	-1,000 to +1,000	Support from -10.00 to +10.00.	[13]
yyyy + 4	Integral constant Ti/TZ	1 to 32,767	A value of Ti/(Sampling time × Periodic cycle) is set.	[14]
yyyy + 5	Derivative constant TD/TZ	1 to 32,767	A value of TD/(Sampling time × Periodic cycle) is set.	[15]
yyyy + 6	Derivative delay constant Tn/TZ	1 to 32,767	A value of Tn/(Sampling time × Periodic cycle) is set.	[16]
yyyy + 7	Output upper limit value UL	-32,767 to 32,767	Satisfy the following relations.	[17]
yyyy + 8	Output lower limit value LL	-32,767 to 32,767	LL≤INIT≤UL	[18]
yyyy + 9	Initial value INIT	-32,767 to 32,767		[19]
yyyy + A yyyy + B	Set value I/O No. [Write]	Word No. of I/O to set a set value is set.		[20]
yyyy + C yyyy + D	Measured value I/O No. [Write]	Word No. of I/O to set a measured value is set.		[21]
yyyy + E yyyy + F	Out value I/O No. [Write]	Word No. of I/O to output PID calculation result is set.		[22]
yyyy + 10	Measured value bit pattern [Write]	Method to convert a set value to16-bit data which performs PID operation is set. One from H0001 to H0004 is set according to *1.		[23]
yyyy + 11	Measured value bit pattern [Write]	Method to convert data read from I/O No. 21 of the measured value into 16-bit data is set. [See the set value bit pattern 23]		[24]
yyyy + 12	Out value bit pattern [Write]	It is written to the out value I/O after converting PIDOP process or PID calculation result according to the out value bit pattern 25. One from H0001 to H0004 in *2 is set according to types of output I/O.		[25]
yyyy + 13 ～ yyyy + 33	PID calculation area [No using by user]	Do not use this area on user program because PIDIT, PIDOP, and PIDCL processes use.		[26]

(3) Bit table

Address	PID control table	Details	Remarks
zzzz	Execution flag [Write]	· PID calculation value is initialized checking the then PID constant at the rising of an execution flag (0→1). If it is OK, PID RUN flag [58]=1. If it is error, PID RUN flag [58]=0 and PID calculation is not performed. · While an execution flag=1, PID calculation is performed. · If an execution flag=0, PID calculation is finished and the output is set to 0.	[50]
zzzz + 1	Nonbumpless flag [Write]	0: Perform the bumpless process. 1: Perform the nonbumpless process.	[51]
zzzz + 2	PID constant change flag [Write]	· When PID constant change flag is switched from OFF to ON, the calculation is performed using re-read PID constant to use for PID calculation. · After a change of a PID constant is complete, user needs to switch this flag to OFF · If error is found on a PID constant (PID constant OK=0), the PID calculation value is held according to the previous PID constant.	[52]
zzzz + 3	S flag [Write]	S flag is a flag to change the out value to the initial value by 1. The output is shown as follows according to the relations between the out upper limit value[17], the out lower limit value[18], and the initial value[19]. Output lower limit[18]>Output upper limit[17] ···no output Output lower limit[18]≤Initial[19]≤Output upper limit[17] ···Output the initial[19] Output lower limit[18]≤Output upper limit[17]≤Initial[19] ···Output the output lower limit[18] Initial[19]≤Output lower limit[18]≤Output upper limit[17] ···Output the output upper limit[18] S flag has priority over R flag.	[53]
zzzz + 4	R flag [Write]	R flag is a flag to clear the out value to set to 0 by 1.	[54]
zzzz + 5	D-FREI flag [Write]	0: PID calculation is performed without differential and integral calculus. 1: PID calculation is performed with differential and integral calculus.	[55]
zzzz + 6	Unused		
zzzz + 7	Unused		
zzzz + 8	PID RUN flag [Write]	If PIDOP detects the rising of an execution flag [50], rationality between [12] to [16] and [20] to [22] is checked, and the result is set to PID RUN flag [58]. 1: Rational 0: Irrational When PID RUN flag [58]=1, If PIDOP detects the rising of an execution flag [50], PID RUN[58]=0 and PID process ends.	[58]
zzzz + 9	PID calculation in progress flag [Read]	PIDCL sets 1 to PID calculation in progress flags [59] for the loop to calculate PID, and 0 to other flags [59].	[59]
zzzz + A	PID constant OK flag [Read]	If PIDOP detects the rising of PID constant change flag[52], rationality of PID constant from [12] to [16] is checked, and the result is set to PID constant OK flag [60].	[60]
zzzz + B	Upper limit over flag [Read]	If the out value of PID calculated by PIDCL is larger than the output upper limit UL[17], the upper limit over flag [61] becomes 1.	[61]
zzzz + C	Lower limit over flag [Read]	If the out value of PID calculated by PIDCL is smaller than the output lower value LL[18], the lower over flag[62] becomes 1.	[62]
zzzz + D	FUN 2 error flag [Read]	If error is found in bit patterns of the output upper limit value[17], the output lower limit value[18], and from [23] to [25] on PIDCL process, PIDCL error[63]=1. Causes of error are set to the error code 2[2]. PID calculation is performed even if error is found. PIDCL error flag[63]=0 if no error. The error code 2[2] is set to nothing.	[63]
zzzz + E	Unused		
zzzz + F	Unused		

■ Error code list

Error code is represented by 4 digits in hexadecimal.



The lower 2 digits represent causes of error.

Represents the loop number.
If here is H00, it is error without relation to the loop number.
If here is from H01 to H04, it is error to the loop of the loop number.

(1) Error code 0

Error code which occurred on PIRIT process and a part of PIDOP process is set to the error code 0.

If there is no error, the previous state is retained.

Error code	Description and cause	Measurement	Remarks
0001	PIDIT was executed again after PIDIT has already ended normally.	Do not execute PIDIT after executing normally.	"PIDIT Normal END [5]" retains the previous value.
0002	The number of loops [3] is 0.	Set the number of loops [3] within a range from 1 to 64.	
0003	The number of loops [3] is 65 or more.	Set the number of loops [3] within a range from 1 to 64.	
0004	PID control table exceeds the maximum No. of the internal output.	Exceeding the maximum No. of the internal output can be avoided by changing the top in the PID control table or the number of loops [3].	You can change the size of PID control table. If the number of loops [3] exceeds the end of I/O, "PIDIT Normal End [5]" retains the previous value.
xx05	Word table for loop xx exceeds the maximum No. of the internal output.	Reset No.[4] of the internal output for loop.	Size of a bit table is 16 bits per loop.
xx06	Bit table for loop xx exceeds the maximum No. of bit.	Reset bit No.[11].	Size of a bit table is 16 bits per loop.
xx07	Output upper limit value[17] for loop xx is outside the range.	Set the output upper limit value[17] within a range from –32767 to 32767.	
xx08	Output lower limit value[18] for loop xx is outside the range.	Set the output lower limit value[18] within a range from –32767 to 32767.	
xx09	Initial value[19] for loop xx is outside the range.	Set the initial value[19] within a range from –32767 to 32767.	
xx0A	The relation of size between Output upper limit value[17], Output lower limit value[18], and Initial value[19] for loop xx is wrong.	Set as follows, Output lower limit value[18]≤Initial value[19]≤Output upper limit value[17]	
xx0B	Set value bit pattern[23] for loop xx is outside the range.	Set the set value bit pattern[23] within a range from 1 to 4.	
xx0C	Measured value bit pattern[24] for loop xx is outside the range.	Set the measured value bit pattern[24] within a range from 1 to 4.	
xx0D	Out value bit pattern[25] for loop xx is outside the range.	Set the out value bit pattern[25] within a range from 1 to 4.	
0020 (N.B.)	Though PIDIT does not end, PIDOP is being executed.	Execute PIDOP after PIDIT is completed normally.	It is set to the error code 0 specified by s of PIDOP(s).
0021 (N.B.)	s of PIDOP(s) is different from s of (1)PID control table PIDIT.	Set the same internal output as s of PIDIT(s) to s of PIDOP(s).	It is set to the error code 0 specified by s of PIDOP(s).

(N.B.) Error code 0020 and 0021 overwrite the error (0001 to xx0D) which occurred before then.
Therefore, please execute always PIDOP after confirming that PIDIT is executed normally.

(2) Error code 1

Error code which occurred on PIDOP process is set to the error code 1. If error is not found, the previous state is retained.

Error code	Description and cause	Measurement	Remarks
0020	Though PIDIT does not end normally, PIDOP is being executed.	Execute PIDOP after PIDIT has been executed normally.	It is set to the error code 0 specified by s of PIDOP(s).
0021	s of PIDOP(s) is different from s of [1]PIDIT(s) in PID control table.	Set No. of the same internal output as s of PIDIT(s) to s of PIDOP(s).	It is set to the error code 0 specified by s of PIDOP(s).
xx22	I/O No.[20] of the set value for loop xx is error.	Set I/O No.[20] of the set value by I/O address coding command.	This error may occur at the rising of an execution flag.
xx23	I/O No.[21] of the measured value for loop xx is error.	Set I/O No.[21] of the measured value by I/O address coding command.	
xx24	I/O No.[22] of the out value for loop xx is error.	Set I/O No.[22] of the out value by I/O address coding command.	
xx25	Sampling time[12] for loop xx is outside the range.	Set the sampling time[12] within a range from 1 to 200.	This error may occur at the rising of an execution flag or PID constant change.
xx26	Sampling time[12] for loop xx is not multiples of the number of loops[3].	Set the sampling time[12] with multiples of the number of loops[3].	
xx27	Proportional gain[13] for loop xx is outside the range.	Set the proportional gain[13] within a range from –1000 to 1000.	
xx28	Integral constant[14] for loop xx is outside the range.	Set the integral constant[14] within a range from 1 to 32767.	
xx29	Derivative constant[15] for loop xx is outside the range.	Set the derivative constant[15] within a range from 1 to 32767.	
xx2A	Derivative delay constant[16] for loop xx is outside the range.	Set the derivative delay constant[16] within a range from 1 to 32767.	
xx30	The relation of size between Output lower limit value[18] and Output upper limit value[17] is error.	Set as follows, Output lower limit value[18]≤Output upper limit value[17].	If S flag[53] is turned ON when PID RUN flag[58] is OFF, this error may occur.
xx31	I/O No.[22] of the out value for loop xx is error.	Set I/O No.[22] of the out value by I/O address coding command.	If S flag or R flag is turned ON when PID RUN flag[58] is OFF, this error may occur.
xx32	Out value bit pattern for loop xx is outside the range.	Set the out value bit pattern within a range from 1 to 4.	

(3) Error code 2

Error code	Description and cause	Measurement	Remarks
0040			(Reserve)
xx41	Set value bit pattern[2P] for loop xx is outside the range.	Set the set value bit pattern[2P] within a range from 1 to 4.	When the bit pattern is outside the range, the process is continued as “4. Not convert”.
xx42	Set value bit pattern[24] for loop xx is outside the range.	Set the set value bit pattern[24] within a range from 1 to 4.	
xx43	Out value bit pattern[25] for loop xx is outside the range.	Set the out value bit pattern[25] within a range from 1 to 4.	
xx44	The relation of size between Output lower limit value[18] and Output upper limit value[17] is error.	Set as follows, Output lower limit value[18]≤Output upper limit value[17].	When the relation of size is error, the process is continued but the output is not carried out.

Name	[FIFO] Initial

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
FIFIT (p, n)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		n is from 0 to 256.
Condition	Time	Condition	Time	
–	1.6	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
p	Top I/O of FIFO											✓						
n	Size of FIFO																	✓

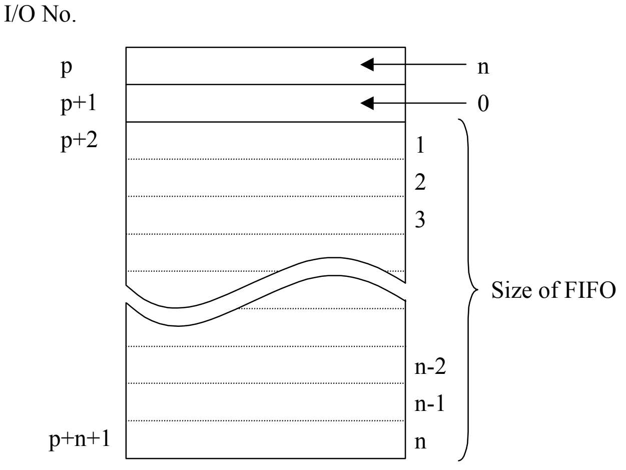
Function

- FIFO stands for First-In First-Out. Data is stored in the buffer and what comes in first is handled first, then processing proceeds sequentially in the same order.

  This command initializes FIFO.
- The top I/O No. P and the size of FIFO are specified.

  Case of 0≤n≤256 : set 'n' to 'p'.

  Case of 257≤n : set 256 to 'p'.
- p+1 is set to 0 which is an initial set value as the number of uses for FIFO.
- FIFO is set n+2 words from p to p+n+1.



Cautionary notes

- p+n+1 should be used within the I/O range. If exceeded, DER=1 and [[Maximum value in the range (the end)]-[p-1]] is set to p.
- n should be set from 0 to 256. If n>256, DER(R7F4)=1 and n is set to 256.

Name	[FIFO] Write

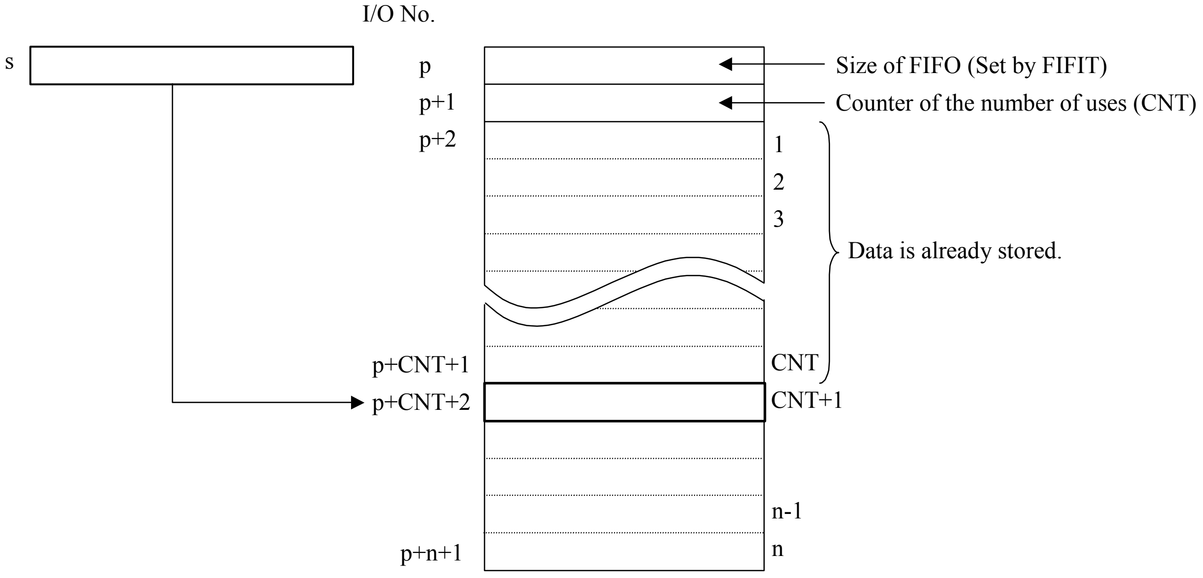
Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
FIFWR (p, s)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		n is from 0 to 256.
Condition	Time	Condition	Time	
–	2	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
p	Top I/O of FIFO											✓						
s	Content to write to FIFO								✓	✓	✓	✓	✓					✓

Function

- Data is written to FIFO buffer of the top I/O number p.
- Case of Counter of the number of uses (CNT) < Size n: the content of s is written to p+CNT+2, and 1 is added to the counter of the number of uses.
- Case of Counter of the number of uses (CNT) ≥ Size n: DER(R7F4) is 1 and it is not written.



Cautionary notes

- p+n+1 should be used within the I/O range. If exceeded, DER=1 and it is not written.

Name	[FIFO] Read

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
FIFRD (p, d)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		n is from 0 to 256.
Condition	Time	Condition	Time	
–	20	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
p	Top I/O of FIFO										✓						
d	I/O to store the read data								✓	✓	✓	✓					✓

Function

- Data is read to FIFO buffer of the top I/O No. p.

  Case of 1 ≤ Counter of the number of uses(CNT) < Size n:

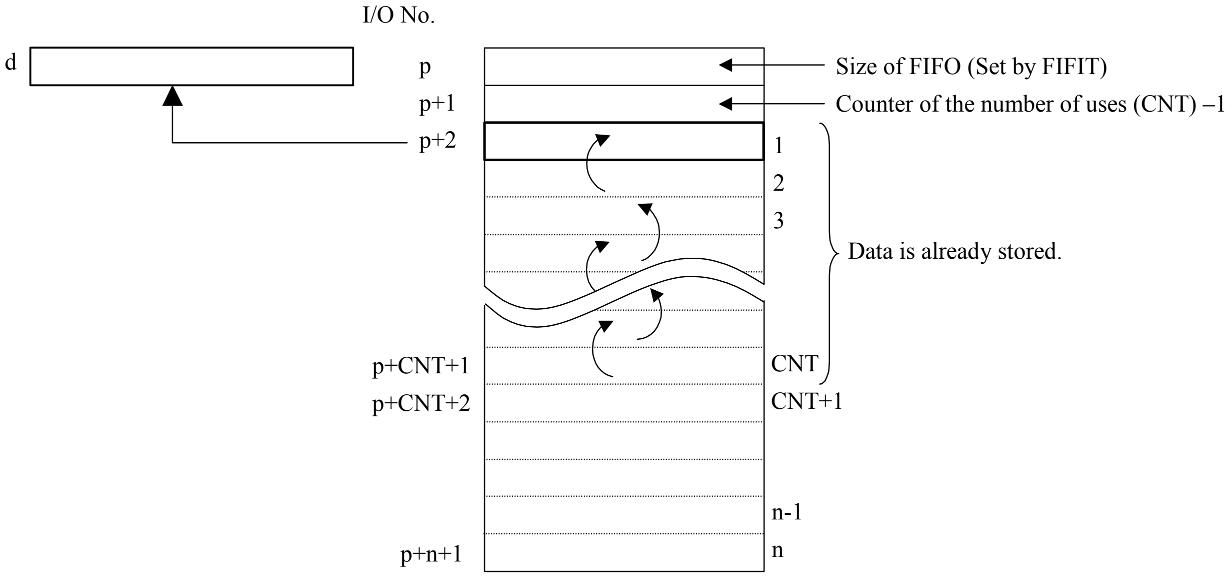
  the content of p+2 which is read is stored in d.

  the content from p+3 to p+CNT+2 is move to the preceding I/O respectively.

  1 is subtracted from the content of CNT.

  Case of Counter of the number of uses (CNT) > Size n, or CNT =0:

  DER(rR7F4) is 1 and it is not read.

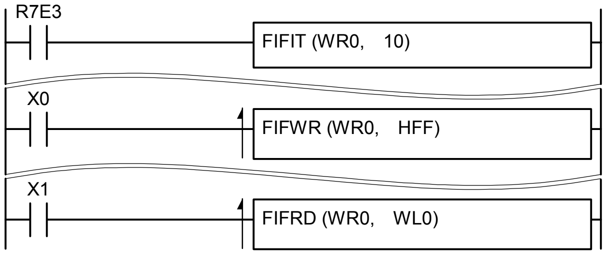


- If data is read, p+CNT+2 stores 0.

Cautionary notes

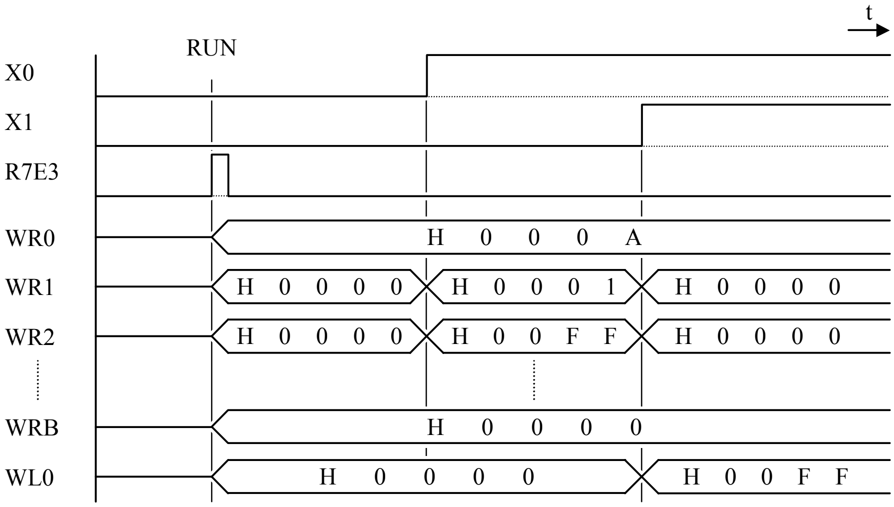
- p+n+1 should be used within the I/O range. If exceeded, DER=1 and it is not read.

Program example



[Program description]

- FIFO buffer is set from WR2 to WRB at the 1st scan after RUN
- HFF is stored at the rising of X0.
- HFF is read to WL0 at the rising of X1.



Name	[Communication support] Creation of check code

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
CCCL (s)	–	3	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		s can be used up to s+6.
Condition	Time	Condition	Time	
–	33.2+1.85n	–	–	

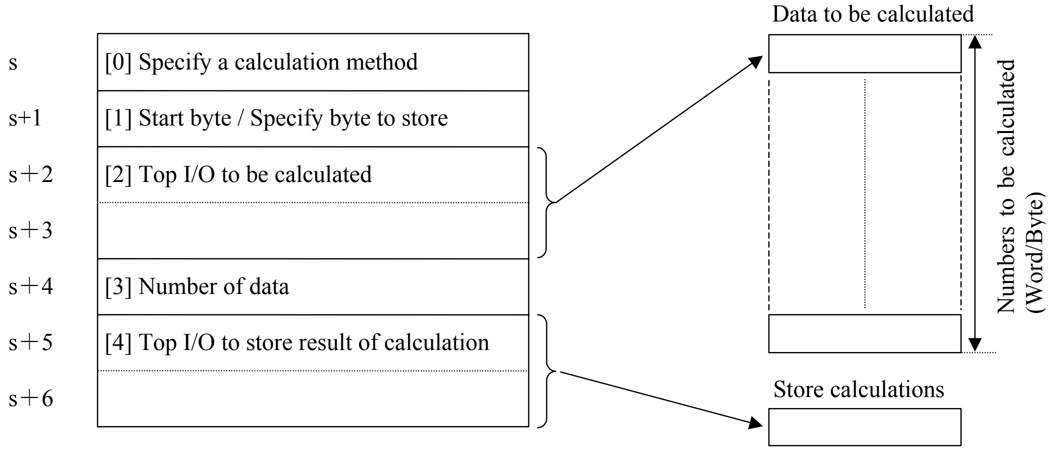
Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	Top I/O in parameter table										✓						✓

Function

As for a general-purpose communication by TRNS 0 and the like, the check code to add to the data frame is created.

Parameter

- 7 words from the word address specified by s are used.



s	[0] Specify a calculation method
s+1	[1] Start byte / Specify byte to store
s+2	[2] Top I/O to be calculated
s+3	
s+4	[3] Number of data
s+5	[4] Top I/O to store result of calculation
s+6	

For [s+2, s+3] and [s+5, s+6], addresses of WR, WL, WM, and WN should be set using I/O address coding command.

[s+0] Specify a

Calculation method:

The calculation method for the check code can be specified from the following 7 kinds.

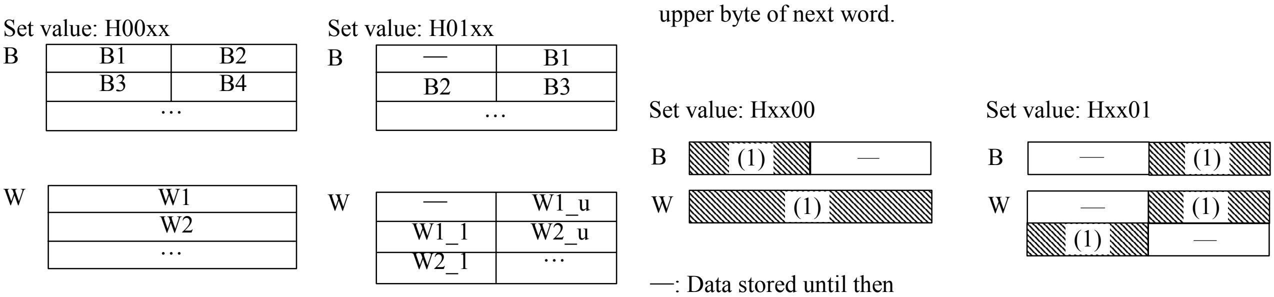
Set value	Calculation expression	Result of calculation	
H0000	(B1)+(B2)+ ··· +(Bn)	Byte	
H0001	(B1)+(B2)+ ··· +(Bn)	Word	Store a found value in the order, upper part then lower part
H0002	(B1)+(B2)+ ··· +(Bn)	Word	Store a found value in the order, lower part then upper part
H0003	(B1)+(B2)+ ··· +(Bn)	Byte	Store a found value in word (in the order, upper then lower) after converted it to ASCII
H0004	(B1)+(B2)+ ··· +(Bn)	Byte	Store a found value in word (in the order, lower then upper) after converted it to ASCII
H0005	(W1)+(W2)+ ··· +(Wn)	Word	Store a found value in the order, upper part then lower part
H0006	(W1)+(W2)+ ··· +(Wn)	Word	Store a found value in the order, lower part then upper part
H0010	{(B1)×or(B2)}×or···×or(Bn)	Byte	
H0011	{(B1)×or(B2)}×or···×or(Bn)	Byte	Store a found value in word (in the order, upper then lower) after converted it to ASCII
H0012	{(B1)×or(B2)}×or···×or(Bn)	Byte	Store a found value in word (in the order, lower then upper) after converted it to ASCII
H0013	{(W1)×or(W2)}×or···×or(Wn)	Word	Store a found value in the order, upper part then lower part
H0014	{(W1)×or(W2)}×or···×or(Wn)	Word	Store a found value in the order, upper part then lower part
H0020	LRC	Byte	
H0021	CRC16	Word	
Others	DATA Error (DER ON)		

[s+1] Specify Start byte / Byte to store:

Only when the check code is calculated in the byte units, the start byte can be specified of either the upper byte or the power byte. Also the byte to store calculations can be specified of the upper byte or the lower byte.

<Upper byte>
Specify the start byte for calculation
H00xx: Start calculation from the upper byte
H01xx: Start calculation from the lower byte
Others: DATA Error (DER ON)

<Lower byte>
Specify the byte to store result of calculation
Hxx00: Set result of calculation to the upper byte
Hxx01: Set result of calculation to the lower byte*
Others: DATA Error (DER ON)
* When value to store is word, the lower byte is stored in the upper byte of next word.



—: Data stored until then
(1): Store after overwriting result of calculation

[s+2, s+3] Top I/O No. to be calculated:

Addresses of WR, WL, WM, and WN should be set using I/O address coding command.

[s+4] Number of data:

Case of byte setting ··· The number of bytes of data is set. (H0001～HFFFF)

Case of word setting ··· The number of words of data is set. (H0001～HFFFF)

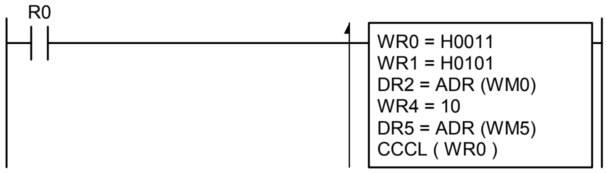
[s+5, s+6] I/O No. to store result of calculation:

Addresses of WR, WL, WM, and WM is set using I/O address coding command.

Cautionary notes

- The internal output used for s parameter table and the area to be calculated should be specified within the I/O number.
- The check code to be calculated is only the internal output of word. If other than the internal output of word is specified, the command is not executed because of DER=1.
- Care must be taken that the area to be calculated does not overlap with s parameter. If overlapped, the command is not executed because of DER=1.
- If I/O other than usable I/O is specified to the area to store calculations, the command is not executed because of DER=1.
- If the area to store calculations overlaps with s parameter table, the command is executed because of DER=1.

Program example



```
WR0 = H0011
WR1 = H0101
DR2 = ADR (WM0)
WR4 = 10
DR5 = ADR (WM5)
CCCL ( WR0 )
```

[Program description]

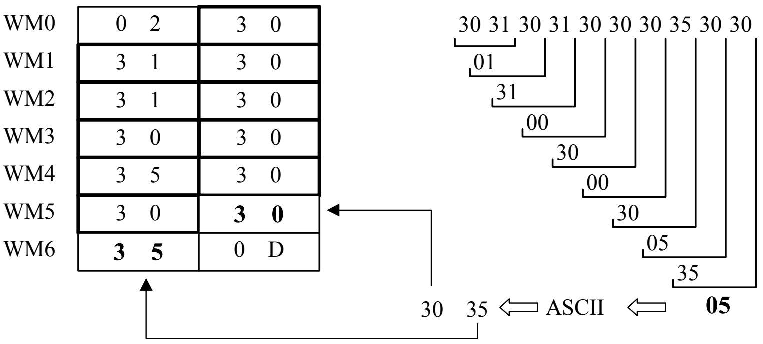
- CCCL command is executed by setting a parameter for check code calculation at the rising edge of R0.
- Constitution of sending frame   The check code is converted to ASCII after XOR is operated at every byte.

STX	Data	C.C.	CR
(02)	(30313031303030353030)	(?)	(0D)

- When the sending data area is the data composition as follows, suppose that R0 was turned ON.

WM0	0 2	3 0
WM1	3 1	3 0
WM2	3 1	3 0
WM3	3 0	3 0
WM4	3 5	3 0
WM5	3 0	? ?
WM6	? ?	0 D

- If a sample program is executed, the results is as follows.



WM0	0 2	3 0
WM1	3 1	3 0
WM2	3 1	3 0
WM3	3 0	3 0
WM4	3 5	3 0
WM5	3 0	**3 0**
WM6	**3 5**	0 D

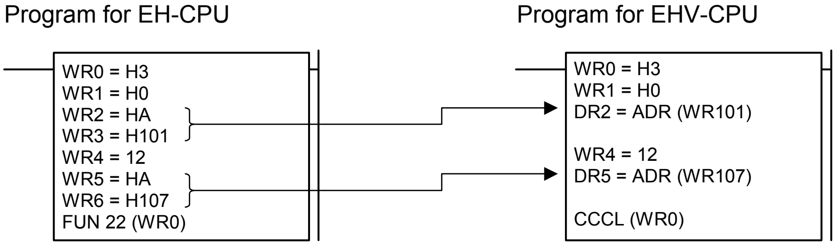
PRN ➔ PRJ

This command is equivalent to FUN 22 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 22 (s) for EHV, how to convert is as follows.

FUN 22 (s) ➔ CCCL (s)

* If converted by a conversion tool, it is converted as mentioned above. However you need to modify a part of s parameter.



Program for EH-CPU

```
WR0 = H3
WR1 = H0
WR2 = HA   }
WR3 = H101 }
WR4 = 12
WR5 = HA   }
WR6 = H107 }
FUN 22 (WR0)
```

Program for EHV-CPU

```
WR0 = H3
WR1 = H0
DR2 = ADR (WR101)

WR4 = 12
DR5 = ADR (WR107)

CCCL (WR0)
```

Name	[Communication support] Collation of check code

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
CCCMP (d, s)	–	4	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		・s can be used up to s+6. ・d can be used up to d+2.
Condition	Time	Condition	Time	
–	43.2+1.85n	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	I/O to store result of collation											✓						
s	Top I/O in parameter table											✓						✓

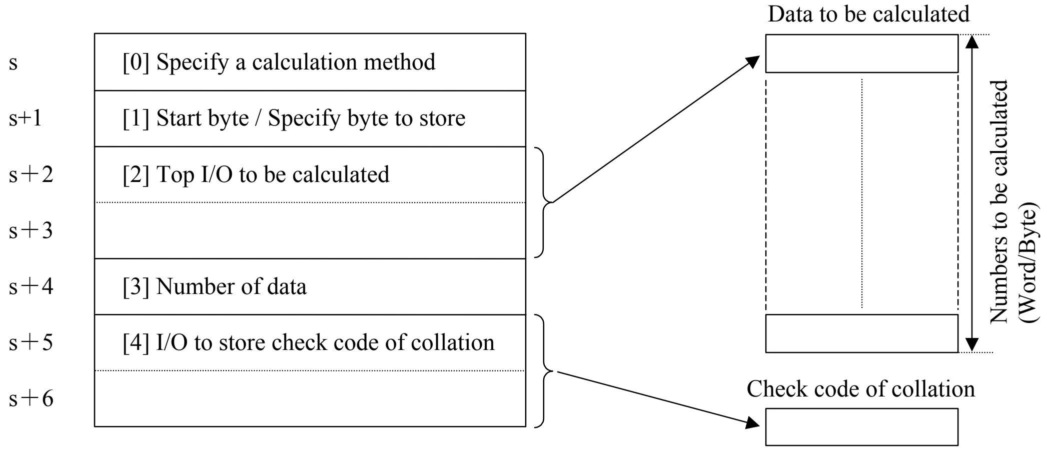
Function

In a general-purpose communication by TRNS 0 and the like, a check code calculated from the data frame is collated with a check code added to the data frame.

Parameter

d : Result of collation is stored. 3 words from the specified word I/O are used.

s : Table for the parameter to calculate the check code. 7 words from the specified word I/O are used.



s	[0] Specify a calculation method
s+1	[1] Start byte / Specify byte to store
s+2	[2] Top I/O to be calculated
s+3	
s+4	[3] Number of data
s+5	[4] I/O to store check code of collation
s+6	

d	[0] Result of collation
d+1	[1] Result of calculation
d+2	[2] Result of calculation

For [s+2, s+3] and [s+5, s+6], addresses of WR, WL, WM, and WN should be set using I/O address coding command.

[s+0] Specify a calculation method:

The calculation method of the check code can be specified form the following 7 kinds.

Set value	Calculation expression	Result of calculation	
H0000	(B1)+(B2)+ ··· +(Bn)	Byte	
H0001	(B1)+(B2)+ ··· +(Bn)	Word	Store the found value in the order, upper part then lower part
H0002	(B1)+(B2)+ ··· +(Bn)	Word	Store the found value in the order, lower part then upper part
H0003	(B1)+(B2)+ ··· +(Bn)	Byte	Store the found value in word (in the order, upper then lower) after converted it to ASCII
H0004	(B1)+(B2)+ ··· +(Bn)	Byte	Store the found value in word (in the order, lower then upper) after converted it to ASCII
H0005	(W1)+(W2)+ ··· +(Wn)	Word	Store the found value in the order, upper part then lower part
H0006	(W1)+(W2)+ ··· +(Wn)	Word	Store the found value in the order, lower part then upper part
H0010	{(B1)×or(B2)}×or···×or(Bn)	Byte	
H0011	{(B1)×or(B2)}×or···×or(Bn)	Byte	Store the found value in word (in the order, upper then lower) after converted it to ASCII
H0012	{(B1)×or(B2)}×or···×or(Bn)	Byte	Store the found value in word (in the order, lower then upper) after converted it to ASCII
H0013	{(W1)×or(W2)}×or···×or(Wn)	Word	Store the found value in the order, upper part then lower part
H0014	{(W1)×or(W2)}×or···×or(Wn)	Word	Store the found value in the order, upper part then lower part
H0020	LRC	Byte	
H0021	CRC16	Word	
Others	DATA Error (DER ON)		

[s+1] Specify Start byte for calculation / Start byte for collation:

Only when the check code is calculated in the byte units, the start byte can be specified to either the upper byte or the lower byte. Also the start byte for collation can be specified to either the upper byte or the lower byte.

<Upper byte>
Specify start byte for calculation
H00xx: Start calculation from the upper byte
H01xx: Start calculation from the lower byte
Others: DATA Error (DER ON)

Set value: H00xx

B	B1	B2
	B3	B4
	···	
W	W1	
	W2	
	···	

Set value: H01xx

B	—	B1
	B2	B3
	···	
W	—	W1_u
	W1_1	W2_u
	W2_1	···

<Lower byte>
Specify start byte for collation
Hxx00: Set result of calculation to the upper byte
Hxx01: Ste result of calculation to the lower byte*
Others: DATA Error (DER ON)
* When value to store is word, the lower byte is store in the upper byte of next word.

Set value: Hxx00

B	▨ (1) ▨	—
W	▨ (1) ▨	

Set value: Hxx01

B	—	▨ (1) ▨
W	—	▨ (1) ▨
	▨ (1) ▨	—

—: Data to be stored until then
(1): Store overwriting result of calculation

[s+2, s+3] Classification of the top I/O to be calculated:

Addresses of WR, WL, WM, and WN should be set I/O address coding command.

[s+4] Number of data:

Case of byte setting ··· Set the number of bytes of data. (H0001 - HFFFF)

Case of word setting ··· Set the number of words of data. (H0001 - HFFFF)

[s+5, s+6] Classification of I/O to store result of calculation:

Addresses of WR< WL, WM, and WN should be set using I/O address coding command.

[d+0] Result of collation:

OK—H8000, NG—H80FF

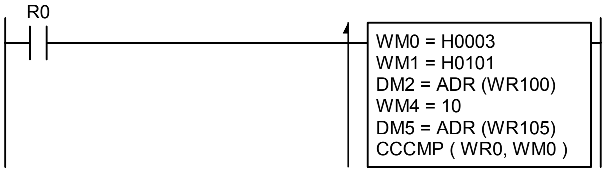
[d+1, d+2] Result of collation:

The check code calculated actually is stored. When the check code to collate extends over two words, a format of a result of this calculation also extends over two words.

Cautionary notes

- The internal output used for s parameter table, the internal output used for d parameter table, and the area to be calculated should be specified within the range of I/O numbers.
- The check code to be calculated is only the word internal output. If other than the word internal output are specified, the command is not executed because of DER=1.
- Care must be taken that the area to be calculation does not overlap with d and s parameters.
  If the area overlaps, the command is not executed because of DER=1.

Program example



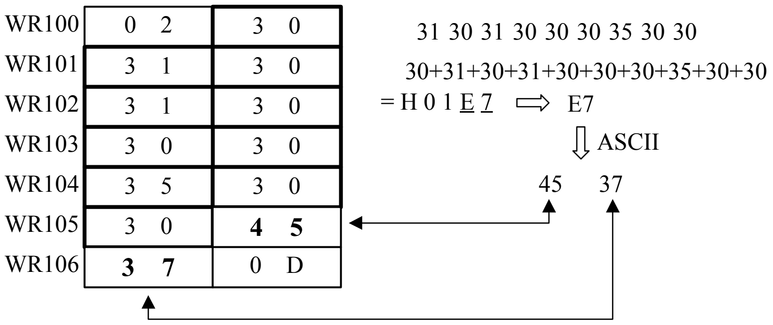
```
WM0 = H0003
WM1 = H0101
DM2 = ADR (WR100)
WM4 = 10
DM5 = ADR (WR105)
CCCMP ( WR0, WM0 )
```

[Program description]

- The parameter for collating check code is set and the CCCMP command is executed at the rising edge of R0.
- When the receiving data area is the data composition as follows, suppose that R0 was turned ON.

WR100	0 2	3 0
WR101	3 1	3 0
WR102	3 1	3 0
WR103	3 0	3 0
WR104	3 5	3 0
WR105	3 0	4 5
WR106	3 7	0 D

- If a sample program is executed, the result is as follows.



WR100	0 2	3 0
WR101	3 1	3 0
WR102	3 1	3 0
WR103	3 0	3 0
WR104	3 5	3 0
WR105	3 0	**4 5**
WR106	**3 7**	0 D

- WR0=H8000 since the check code is matching. (If it is not matching, WR0=H80FF.)

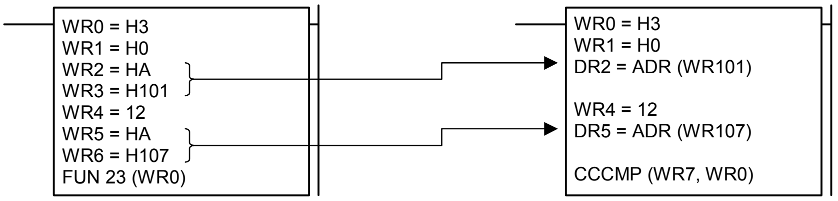
PRN ➔ PRJ

This command is equivalent to FUN 23 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 22 (s) for EHV, how to convert is as follows.

FUN 22 (s) ➔ CCCL (s+7, s)

* If converted by a conversion tool, it is converted as mentioned above. However you need to modify a part of s parameter.



Program for EH-CPU

```
WR0 = H3
WR1 = H0
WR2 = HA
WR3 = H101
WR4 = 12
WR5 = HA
WR6 = H107
FUN 23 (WR0)
```

Program for EHV-CPU

```
WR0 = H3
WR1 = H0
DR2 = ADR (WR101)

WR4 = 12
DR5 = ADR (WR107)

CCCMP (WR7, WR0)
```

Name	[Others] Process stepping

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
IFR (s)	–	3	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		s can be used up to s+6.
Condition	Time	Condition	Time	
–	24	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	Top I/O in parameter table										✓							✓

Function

This is a process stepping command (Sequential control command).

Since a set input and a reset input can be specified toward one point of bit I/O, the process stepping program can be realized with a regular format by combining.

Parameter

7 words from the word I/O specified by s are used.

s	[0] Set input I/O
s+1	
s+2	[1] Process startup I/O
s+3	
s+4	[2] Reset input I/O
s+5	
s+6	Reserve (used by system)

[0] Set input I/O

I/O to turn ON a process startup I/O should be specified.

[1] Process start-up I/O

I/O to be a startup condition the process should be specified.

If [0] set input I/O turns ON, this I/O turns ON. And if [2] reset input I/O turns ON, this I/O turns OFF.

[2] Reset input I/O

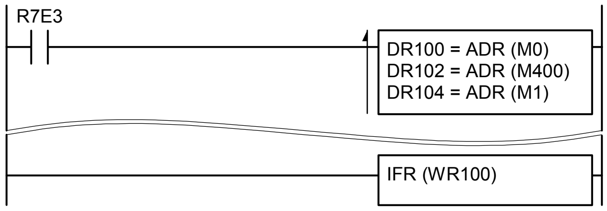
I/O to turn OFF a process startup I/O should be specified.

For I/Os of [0] to [2], addresses of R, L, and M should be specified using I/O address coding command.

Cautionary notes

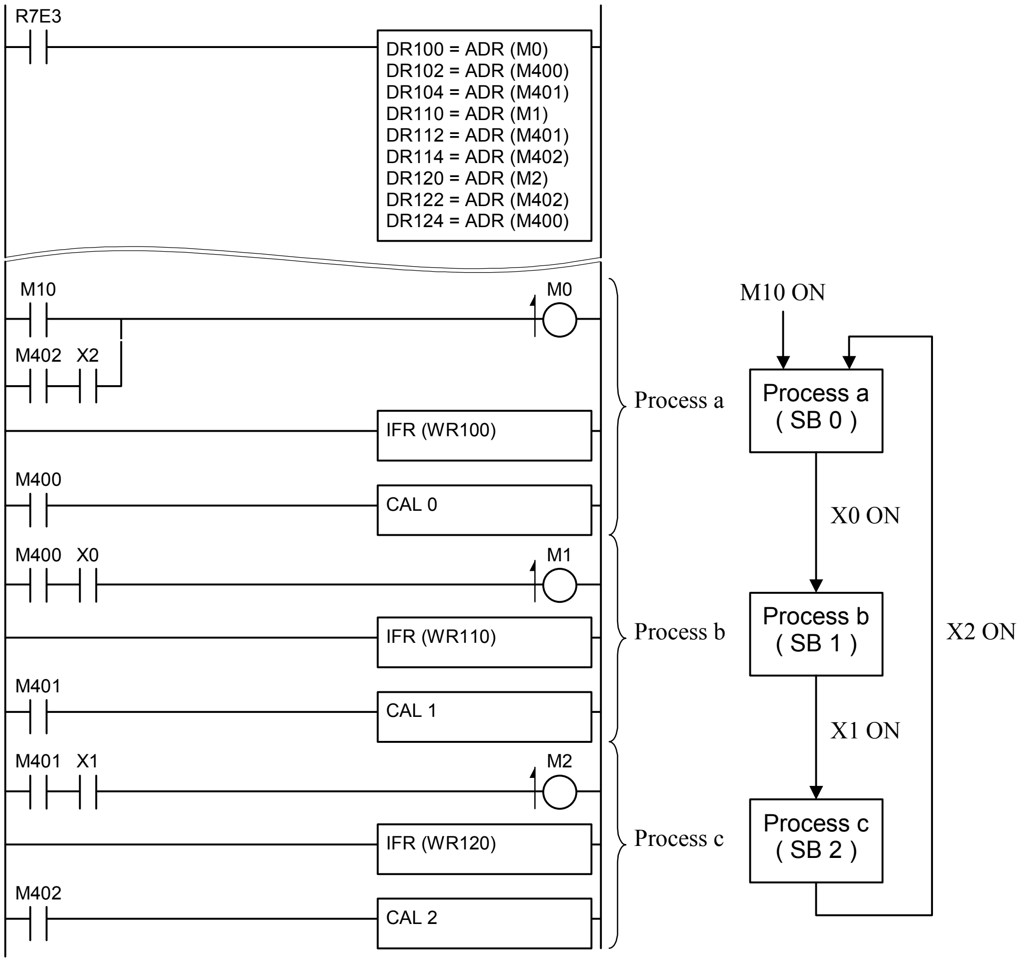
- When s and s+1 (for the set input), and s+4 and s+5 (for the reset input) are turned ON, s+2 (for the reset input) is given priority.
- When the areas specified by s to s+6 are overlapping, and I/O specified by s to s+5 is outside the range, the process is not performed because of DER=1.
- In each bit I/O specified by s parameter, the same I/O should not be specified.

Program example



[Program description]

- The set input (M0), the process startup input (M400), and the reset input (M1) are set at the 1st scan after RUN.
- If M0 is turned ON, M400 is turned ON, and if M1 is turned ON, M400 is turned OFF since IFR (s) is always running.



- The sequential control is possible by writing several IFR(s)s, and by the reset input of the previous process being the set input of the next process.

PRN ➔ PRJ

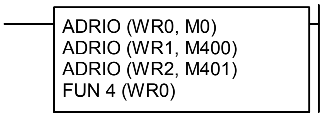
This command is equivalent to FUN 4 (s) in the program (PRN file) of the EH-CPU.

When changing the program which has used FUN 4 (s) for EHV, how to convert is as follows.

FUN 4 (s) ➔ IFR (s)

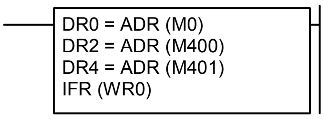
* If converted by a conversion tool, it is converted as follows. However, you need to modify a part of s parameter.

Program for EH-CPU



```
ADRIO (WR0, M0)
ADRIO (WR1, M400)
ADRIO (WR2, M401)
FUN 4 (WR0)
```

Program for EHV-CPU



```
DR0 = ADR (M0)
DR2 = ADR (M400)
DR4 = ADR (M401)
IFR (WR0)
```

[Caution on converting for program]

Although s parameter is 3 words in the program of EH-CPU, s parameter needs 6 words in EHV. When converting a program, please make sure that s parameter area for incremental words is not being used for other purposes.

Name	[Others] Dynamic scan pulse

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
PGEN (s)	–	3	↕	●	●	●	●

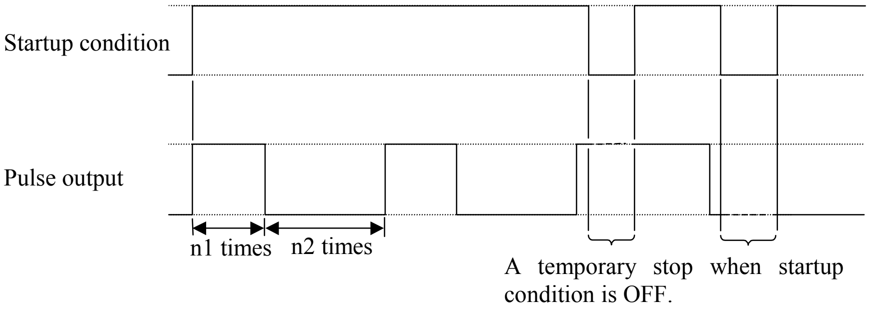
Command processing time (μs)				Remarks
Average		Maximum		s can be used up to s+6.
Condition	Time	Condition	Time	
–	3.6	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	Top I/O in parameter table										✓						✓

Function

• Pulse is created by switching I/O by the number of times which executed this command.
(When only one is written on the program, the pulse depends on the number of scans.)

• If a startup condition is turned OFF after a command is executed once, the state of then is retained. If a startup condition is turned ON again, the action is restarted from the previous state.



Parameter

6 words from the word address specified by s are used.

Parameter	Description	Details
s	Number of times of scan to turn ON	Specify the number of times of scan by which pulse is turned ON. Ex.) If set to 5, pulses for 5 scans are turned ON.
s+1	Number of times of scan to turn OFF	Specify the number of times of scan by which pulse is turned OFF. Ex.) If set to 11, pulses for 11 scans are turned OFF.
s+2, s+3	I/O number of pulse output	Specify the bit I/O number to output the pulse. Address is specified using I/O address coding command.
s+4	Number of times of progress of scan to turn ON	Display the number of times of scan which has passed after the pulse is turned ON. Set by the system.
s+5	Number of times of progress of scan to turn OFF	Display the number of times of scan which has passed after the pulse is turned OFF. Set by the system.

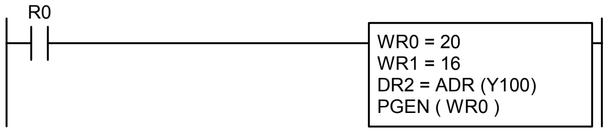
[Reference]

Number of scans		Action of pulse
n1	n2	
n1 = 0	n2 = 0	Turned OFF
	n2≧1	
n1≧1	n2 = 0	Turned ON
	n2≧1	n1 scan is turned ON and n2 scan is turned OFF.

Cautionary notes

- The internal output used for s parameter should be specified within the range of I/O number.
- A program to clear s+4 and s+5 in order to set 0 should be created before this command is executed. (If not cleared, the pulse width of first cycle may not be a set value.)
- When a startup condition is turned OFF, values of the output, s+4, and s+5 are retained.
- Detection for ON and OFF of a startup condition may be delayed for one scan because of switching the output of the specified pulse at the execution of the command.
- If I/O other than bit is specified by s+2 and s+3, the command is not executed because of DER=1.

Program example



```
WR0 = 20
WR1 = 16
DR2 = ADR (Y100)
PGEN ( WR0 )
```

[Program description]

- If R0 is turned ON, the pulse for which 20-scan hours are turned ON and 16-scan hours are turned OFF is output from Y100.
- The progress value of ON is stored in WR4 and the progress value of OFF is stored in WR5.
- If R0 is turned OFF, the output at that time is retained.

PRN ➔ PRJ

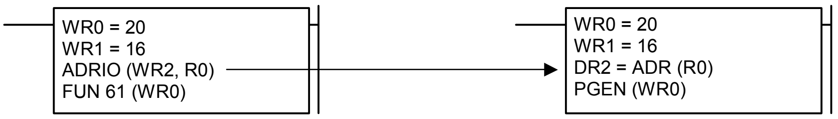
This command is equivalent to FUN 61 (s) in the program (PRN file) of EH-CPU.

When changing the program which has used FUN 61 (s) for EHV, how to convert is as follows.

FUN 61 (s) ➔ PGEN (s)

* If converted by a conversion tool, it is converted as mentioned above. However, you need to modify a part of s parameter.

Program for EH-CPU Program for EHV-CPU



```
WR0 = 20
WR1 = 16
ADRIO (WR2, R0)
FUN 61 (WR0)
```

```
WR0 = 20
WR1 = 16
DR2 = ADR (R0)
PGEN (WR0)
```

[Caution on converting for program]

Although s parameter is 5 words in the program of EH-CPU, s parameter needs 6 words in EHV. When converting a program, please make sure that s parameter area for incremental words is not being used for other purpose. And when a program refers the number of times of progress of scan to turn ON an OFF, please shift the referring internal output one word to the right or left.

[1] Basic commands

[2] Arithmetic commands

[3] Application commands

[4] Control commands

[5] CPU serial communications commands

[6] High-function module transfer commands

Name	Termination of normal scan

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
END	−	2	●	●	●	●	●

Command professing time (µs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
−	0.26	−	−	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT / TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
−	(No argument)																

Function

- The termination of a normal scan program is indicated. This command can execute a normal scan by returning to the top of the normal scan program.
- If there are neither a subroutine program nor an interrupt scan program, this command is unnecessary.
- If there are both a subroutine program and an interrupt scam program, this command is written at the end of a normal scan.
- This command can be used on the program only once. Do not put in a startup condition.

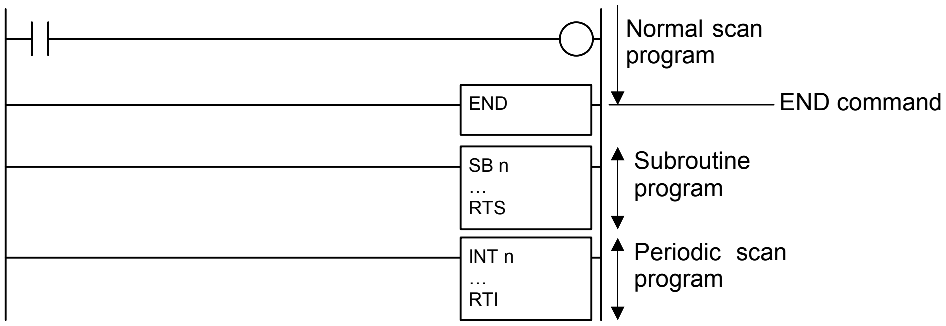
Cautionary notes

Unlike EH-CPU currently in use, the editor detects an assembling error. Note because the program including the cause of the assembling error cannot be written into EHV-CPU.

[Reference: Causes of the assembling error]

No.	Description of error
1	There is no END command.
2	There are two END commands or more.
3	A startup condition is in END command.

Program example



Name	Termination of normal scan (with conditions)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
CEND (s)	—	3	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
Incompletion	0.44	—	—	
Completion	0.46	—	—	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	Condition for scan terminating	✓	✓		✓													

Function

- When a condition for normal scan terminating (s) is ON, this command can execute a normal scan by returning to the top of the normal scan program.
- When s is OFF, the next command is executed.
- This command can be used only on the normal scan program and used many times.
- This command can set a startup condition. In this case, this command is executed when both s and the condition are On.

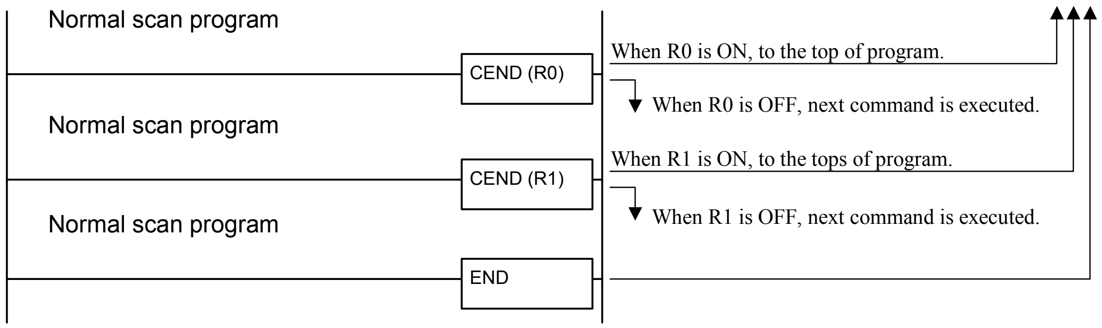
Cautionary notes

- The extension XY cannot be used for the condition for scan terminating.
- Unlike EH-CPU currently in use, the editor detects an assembling error. Note because the program including the cause of the assembling error is written into EHV-CPU.

[Reference: Cause of the assembling error]

No.	Description of error
1	CEND command is behind END command.

Program example


Normal scan program
CEND (R0)
When R0 is ON, to the top of program.
When R0 is OFF, next command is executed.
Normal scan program
CEND (R1)
When R1 is ON, to the tops of program.
When R1 is OFF, next command is executed.
Normal scan program
END

Name	Unconditional jump

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
JMP n	–	3	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	0.4+0.54n	–	–	Code No. 0 to 511 can be used. (Decimal)

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
n	Code No.																	✓

Function

- When a startup condition of JMP n is turned ON, the program is jumped from this command to LBL n of which the code No. is the same as this command.
  JMP n and LBL n should be always used in pairs.
- When a startup condition is incomplete, the next command is executed.
- When this command is put into an arithmetic box simultaneously with other command, this command should be put at the end of the box.
- JMP n command is valid on only the same scan program. (Jumping from the normal scan to the subroutine or the interrupt scan is impossible and the opposite jumping is also impossible.)
- Although nesting of JMP n command is possible, care must be taken so that a jam error does not occur.

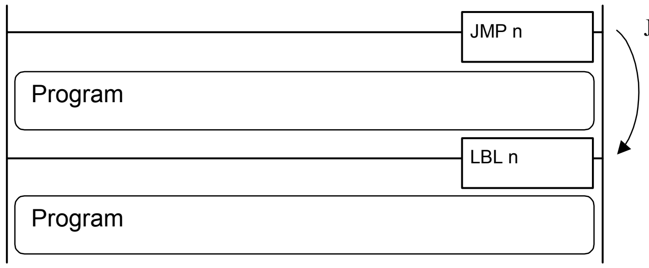
Cautionary notes

- If there is a timer in the jumped program, the output is not turned ON even if the conditions for ON are fulfilled since the progress value is updated but the command is not executed.
- Unlike EH-CPU currently in use, the editor detects an assembling error. Note because the program including the cause of the assembling error cannot be written into EHV-CPU.

[Reference: Causes of assembling error]

No.	Description of error
1	There is no LBL n.
2	It is going to jump to other program areas.

Program example



Jump to LBL n when a startup condition is turned ON.

Name	Jump with conditions

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
CJMP n (s)	–	4	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		Code No. 0 to 511 can be used. (Decimal)
Condition	Time	Condition	Time	
Incompletion	0.38	–	–	
Completion	0.96+0.54n	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
n	Code No.																✓
s	Condition for jumping	✓	✓		✓												

Function

- When a condition for jumping s of CJMP n (s) is turned ON, the program jumps from this command to LBL n of which the code No. is the same as this command. CJMP n (s) and LBL n should be always use in pairs.
- When a startup condition and a condition for jumping are incomplete, the next command is executed.
- When this command is put in an arithmetic box simultaneously with other commands, care must be taken because the program jumps without performing remaining operations in the box if conditions are complete.
- CJMP n (s) command is valid on only the same scan program. (Jumping from the normal scan to the subroutine or the interrupt scan is impossible, and the opposite jumping is also impossible.
- Although nesting of CJMP n (s) command is possible, care must be taken so that a jam error does not occur.

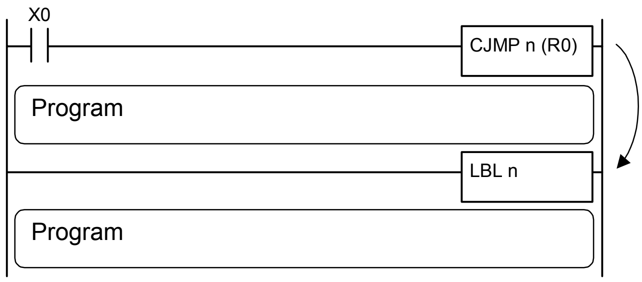
Cautionary notes

- If there is a timer in the jumped program, the output is not turned ON even if the conditions for ON are fulfilled since the progress value is updated but the command is not executed.
- Unlike EH-CPU currently in use, the editor detects an assembling error. Note because the program including the cause of the assembling error cannot be written into EHV-CPU.

[Reference: Causes of assembling error]

No.	Description of error
1	There is no LBL n.
2	It is going to jump to other program areas.

Program example



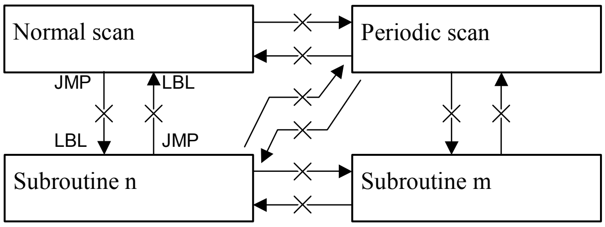
Jump to LBL n when a startup condition and a condition for jumping are both ON.

In the left example, jump to LBL n when X0 and R0 are both ON.

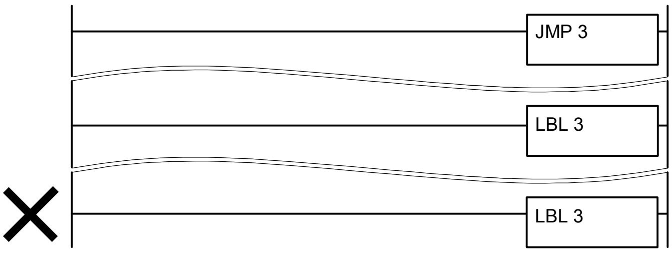
Reference Grammar of JMP and CJMP

[1] LBL n of which the code No. is the same as the code No. of JMP command. is needed.

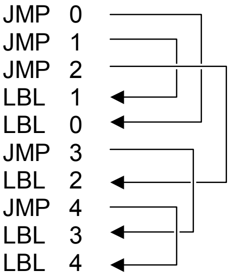
[2] Jumping to the area other than areas which have JMP command is impossible.



[3] LBL n of which the code No. is the same as the code No. of JMP command must not be repeated.

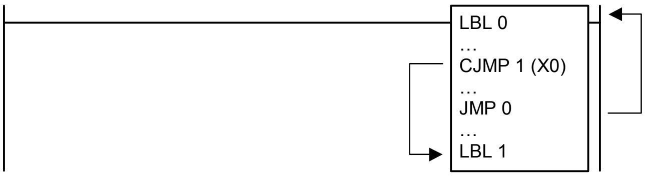


[4] Nesting of JMP command is possible.



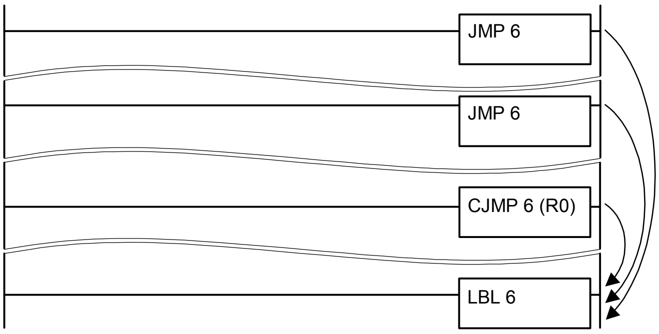
```
JMP  0
JMP  1
JMP  2
LBL  1
LBL  0
JMP  3
LBL  2
JMP  4
LBL  3
LBL  4
```

[5] JMP command can jump forward of this command.



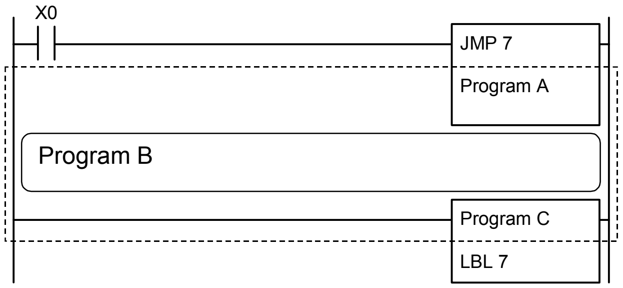
- JMP 0 jumps to the forward LBL 0.
- If the input X0 is turned ON, jumping from CJMP 1 to LBL 1 can get out of the loop from JMP 0 to LBL 0.
- If there is no command to get out of the loop like CJMP 1, the loop from JMP0 to LBL 0 repeats limitlessly.

[6] JMP command of the same code No. can be repeated.



[7] A startup condition can be programmed to JMP command.

Startup condition

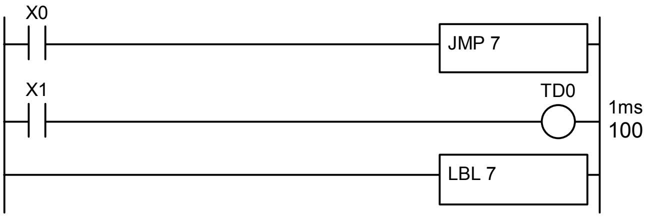


If jumping from JMP 0 to LBL 0, the program A, B, and C are not executed.

[8] CJMP command also obeys the grammar from [1] to [7].

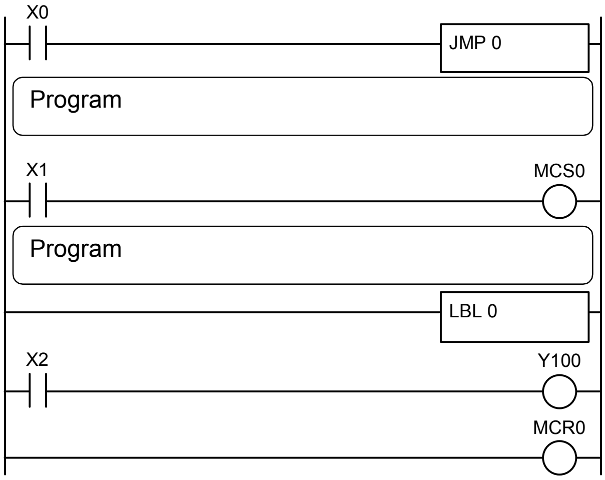
Cautionary notes

- EHV-CPU updates the progress value at the command execution of a timer. The timer may not be turned ON correctly if a program not to scan a portion to execute the timer command is created after the timer is started up.



If X0 is turned ON after X1 is turned ON, a progress value of TD0 is updated even if jumping from JMP1 to LBL1. If X0 is keeping ON, TD0 is not turned ON even if the progress value of TD0 exceeds 100.

- Please program with great care since the action is as follows if using by combining JMP command with MCS and MCR.



When not jumping on JMP 0, Y100 is turned ON when X1 is ON and X2 is ON.

When jumping in JMP 0, Y100 sis turned ON if X2 is ON.

- Do not create a circuit jumped out from between MCS and MCR.

Name	Label

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
LBL n	–	1	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		Code No. 0 to 511 can be used. (Decimal)
Condition	Time	Condition	Time	
–	0.02	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
n	Code No.																	✓

Function

- This command indicates where to jump when JMP n and CJMP n are executed. (n is always used in pairs.)
- n in LBL n cannot be used on the same program repeatedly.
- Nothing is performed by this command itself.
- Even if putting a startup condition in LBL n, it is ignored.

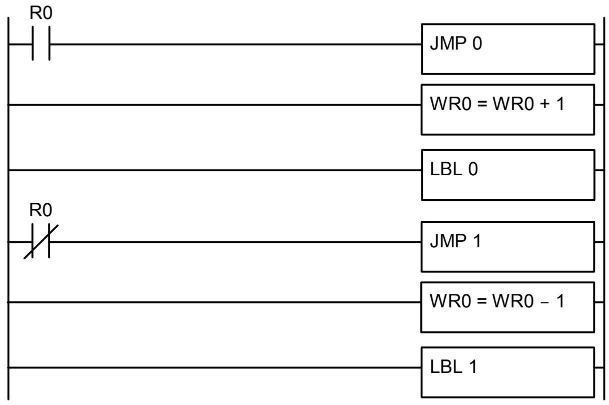
Cautionary notes

Unlike EH-CPU currently in use, the editor detects an assembling error. Note because the program including the cause of the assembling error cannot be written into EHV-CPU.

[Reference: Cause of assembling error]

No.	Description of error
1	There is LBL of the same No.

Program example



[Program description]

- When R100 is ON, JMP 0 is executed but JMP 1 is not executed. Therefore, the content of WR0 decreases one by one per 1 scan.
- When R100 is OFF, JMP 0 is not executed but JMP 1 is executed. Therefore, the content of WR0 increases one by one per 1 scan.

Name	FOR

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
FOR n (s)	—	4	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		Code No.0 to 99 can be used. (Decimal)
Condition	Time	Condition	Time	
—	1.14	—	1.42	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
n	Code No.																✓
s	Number of repeating times								✓	✓	✓						

Function

- It jumps from NEXT n of the same code No. to this command.
- When s, the number of repeating times, is larger than 0, the next command of FOR n (s) is executed.
- When s, the number of repeating times, is 0, it jumps to the next command of NEXT n.
- FOR n (s) and NEXT n should be always used in pairs. And NEXT n should be put in the back of FOR n.
- FOR n (s) cannot be used repeatedly.
- FOR n (s) and NEXT n should be used on the same program area.
  (FOR n (s) cannot be programmed on the normal scan and NEXT n cannot be programmed on the subroutine area.)
- It can be nested up to 5 times from FOR n (s) to NEXT n.

Cautionary notes

Unlike EH-CPU currently in use, the editor detects an assembling error. Note because the program including causes of the assembling error cannot be written into EHV-CPU.

[Reference: Causes of assembling error]

No.	Description of error
1	There is FOR of the same No.
2	NEXT of No. corresponding has not been defined.
3	NEXT is before FOR.
4	Area error of NEXT
5	Nesting error from FOR to NEXT
6	FOR nesting over flow

Program example

See the description pages of “NEXT n”.

Name	NEXT

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C
NEXT n	—	3	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		Code No.0 to 99 can be used. (Decimal)
Condition	Time	Condition	Time	
—	0.3	—	3.38	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
n	Code No.																	✓

Function

It subtrats 1 from s, the number of repeating times, of FOR s (s) of the same No., and then it jumpes to FOR n (s).

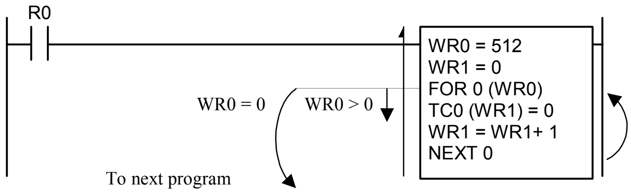
Cautionary notes

Unlike EH-CPU currently in use, the editor detects an assembling error. Note because the program including causes of the assembling error cannot be written into EHV-CPU.

[Reference: Causes of assembling error]

No.	Description of error
1	There is NEXT of the same No.
2	FOR of No. corresponding has not been defined.
3	FOR nesting over flow

Program example



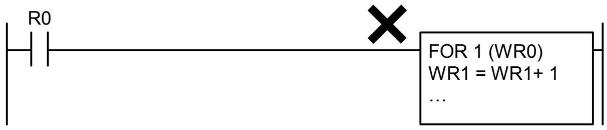
```
WR0 = 512
WR1 = 0
FOR 0 (WR0)
TC0 (WR1) = 0
WR1 = WR1+ 1
NEXT 0
```

[Program description]

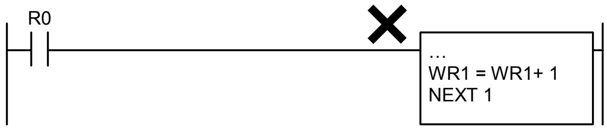
- If R0 is turned ON, all data in 512 points of the progress value (TC n) of a timer counter is erased and set to 0.
- FOR to NEXT, the command keeps on being executed until s becomes 0, once it starts up.
- As for FOR 0 (WR0), when WR>0, commands following ‘TC0(WR1)=0’ are executed, and it jumps to FOR 0 (WR0) after subtracting 1 from WR0 in NEXT 0.
- As for FOR 0 (WR0), when WR0 = 0, it jumps to the next command of this box.

Reference	Grammar of FOR and NEXT

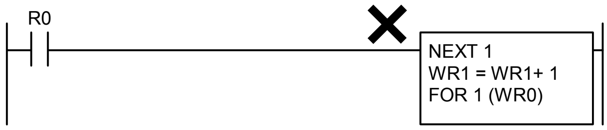
[1] NEXT command of the same code No. as the code No. of FOR command is needed after FOR.



There is not NEXT command to FOR command on the user program.



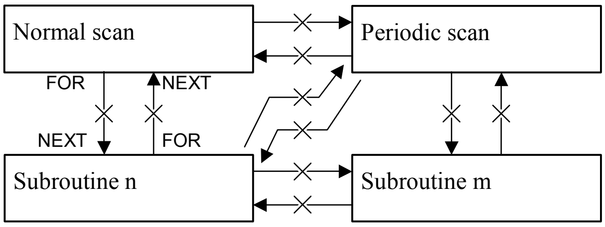
There is no FOR command before NEXT command.



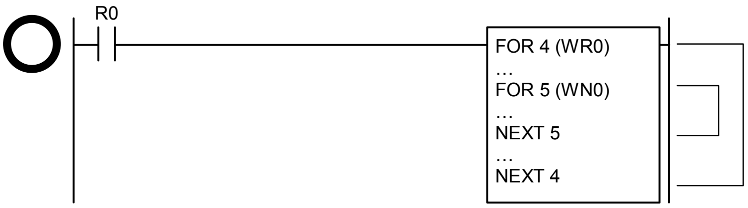
There is NEXT command before FOR command.

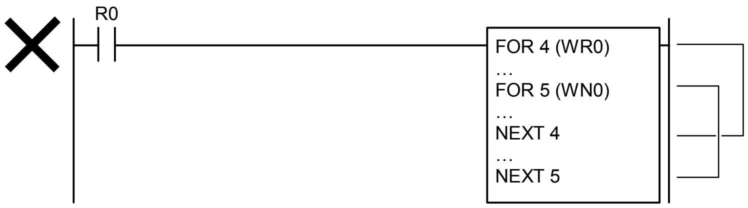
[2] FOR and NEXT commands of the same code No. cannot be repeated.

[3] FOR and NEXT commands have to be in the same area.

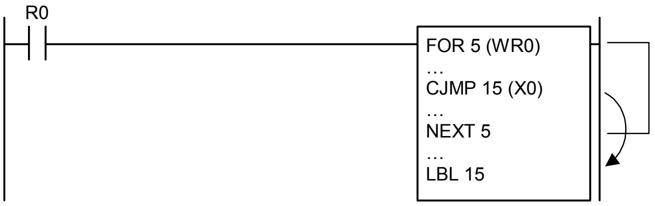


[4] The nesting structure of FOR - NEXT should be set.



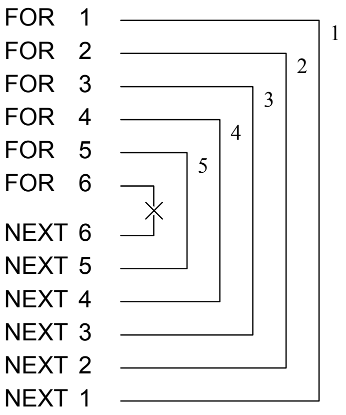


[5] It is possible to get out of a loop of FOR – NEXT by the jump command.

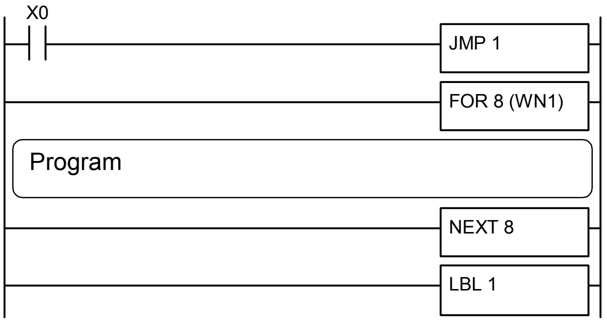


If X0 is turned ON without performing a loop of FOR 5 to NEXT 5 the number of times of repetition (the content of WR0), it gets out of a loop.

[7] It can be nested up to 5 times from FOR to NEXT. If it contains a subroutine, FOR to NEXT in the subroutine is counted.



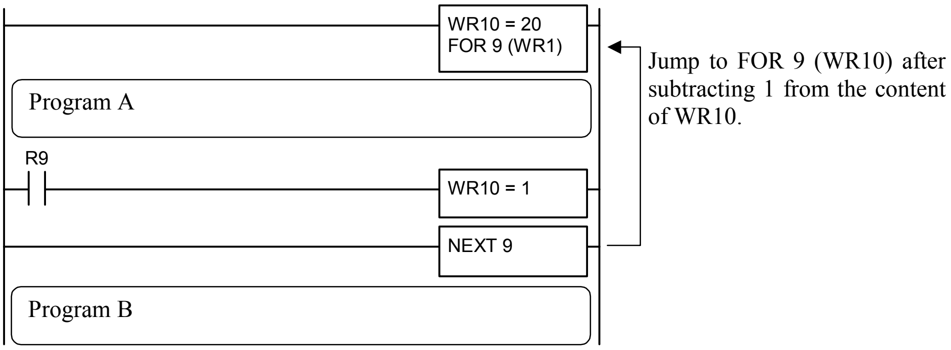
[8] Do not put a startup condition in FOR to NEXT. If the startup condition is needed, the following circuit should be created.



When X0 is OFF, … the program equal to the value of WN1 is executed.

When X0 is ON, … the program is not executed because of jumping from JMP 1 to LBL 1.

[9] Changing the number of times of repetition on the program is possible.


WR10 = 20
FOR 9 (WR1)
Program A
R9
WR10 = 1
NEXT 9
Program B
Jump to FOR 9 (WR10) after subtracting 1 from the content of WR10.

When R9 is OFF, … The program B is executed after repeating the program A 20 times.

When R9 is ON, … The content of WR10 becomes 0, since the number of times of repetition WR10 becomes 1 and 1 is subtracted on the processing of NEXT 9. Therefore, the program B is executed after the repetition of the program A terminates.

Name	Read of subroutine

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
CAL n	–	3	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		Code No.0 to 199 can be used. (Decimal)
Condition	Time	Condition	Time	
–	0.82	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
n	Code No.																	✓

Function

- When a startup condition of CAL n is ON, the subroutine program of the same code No. from this command (the program surrounded by SB n to RTS) are executed.
- When the startup condition is OFF, the next program is executed.
- CAL of another subroutine can nest to 5 times in a subroutine.
- The subroutine can be called on the interrupt scan program.

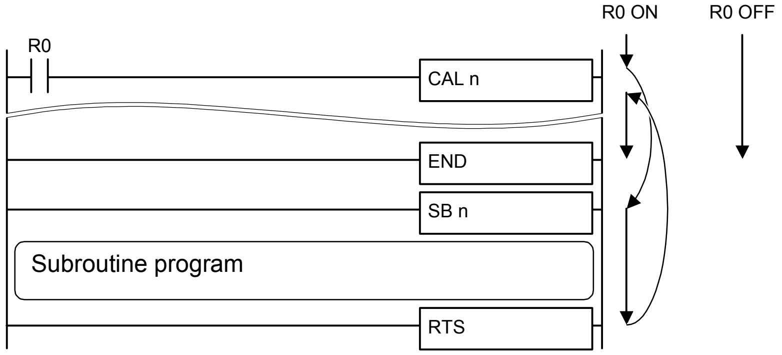
Cautionary notes

Unlike EH-CPU currently in use, the editor detects an assembling error. Note because the program including causes of the assembling error cannot be written into EHV-CPU.

[Reference: Causes of assembling error]

No.	Description of error
1	SB corresponding has not been defined.
2	Nesting error

Program example



[Program description]

- When R0 is ON, the subroutine program is executed at CAL n. The next program of CAL n is re-executed after execution
- When R0 is OFF, the next program is executed without executing the subroutine program.

Name	Start of subroutine program

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
SB n	–	1	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		Code No. 0 to 199 can be used. (Decimal)
Condition	Time	Condition	Time	
—	0.02	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
n	Code No.																	✓

Function

- This command means the start of a subroutine program. (No processing)
- n for SB n cannot be used repeatedly on the same program.
- It is ignored even if a startup condition is put in SB n.
- SB n and RTS should be always used in pairs.
- A subroutine program from SB n to RTS should be written to a sheet for the subroutine or written after END command.

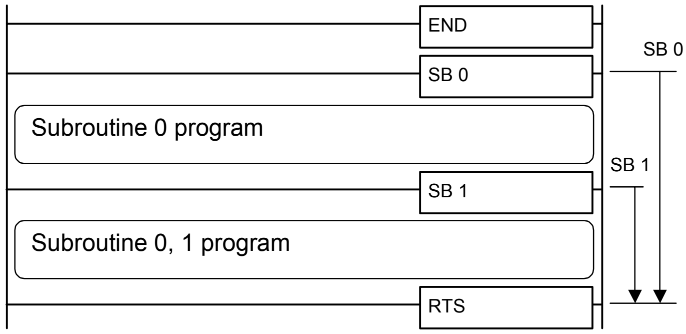
Cautionary notes

Unlike EH-CPU currently in use, the editor detects an assembling error. Note because the program including causes of the assembling error cannot be written into EHV-CPU.

[Reference: Causes of assembling error]

No.	Description of error
1	There is SB of the same No.
2	CAL corresponding has not been defined.

Program example



[Program description]

- If CAL 0 is executed, the program from SB 0 to RTS is executed as a subroutine.
- If CAL 1 is executed, the program from SB 1 to RTS is executed as a subroutine.

Name	Termination of subroutine program

Ladder format	Number of steps		Condition code				
RTS	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
	–	2	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
–	0.74	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
–	(No argument)																	✓

Function

- This command declares the termination of a subroutine program.
- If this command is executed, the program is executed from the next of CAL n command calling the subroutine.

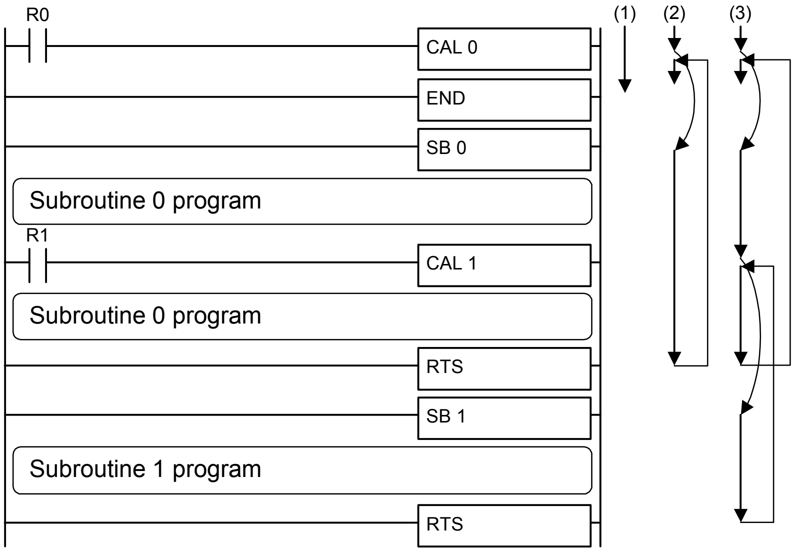
Cautionary notes

- A startup condition should not be put in this command.
- Unlike EH-CPU currently in use, the editor detects an assembling error. Note because the program including causes of the assembling error cannot be written into EHV-CPU.

[Reference: Causes of assembling error]

No.	Description of error
1	There are several RTS.
2	Area error of RTS
3	Startup condition error of RTS

プログラム例



[Program description]

If both R0 and R1 are OFF, it is executed like (1), if only R0 is ON, it is executed like (2), and if both R0 and R1 are ON, it is executed like (3).

Control
RTS

Name	Start of periodic scan program

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
INT (s)	–	1	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		・Up to 4 can be used. ・Cycle can specify from 1 to 60,000 [ms].
Condition	Time	Condition	Time	
–	110+70n	–	–	

Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	Cycle (ms)																	✓

Function

- This command declares the start of an interrupt scan program.
- A cycle (units: ms) of a periodic interrupt scan is specified of s.
- The shorter the cycle is, the higher the order of priority of the interrupt is.
- INT (s) and RTI should be always used in pairs.
- INT (s) is ignored even if a startup condition is put in.
- The interrupt program from INT (s) to RTI should be written to a sheet for a subroutine or written after END command.

Cautionary notes

- The same cycle cannot be used repeatedly.
- A progress value is undated at the execution of the timer command on EHV-CPU. Therefore, the timer may not be turned ON correctly if a program not to scan a portion to execute the timer command is created using the interrupt scan. (The timer is not turned ON if the time not to scan the portion to execute the timer command exceeds the time (= 'time base' × 65,535)). And note because the previous progress value is retained until the timer command is executed.
- Unlike EH-CPU currently in use, the editor detects an assembling error. Note because the program including causes of the assembling error cannot be written into EHV-CPU.

[Reference: Causes of assembling error]

No.	Description of error
1	There is INT of the same No. (There are several INT of the same cycle.)
2	INT has not been defined.

Program example

See the description pages for ”RTI”.

PRN ➔ PRJ

This command is equivalent to INT 0 to 3 in the program (PRN file) of EH-CPU.

When changing the program which has used INT0 to INT3 for EHV, how to convert is as follows.

INT 0 ➔ INT (5)

INT 1 ➔ INT (10)

INT 2 ➔ INT (20)

INT 3 ➔ INT (40)

* If converted by a conversion tool, it is converted as mentioned above.

[Caution on converting the program]

Please modify the program after converting as follows if converting the program which has been used on CPU modules other than EH-CPU516 and 548.

INT (5) ➔ INT (10)

INT (10) ➔ INT (20)

INT (20) ➔ INT (40)

Name	Termination of periodic scan program

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
RTI	—	2	●	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		
Condition	Time	Condition	Time	
—	80	—	—	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
—	(No argument)																

Function

- This command declares the termination of an interrupt scan program.
- The processing returns to the program which was executing before the interrupt scan by executing this command.

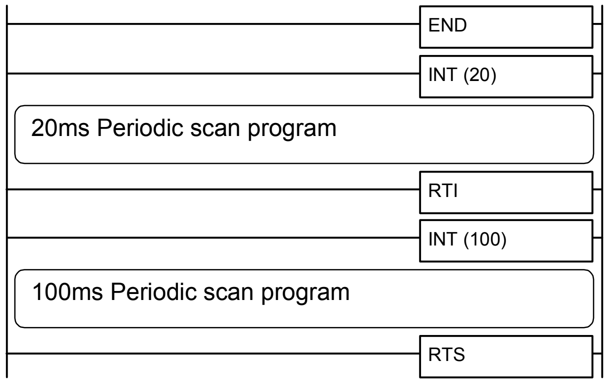
Cautionary notes

- A startup condition should not be put in this command.
- Unlike EH-CPU currently in use, the editor detects an assembling error. Note because the program including causes of the assembling error cannot be written into EHV-CPU.

[Reference: Causes of assembling error]

No.	Description of error
1	RTI has not been defined.
2	Area error of RTI
3	Startup condition error of RTI

Program example



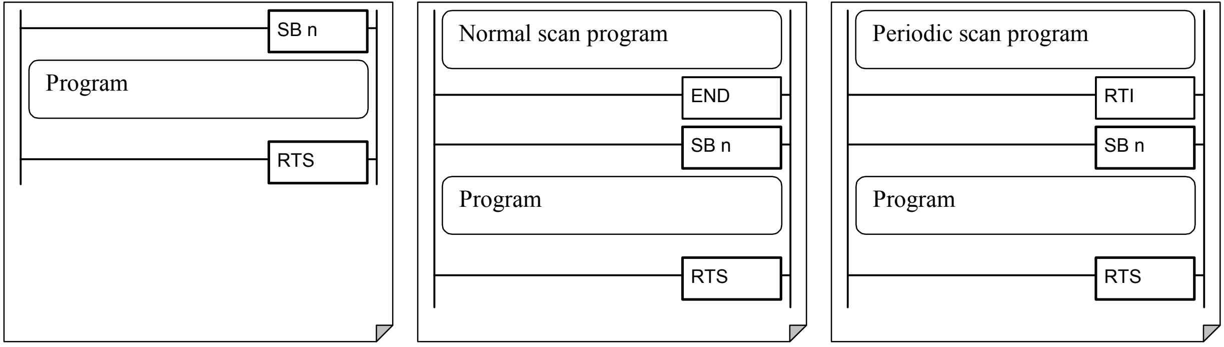
[Program description]

・After the start of RUN, the program from INT (20) to RTI is executed every 20 ms.

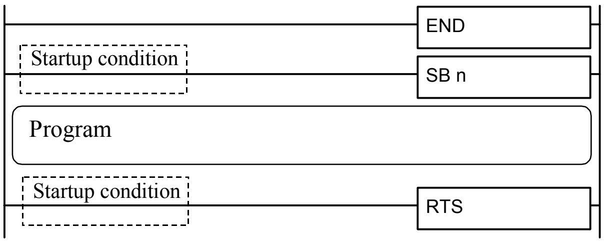
・After the start of RUN, the program from INT (100) to RTI is executed every 100 ms.

Reference	Grammar of SB n, RTS, INT, and RTI

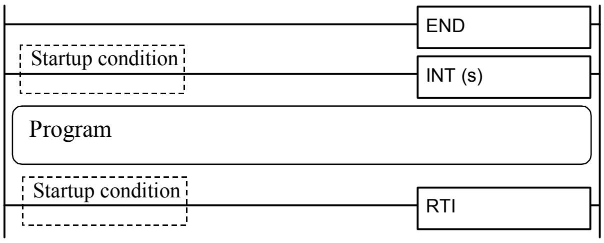
[1] A subroutine is written to a sheet for the subroutine. And after END command of the normal scan, a subroutine can be written after RTI command in the periodic scan sheet.



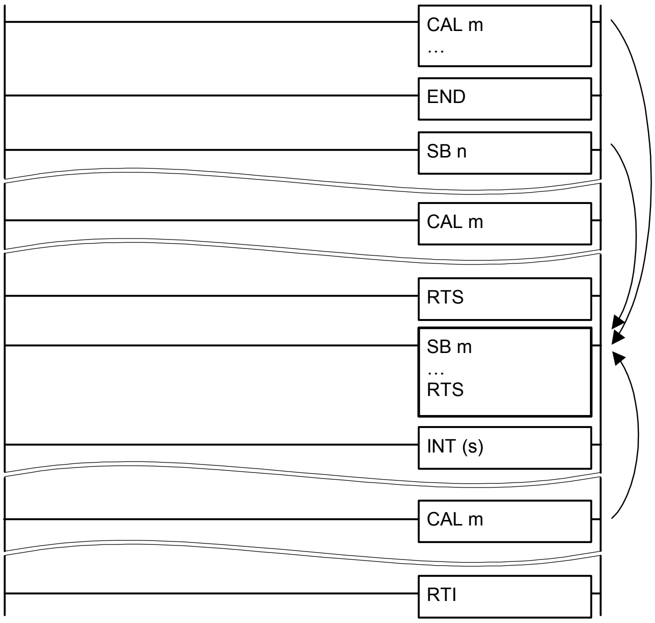
[2] The start command (SB n) and the termination command (RTS) of the subroutine should be programmed without a startup condition.



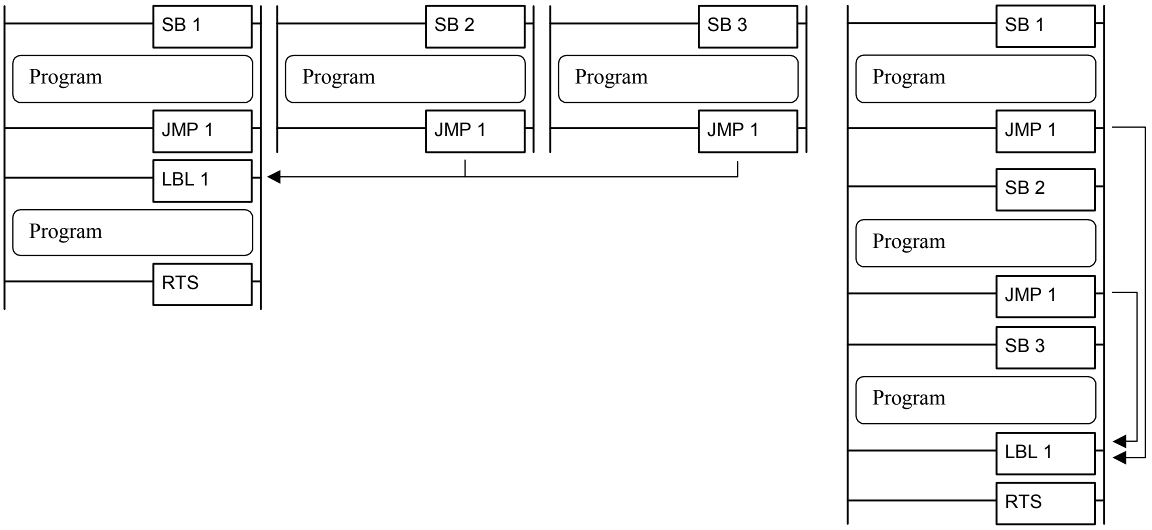
[3] The start command of the periodic scan (INT (s)) and the termination command of the scan (RTI) should be programmed without a startup condition.



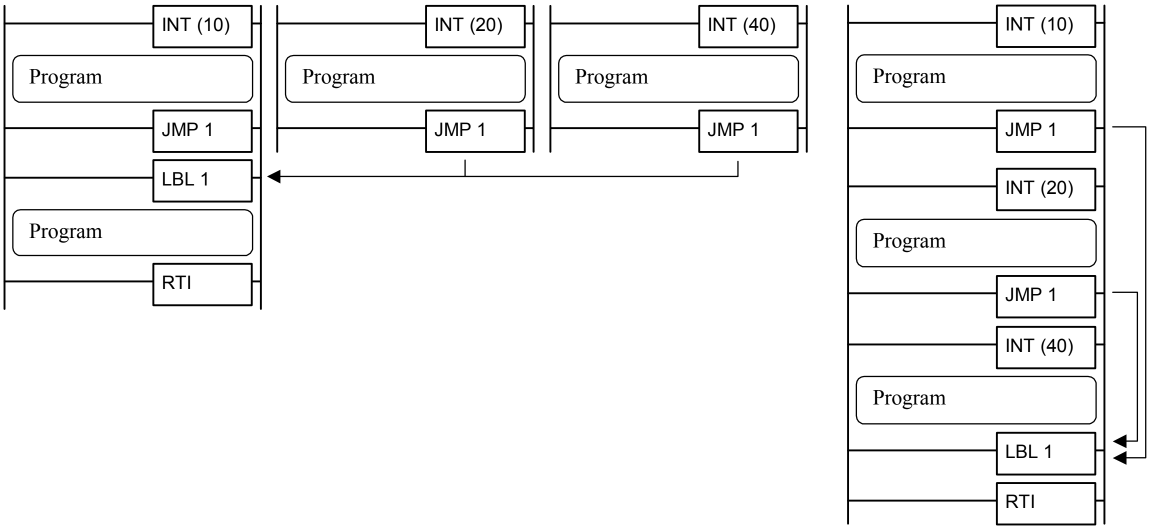
[4] The same subroutine can be called from the normal scan, the interrupt scan, and the subroutine.



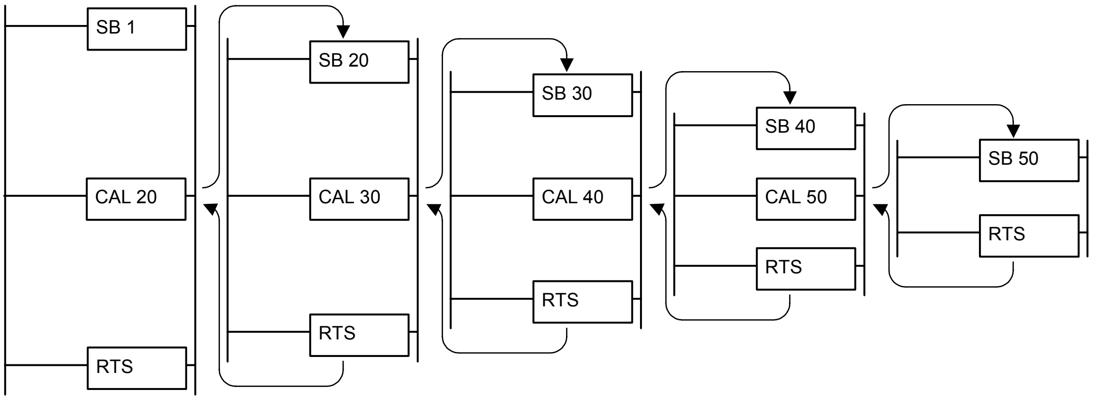
[5] A subroutine with many entrances and one exit can be programmed.



[6] A periodic scan with many entrances and one exit can be programmed.



[7] A subroutine can nest to 5 times.



The order of the program of the subroutine is not related to the order of nesting.

[1] Basic commands

[2] Arithmetic commands

[3] Application commands

[4] Control commands

[5] CPU serial communications commands

[6] High-function module transfer commands

Name	CPU serial communication port Data transfer command

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
TRNS 0 (s, t)	–	6	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		・s parameter can be used up to s+E. ・t parameter can be used up to t+B. ・n of processing time is the number of bytes.
Condition	Time	Condition	Time	
–	552.4+0.64n	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, TDN, SS, WDT, MS, TMR, CU, RCU, CT	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	The start I/O of parameter area										✓						
t	The start I/O of communication control bit				✓												

Function

This is a command to communicate in a serial port of CPU module.

Executing this command can send data from the serial port and receive responses from external devices.

Parameter

TRNS 0 command uses 4 internal output areas shown below.

- Parameter for communication (s parameter area)

Area to set parameters, such as transmission speed and transmission character configuration for communiation.

- Bit for communication control (t parameter area)

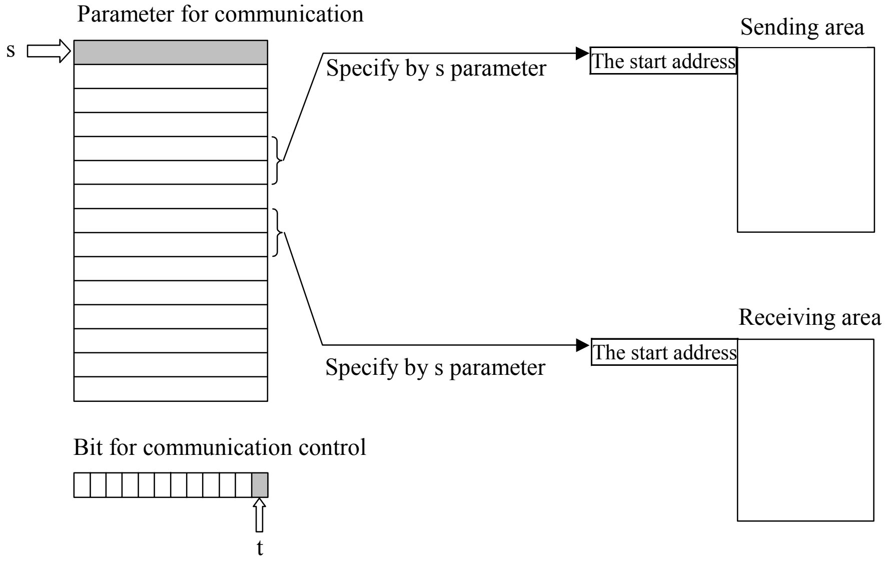
Area to start TRNS 0 command and display a comamnd end and error information.

- Sending area

Area to set sending data.

- Receiving area

Area to store received data after sending.



(1) s parameter

The start I/O of “a table which stores each type of parameter for communication” is set to s.

[The details of s parameter area]

s	[1] Return code
s+1	[2] System area (No using by user)
s+3	[3] Timeout time
s+4	[4] Start I/O of sending data area
s+6	[5] Size of sending data area
s+7	[6] Start I/O of receiving data area
s+9	[7] Size of receiving data area
s+A	[8] Receiving data length
s+B	[9] Start code
s+C	[10] Termination code
s+D	[11] Transmission speed
s+E	[12] Transmission format

▨ No writing area by users

□ Setting area by users

[1] Return code:

The executed result of TRNS 0 is set in the lower 8 bits.

Case of normal end = 0

Case of abnormal end ≠ 0 (Seethe error code list.)

[2] System area:

This is used on the system processing for TRNS 0 when TRNS 0 is executed. Users cannot use this area.

[3] Timeout time:

The timeout time from beginning to end of execution of TRNS 0 is specified.

= 0: The timeout time is not checked.

≠ 0: The timeout of ‘×10ms’ is checked.

(It can set up to HFFFF.)

[4] The start I/O of sending data area:

The start I/O of an area to store sending data by TRNS 0 is specified.

The start I/O of the sending data area is coded by I/O address coding command before executing TRNS 0 to store in s+4 and s+5. (Usable I/O is WR, WL, WM, and WN.)

[5] Size of sending data area:

The size of the sending data area is specified in word units.

[6] The start I/O of receiving data area:

The start I/O of an area to store the response data to the sending data is specified.

The start I/O in the receiving data area is coded by I/O address coding command before executing TRNS 0 to store in s+7 and s+8. (Usable I/O is WR, WL, WM, and WN.)

[7] Size of receiving data area:

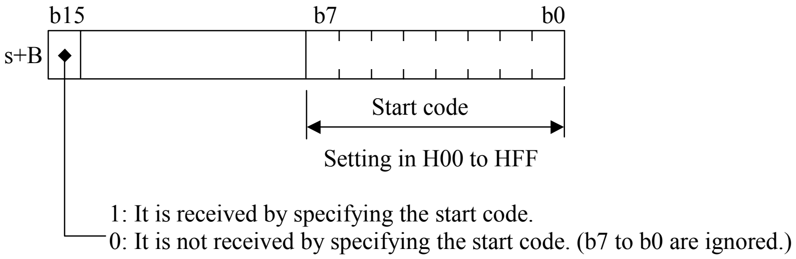
The size of the receiving data area is specified in word units.

[8] Receiving data length:

The receiving data length is specified in byte units. But the length should not exceed 1,024 bytes or the receiving data area. If exceeded, it becomes abnormal end because of DER=1.

[9] Start code:

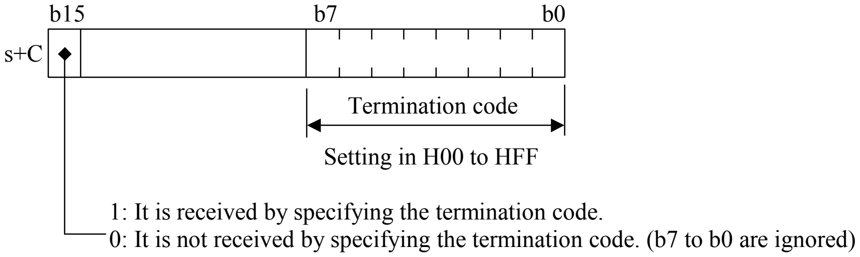
The code to start the receiving is specified.



1: It is received by specifying the start code.
0: It is not received by specifying the start code. (b7 to b0 are ignored.)

[10] Termination code:

The code to terminate the receiving is specified.



1: It is received by specifying the termination code.
0: It is not received by specifying the termination code. (b7 to b0 are ignored)

[11] Transmission speed:

Transmission speed is specified.

Transmission speed	Set value
300 bps	H0000
600 bps	H0001
1,200 bps	H0002
2,400 bps	H0003
4,800 bps	H0004

Transmission speed	Set value
9,600 bps	H0005
19,200 bps	H0006
38,400 bps	H0007
57,600 bps	H0008

[12] Transmission format:

Transmission format is specified.

Transmission format			Set value
7bit	even parity	2 stops	H0000
7bit	odd parity	2 stops	H0001
7bit	even parity	1 stop	H0002
7bit	odd parity	1 stop	H0003
8bit	no parity	2 stops	H0004
8bit	no parity	1 stop	H0005
8bit	even parity	1 stop	H0006
8bit	odd parity	1 stop	H0007

(2) t parameter

The start I/O of “Bit table to control communication” is set to t.

[The details of t parameter]

t+B											t
[10]	[9]	[8]	╲	╲	[7]	[6]	[5]	[4]	[3]	[2]	[1]

□ Set bit by user

╲ Un used

[1] Execution of communication:

The user program sets 1 when TRNS 0 is executed.

TRNS 0 resets it to 0 if communication terminates.

[2] Normal end:

It is set to 1 if communication terminates normally by TRNS 0.

And when communication is started (t bit is turned ON), TRNS 0 resets it to 0.

[3] Abnormal end (ABEND):

It is set to 1 if communication terminates abnormally by TRNS 0.

And when communication is started (t bit is turned ON), TRNS 0 resets it to 0.

[4] Initial requirement:

When TRNS 0 is set to the initial state, it is set to 1. The initial requirement under communicating terminates communication forcedly.

[5] Initial end:

When the initial of TRNS 0 terminates normally, it is set to 1. (In this case, [4] Initial requirement is reset to 0.)

[6] Continuation:

It sets 1 when receiving continuously after terminating the sending. TRNS 0 resets it to 0 after terminating communication.

[7] Parity error / Framing error / Overrun error:

If parity error, framing error, or overrun error occurs under communicating, it is set to 1.

[8] Timeout:

If a time out occurs under communicating, it is set to 1.

[9] Input buffer full:

If a receiving buffer is full, it is set to 1.

[10] Contention error:

If 2 TRNS 0 or more are going to be started simultaneously on the user program, or TRNS 0 and RECV 0 are started simultaneously, it is set to 1. (In this case, communication is terminated forcedly.)

* [7] to [10] are reset to 0 by TRNS 0 initially when TRNS 0 is started.

(3) Sending data area

The setting of data to send follows the composition shown below.

The number of bytes to send is even.

I/O address

Specify by s+4 and s+5

Number of bytes to send (N)	
The 1st byte	The 2nd byte
The 3rd byte	The 4th byte
The 5th byte	The 6th byte
The N-1th byte	The Nth byte

The number of byte to send is odd.

I/O address

Specify by s+4 and s+5

Number of bytes to send (N)	
The 1st byte	The 2nd byte
The 3rd byte	The 4th byte
The 5th byte	The 6th byte
The Nth byte	Invalid data

Size of the sending data area

(4) Receiving data area

The setting of receiving data follows the composition shown below.

The number of bytes to receive is even.

I/O address

Specify by s+7 and s+8

Number of byte to receive (N)	
The 1st byte	The 2nd byte
The 3rd byte	The 4th byte
The 5th byte	The 6th byte
The N-1th byte	The Nth byte

The number of bytes to receive is odd.

I/O address

Specify by s+7 and s+8

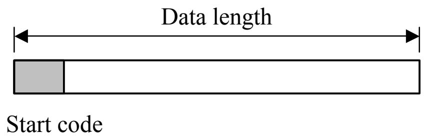
Number of bytes to receive (N)	
The 1st byte	The 2nd byte
The 3rd byte	The 4th byte
The 5th byte	The 6th byte
The Nth byte	Invalid data

Size of the receiving data area

Method of data communication

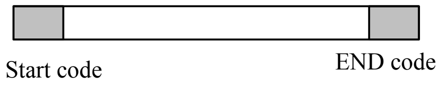
A method of data communication is specified from the following 4 ways.

(1) To specify by the start code and the receiving data length.



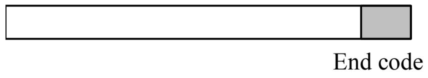
s+A : The receiving data length (byte)
s+B : H80□□ (□□ is the start code.)
s+C : H0000

(2) To specify by the start code and the end code.



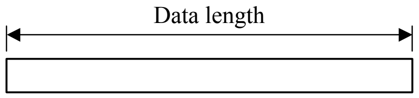
s+A : H0000
s+B : H80□□ (□□ is the start code.)
s+C : H80□□ (□□ is the end code.)

(3) To specify by the end code.



s+A : H0000
s+B : H0000
s+C : H80□□ (□□ is the end code.)

(4) To specify by the receiving data length.

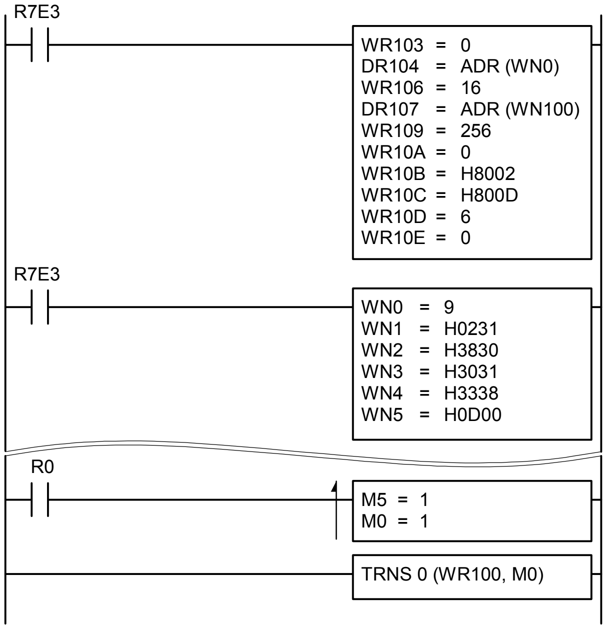


s+A : The receiving data length (byte)
s+B : H0000
s+C : H0000

Cautionary notes

- It can act when a serial communication port is set for ‘a general-purpose port’.
- TRNS 0 initializes an internal work area at the 1st scan after RUN. Thus the set of bit (t+0) to execute communication should be executed after the 2nd scan.
- When a startup condition is before TRNS 0, the startup condition should not be specified since a system software may not be able to execute the initializing process normally.
- s+E and t+B should be used within the range of I/O. It is impossible to write a parameter outside the range of I/O.
- If timeout occurred when receiving data, data which had received by the time the timeout has occurred is stored in the receiving data area if those data are normal.
- When communicating by RS-232C port, ER signal is turned ON at the timing of receiving the execution of communication normally.
- When communicating by RS-232C port, ER signal is turned OFF at the timing of followings.
  (1) When the initial request is turned ON under communicating. (But ER signal is keeping ON when being done after communication is completed.)
  (2) When switching from RUN to STOP, and then to RUN under communicating. (But ER signal is keeping ON when being done after communication is completed.)
  (3) When timeout occurs under communicating.
  (4) When the range error occurs because s and t parameters are rewritten under communicating.
- When writing commands into the periodic scan, the cycle of the periodic scan should be 10 ms or more.

Program example



```
WR103 = 0
DR104 = ADR (WN0)
WR106 = 16
DR107 = ADR (WN100)
WR109 = 256
WR10A = 0
WR10B = H8002
WR10C = H800D
WR10D = 6
WR10E = 0

WN0  = 9
WN1  = H0231
WN2  = H3830
WN3  = H3031
WN4  = H3338
WN5  = H0D00

M5 = 1
M0 = 1

TRNS 0 (WR100, M0)
```

R7E3: 1st scan turns ON after RUN
No timeout
Sending area 16 words from WN0
Receiving area 256 words from WN100
Method of receiving data:
Start code H02, End code H0D
Transmission speed 19.2k bps
Transmission format 7 bits, EVEN, 2 stops

Sending data Total 9 bytes
Inverter (SJ300 / L300P) command
Forward to station No.18
02 31 38 30 30 31 33 38 0D
(STX 18 00 1 BCC CR)

If R0 is turned ON, the bit (M0) to execute is turned ON and data is sent.
If M5 is turned ON, responses data from the other device can be receive after sending.

[Program description]

This is a sample program to send a forward data to our inverter SJ300/L300P. The parameter of TRNS 0 and the sending data are set at the first scan after RUN. The bit to execute M is started if R0 is turned ON, and data is sent. If the command is executed normally, the response from the inverter is stored in WN100, or after.

Return code

The following table is a list of the return code to be stored in the top of s parameter area after executing TRNS 0 / RECV 0.

Return code	Name	Description	Countermeasure
H00	Normal end	Sending and receiving were terminated normally.	—
H22	Setting error of sending area	Setting of the top of the sending area is not correct.	Set the top of the sending area within correct range.
H23	Range error of sending area	The end of the sending area exceeds the range of I/O.	Set the sending area within correct range.
H24	Setting error of receiving area	Setting of the top of the receiving area is not correct.	Set the top of the receiving area within correct range.
H25	Range error of receiving area	The end of the receiving area exceeds the range of I/O.	Set the receiving area within correct range.
H26	Setting error of sending data length	Setting of the sending data length is more than the sending area length.	Set the sending data length within the range of the sending area.
H27	Setting error of receiving data length	Setting of the receiving data length is more than the receiving area length.	Set the receiving data length within the range of the receiving area.
H28	Area overlap error *1	There is an overlapped area between s parameter, t parameter, the sending area, and the receiving area.	Set each area without overlapping those areas.
H30	Timeout *2	Processing of sending and receiving was not terminated within the specified time.	Make the set value larger, or check the details of processing.
H40	Data over of receiving area *3	There is no space because the receiving area is filled with the receiving data	Make the receiving area larger.
H41	Parity error Framing error Overrun error *4	One of parity error, framing error, and overrun error occurred on the communication processing.	Check the transmission route of a general-purpose port and data format, etc.
H44	Contention error	TRNS 0/RECV 0 was started simultaneously at 2 places or more.	Do not start simultaneously at 2 places or more.
H45	Parameter error	The set values of baud rate of TRNS 0/RECV 0 and the transmission code are not correct.	Set the correct value.
H46	Error of port specification	TRNS 0 or RECV 0 was started when a general-purpose port was not specified.	Check which port has been specified.
H55	Un-connection error of lines Error of cutting line under communicating	When connecting the modem, TRNS 0 or RECV 0 was started without connecting the line after being switched between dedicated port and general purpose port by TRNS 8. After being switched between dedicated line and general-purpose line by TRNS 8 when connecting the modem, as a result of using TRNS 0 or RECV 0 in the state where a line connection has been completed, the line was cut off, when telephoning.	Check whether the line is connecting.

*1 Though a return code of area overlap error is H28, note that the return code may not be displayed as H28 if the return code area overlaps with a part of t parameter.

*2 Though it becomes a timeout error if a timeout occurs under receiving data, received data by the time the timeout occurs is stored in the receiving data area.

*3 The size of the receiving area is up to 1,024 bytes.

*4 The receiving data is not guaranteed at the time of receiving.

PRN ➔ PRJ

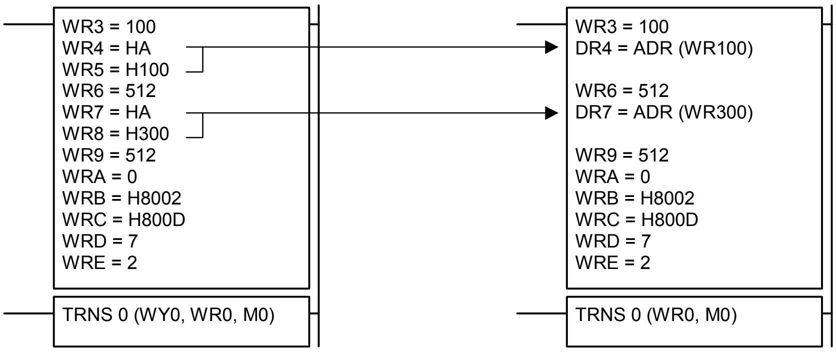
This command is equivalent to TRNS 0 (d, s, t) in the program (PRN file) of EH-CPU.

When changing the program which has used TRNS 0 (d, s, t) for EHV, how to convert is as follows.

TRNS 0 (d, s, t)	➔	TRNS 0 (s, t)
s+4 : I/O types of the sending data area	➔	s+4, s+5 : Sending data by I/O address coding command
s+5 : I/O No. of the sending data area	➔	Specify the start address of the area
s+7 : I/O types of the receiving data area	➔	s+7, s+8 : Receiving data by I/O address coding command
s+8 : I/O No. of the receiving data area	➔	Specify the start address of the area

Ex.) Case of TRNS 0 (WY0, WR0, M0), 512 words from the sending data area, and 512 words from the sending data area WR300.

Program for EH-CPU Program for EHV-CPU



```
WR3 = 100
WR4 = HA
WR5 = H100
WR6 = 512
WR7 = HA
WR8 = H300
WR9 = 512
WRA = 0
WRB = H8002
WRC = H800D
WRD = 7
WRE = 2

TRNS 0 (WY0, WR0, M0)
```

```
WR3 = 100
DR4 = ADR (WR100)

WR6 = 512
DR7 = ADR (WR300)

WR9 = 512
WRA = 0
WRB = H8002
WRC = H800D
WRD = 7
WRE = 2

TRNS 0 (WR0, M0)
```

* A conversion tool cannot convert a specific area of the start I/O of the sending data area and the receiving data area. Thus please convert as mentioned above.

[Caution on converting a program]

The difference in action of TRNS 0 / RECV 0 between EH-CPU and EHV-CPU is shown below.

Item	EH-CPU	EHV-CPU
The receiving data at the time of occurrence of timeout	The receiving data is cancelled.	Received data by the time the timeout occurs is stored in the receiving data area.
Error of communication data	Parity error, framing error, and overrun error can be distinguished.	Parity error, framing error, and overrun error cannot be distinguished.

Name	CPU serial communication port Data receiving command

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
RECV 0 (s, t)	–	6	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		・s parameter can be used up to s+E. ・t parameter can be used up to t+B. ・n for processing time is the number of bytes.
Condition	Time	Condition	Time	
–	550.4+0.61n	–	–	

Usable I / O		Bit						Word					Double word				Constant
		X	Y	EX, EY	R, L, M	TD, SS, MS, CU, CT, TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
s	The start I/O of parameter area										✓						
t	The start I/O of communication control bit				✓												

Function

This is a command to communicate in a serial port of CPU module.

Executing this command can receive data from external devices on the serial port and send data after receiving.

Parameter

RECV 0 command uses 4 internal output areas shown below.

- Parameter for communication (s parameter area)

Area to set parameters, such as transmission speed and transmission character configuraiton for communication.

- Bit for communication control (t parameter area)

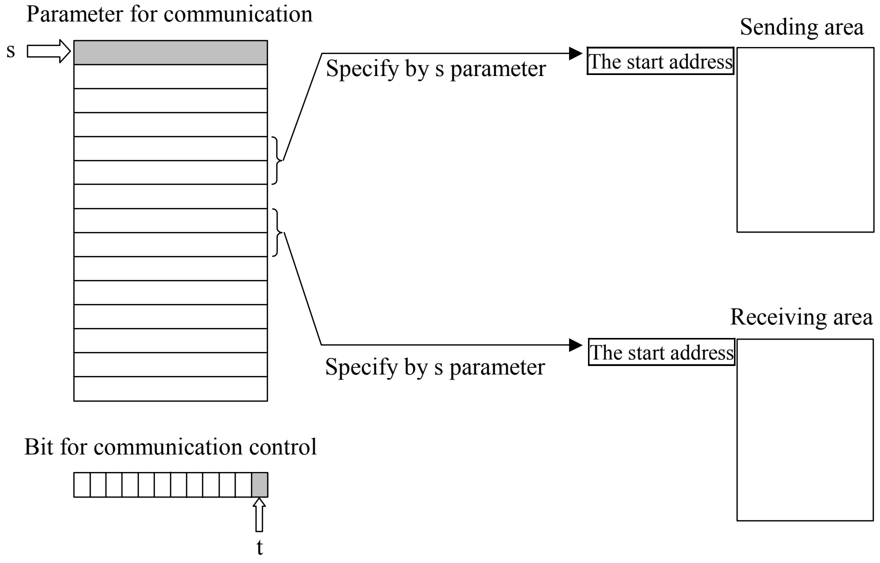
Area to start RECV 0 command and display the command end and error information.

- Sending area

Area to set sending data.

- Receiving area

Area to store received data .



(1) s parameter

The start I/O of “a table which stores each parameter for communication” is set to s.

Each parameter’s meaning is the same as TRNS 0 command. See the description of TRNS 0 for details.

(2) t parameter

The start I/O of “a bit table to control communication” is set to t.

A meaning of each bit is the same as the content of TRNS 0 command except for a continuity bit (t+5). See the description of TRNS 0 for details.

t+B											t
[10]	[9]	[8]	⧅	⧅	[7]	[6]	[5]	[4]	[3]	[2]	[1]

□ Set bit by user

⧅ Un used

[6] Continuation:

It sets 1 when sending continuously after terminating the receiving. RECV 0 resets it to 0 after terminating communication.

(3) Sending data area

The composition of the sending data area is the same as TRNS 0. See the description of TRNS 0 for details.

(4) Receiving data area

The composition of the receiving data area is the same as TRNS 0. See the description of TRNS 0 for details.

Cautionary notes

- RECV 0 initializes an internal work area at the 1st scan after RUN. Thus the set of bit (t+0) to execute communication should be executed after the 2nd scan.
- When a startup condition is before RECV, the startup condition should not be specified since system software may not be able to execute the initializing process normally.
- s+E and t+B should be used within the range of I/O. It is impossible to write a parameter outside the range of I/O.
- If timeout occurred when receiving data, data which had received by the time the timeout has occurred is stored in the receiving data area if those data are normal.
- When writing commands into the periodic scan, the cycle of the periodic scan should be 10 ms or more.

Method of data communication

A method of data communication is specified from the following 4 ways.

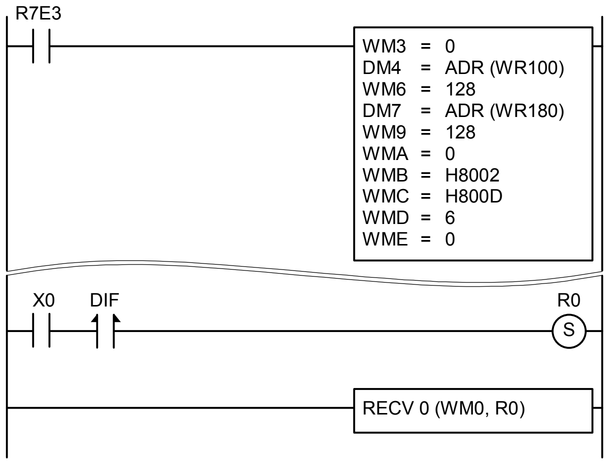
(1) To specify by the start code and the receiving data length.

(2) To specify by the start code and the end code.

(3) To specify by the end code.

(4) To specify by the receiving data length.

Program example



R7E3: 1st scan turns ON after RUN.
No timeout
Sending area 128 words from WR100
Receiving area 128 words from WR180
Method of data receiving:
Start code H02, End code H0D
Transmission speed 19.2k bps
Transmission format 7 bits, EVEN, 2 stops

[Program description]

The parameter of RECV 0 is set at the first scan after RUN.

When X0 turns ON, the executing bit R0 is started and it waits for data receiving. (It keeps waiting until data is received since the setting is ‘No timeout’.)

If data from external devices is received normally, the receiving data is stored in WR180 or after.

PRN ➔ PRJ

This command is equivalent to RECV 0 (d, s, t) in the program (PRN file) of EH-CPU.

When changing the program which has used RECV 0 (d, s, t) for EHV, how to convert is as follows.

RECV 0 (d, s, t)	➔	RECV 0 (s, t)
s+4 : I/O types of sending data area	➔	s+4, s+5 : Sending data by I/O address coding command
s+5 : I/O No. of sending data area		Specify the start address of the area
s+7 : I/O types of receiving data area	➔	s+7, s+8 : Receiving data by I/O address coding command
s+8 : I/O No. of receiving data area		Specify the start address of the area

* A conversion tool cannot convert a specific part of each start I/O of the sending data area and the receiving data area. Please convert referring the description pages of TRNS 0.

[1] Basic commands

[2] Arithmetic commands

[3] Application commands

[4] Control commands

[5] CPU serial communications commands

[6] High-function module Transfer commands

Name	Data transfer command for EH-ID (ID reader interface module)

Ladder format	Number of steps		Condition code				
	Condition	Steps	R7F4	R7F3	R7F2	R7F1	R7F0
			DER	ERR	SD	V	C
TRNS 7 (d, s, t)	—	6	↕	●	●	●	●

Command processing time (μs)				Remarks
Average		Maximum		・s parameter can be used up to s+10. ・t parameter can be used up tot+5. ・n for processing time is the number of bytes.
Condition	Time	Condition	Time	
—	619.77+0.23n	—	—	

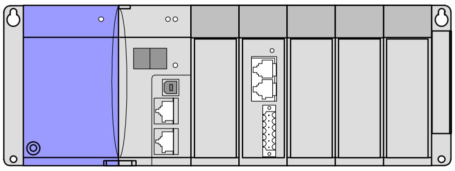
Usable I / O		Bit							Word					Double word				Constant
		X	Y	EX EY	R, L, M	TD, SS, MS, CU, CT	TDN, WDT, TMR, RCU,	WR, WN (.m)	WX	WY	WEX, WEY	WR, WL, WM, WN	TC	DX	DY	DEY	DR, DL, DM	
d	The position to mount a module									✓								
s	The top I/O of parameter area											✓						
t	The top I/O of bit of communication control				✓													

Function

Executing this command can communicate on EH-ID (ID reader interface module).

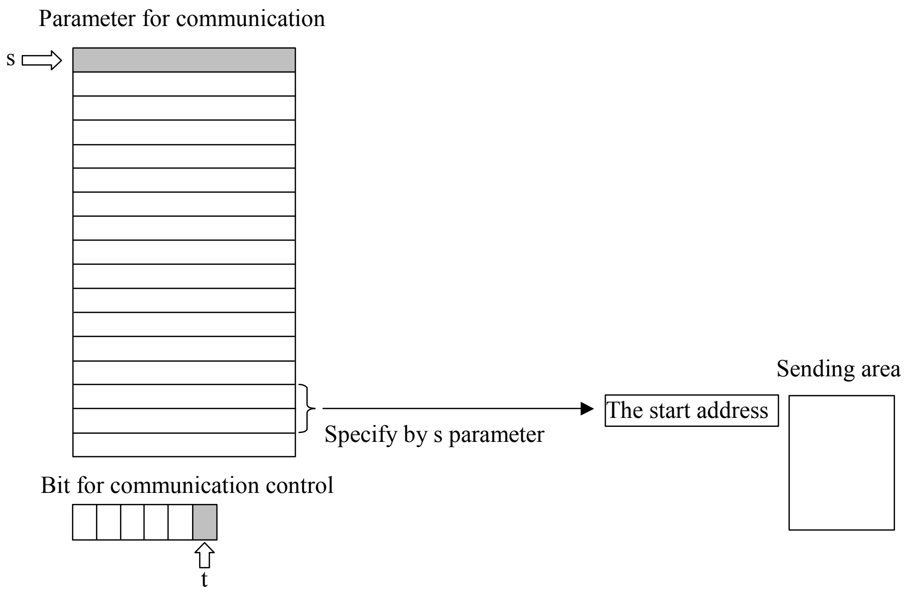
TRNS 7 is a command to transfer the command and the sending data from CPU module to EH-ID.

(RECV 7 is a command to read data that EH-ID has received.)



Specify the position to mount a module.

d(WYus4 u: Unit No. s: Slot No.)



Parameter

(1) d parameter

The position (a minimum No. in WY of I/O allocation) to mount EH-ID on which TRNS 7 command can communicate is set to d.

(2) s parameter

The top I/O of “a table which stores each type of parameter for communication” is set to s.

[Details of s parameter area]

s	[1] Return code
s+1	[2] System area (No using by user) Using 10 words
s+A	
s+B	[3] Command code
s+C	[4] Specify port
s+D	[5] Timeout
s+E	[6] The start I/O of sending data area
s+10	[7] The number of bytes to send

▨ No writing area by user

□ Setting area by user

[1] Return code:

Result that TRNS 7 is exwcuted is set to lower 8 bits.

Normal end = 0

Abnormal end (ABEND) ≠ 0 (See a list of retuen code of TRNS 7.)

[2] System area:

It is used on the system processing of TRNS 7 when TRNS 7 is executed. User cannot use this area.

[3] Command code:

The command to EH-ID is set.

H50	:	Initial command
H52	:	Command to write data into EH-ID
H53	:	Command to set mode
H55	:	Display of version
H56	:	Display of mode setting (DIP Sw1)
H57	:	Display of mode setting (DIP Sw2)

[4] Specify port:

Specify the port intended for command/data.

Port 1 : H0000, Port 2 : H0001

[5] Time of timeout

Specify the timeout time from the start to the end of an execution of TRNS 7.

= 0: The timeout time is checked.

≠ 0: The timeout time of ‘×10ms’ is checked.

(Up to HFFFF can be set.)

[Caution on the timeout time]

- A setting by s parameter is the timeout time between EHV-CPU and EH-ID. The timeout time between EH-ID and an external device is set by the command H53.
- Please make this timeout time larger than the timeout time between EH-ID and the external device, which is set by the command H53.

[6] The top I/O of sending data area:

Specify the top I/O of the area which stored data sent by TRNS 7.

The top of the sending data area should be coded by I/O address coding command in order to store in s+E and s+F before executing TRNS 7. (Usable I/O is WR, WL, WM, and WN.)

[5] The number of bytes to send:

Specify the sending data length.

(3) t parameter

The top I/O of "abit table for controlling commnication" is set to t.

[Details of t parameter area]

t+5					t
[6]	[5]	[4]	[3]	[2]	[1]

□ Set bit by user

[1] Execution of communication:

The user program sets 1 when executing TRNS 7. It is resets it to 0 by TRNS 7 when the communication teminates.

[2] Normal end:

When the communication by TRNS 7 teminates normally, it is set to 1.

And when the communication is started (t bits is turned ON), it is reset to 0 by TRNS 7.

[3] Abnormal end:

When the communication by TRNS 7 terminates abnormally, it is set to 1.

And when the communication is started (t bit is turned ON), it is reset to 0 by TRNS 7.

[4] Initial request:

When TRNS 7 returns to an initial condition, it is set to 1.

[5] Initial end:

When an initializing of TRNS 7 terminates normally, it is set to 1. (In this case, [4]Initial request is reset to 0.)

[6] Module initial end:

When an initializing of EH-ID is completed, it is set to 1.

(4) Sending data area

In the area which stores data to send, the sending data should be set according to the following compostiion.

The number of bytes to send is even.

I/O address

Specify by s+E and s+F.

The 1st byte	The 2nd byte
The 3rd byte	The 4the byte
The 5th byte	The 6th byte
The N-1th byte	The Nth byte

The number of bytes to send is odd.

I/O address

Specify by s+E and s+F.

The 1st byte	The 2nd byte
The 3rd byte	The 4th byte
The 5th byte	The 6th byte
The Nth byte	Invalid data