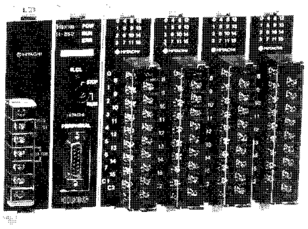
HITACHI

APPLICATION MANUAL
H-SERIES
H-250/252B
(SOFTWARE EDITION)



NJI-168A(X)

WARNING

To ensure that the equipment described by this manual. As well as all equipment connected to and used with it, operate satisfactorily and safety, all applicable local and national codes that apply to installing and operating the equipment must be followed. Since codes can vary geographically and can change with time, it is the user's responsibility to determine which standard and codes apply, and to comply with them.

FAILURE TO COMPLY WITH APPLICABLE CODES AND STANDARDS CAN RESULT IN DAMAGE TO EQUIPMENT AND/OR SERIOUS INJURY TO PERSONNEL.

INSTALL EMARGENCY POWER STOP SWITCH, WHICH OPERATES INDEPENDENTLY OF THE PROGRAMMABLE CONTROLLER TO PROTECT THE EQUIPMENT AND/OR PERSONNEL IN CASE OF THE CONTROLLER MALFUNCTION.

Personnel who are to install and operate the equipment should carefully study this manual and any other referred to by it prior to installation and/or operation of the equipment. Hitachi Industrial Equipment Systems Co., Ltd. constantly strives to improve its products, and the equipment and the manual(s) that describe it may be different from those already in your possession.

If you have any questions regarding the installation and operation of the equipment, or if more information is desired, contact your local Authorized Distributor or Hitachi Industrial Equipment Systems Co., Ltd.

IMPORTANT

THIS EQUIPMENT GENERATES, USES, AND CAN RADIATE RADIO FREQUENCY ENERGY AND, IF NOT INSTALLED AND USED IN ACCORDANCE WITH THE INSTRUCTION MANUAL, MAY CAUSE INTERFERENCE TO RADIO COMMUNICATIONS. AS TEMPORARILY PERMITTED BY REGULATION, IT HAS NOT BEEN TESTED FOR COMPLIANCE WITH THE LIMITS FOR CLASS A COMPUTING DEVICES PURSUANT TO SUBPART J OF PART 15 OF FCC ROULES, WHICH ARE DESIGNED TO PROVIDE PEASONABLE PROTECTION AGAINST SUCH INTERFERENCE.

OPERATION OF THIS EQUIPMENT IN A RESIDENTIAL AREA IS LIKELY TO CAUSE INTERFERENCE IN WHICH CASE THE USER, AT HIS OWN EXPENSE, WILL BE REQUIRED TO TAKE WHATEVER MEASURES MAY BE REQUIRED TO CORRECT THE INTERFERENCE.

LIMITED WARRANTY AND IMITATION OF LIABILITY

Hitachi Industrial Equipment Systems Co., Ltd. (Hitachi) warrants to the original purchaser that the programmable logic controller (PLC) manufactured by Hitachi is free from defects in material and workmanship under normal use and service. The obligation of Hitachi under this warranty shall be limited to the repair or exchange of any part or parts which may prove defective under normal use and service within eighteen (18) months from the date of manufacture or twelve (12) months from the date of installation by the original purchaser which ever occurs first, such defect to be disclosed to the satisfaction of Hitachi after examination by Hitachi of the allegedly defective part or parts. This warranty in expressly in lieu of all other warranties expressed or implied including the warranties of merchantability and fitness for use and of all other obligations or liabilities and Hitachi neither assumes, nor authorizes any other person to assume for Hitachi, any other liability in connection with the sale of this PLC. This warranty shall not apply to this PLC or any part hereof which has been subject to accident, negligence, alternation, abuse, or misuse. Hitachi makes no warranty whatsoever in respect to accessories or parts not supplied by Hitachi. The term “original purchaser”, as used in this warranty, shall be deemed to mean that person for whom the PLC in originally installed.

In no event, whether as a result of breach of contract, warranty, tort (including negligence) or otherwise, shall Hitachi or its suppliers be liable for any special, consequential, incidental or penal damages including but not limited to, loss or profit or revenues, loss of use of the products or any associated equipment, damage to associated equipment, cost of capital, cost of substitute products, facilities, services or replacement power, down time costs, or claims of original purchaser’s customers for such damages.

To obtain warranty service, return the product to your distributor, or send it with a description of the problem, proof of purchase, post paid, insured, and in a suitable package to:

Quality Assurance Dept.
Hitachi Industrial Equipment Systems Co., Ltd.
46-1 Ooaza-Tomioka Nakajo-machi
Kitakanbara-gun, Niigata-ken
959-2608 JAPAN

The Information and/or drawing set forth in this document and all right in and to inventions disclosed herein and patent which might be granted thereon disclosing or employing and the materials, methods, techniques or apparatus described herein are the exclusive property of Hitachi Industrial Equipment Systems Co., Ltd .

No copies of the information or drawings shall be made without the prior constant of Hitachi Industrial Equipment Systems Co., Ltd.

Hitachi Industrial Equipment Systems Co., Ltd. provides customer assistance in varied technical areas. Since Hitachi does not possess full access to data concerning all of the uses and applications of customer's products, responsibility is assumed by Hitachi neither for customer product design nor for any infringement of patents or rights of others, which may result from Hitachi assistance.

The specifications and descriptions contained in this manual were accurate at the time they were approved for printing. Since Hitachi Industrial Equipment Systems Co., Ltd. Incorporated constantly strives to improve all its products, we reserve the right to make changes to equipment and/or manual at any time without notice and without incurring any obligation other than as noted in this manual.

Hitachi Industrial Equipment Systems Co., Ltd. assumes no responsibility for errors that may appear in this manual.

As the product works with user program, and Hitachi Industrial Equipment Systems Co., Ltd. cannot test all combination of user program components, it is assumed that a bug or bugs may happen unintentionally. If it is happened: please inform the fact to Hitachi Industrial Equipment Systems Co., Ltd. or its representative. Hitachi will try to find the reason as much as possible and inform the countermeasure when obtained.

Nevertheless Hitachi Industrial Equipment Systems Co., Ltd. intends to make products with enough reliability, the product has possibility to be damaged at any time. Therefore personnel who are to install and operate the equipment have to prepare with the countermeasure such as power off switch can be operated independently of the controller. Otherwise, it can result in damage to equipment and/or serious injury to personnel.

CAUTION FOR SAFETY

Before installation, operation, maintenance and inspection, you have to read all this manual and other accessory documents carefully and use it correctly.

Use it after mastering all about knowledge of hardware, information of safety and precautions.

And this manual must be reached at hand of last responsible person for maintenance.

In this manual, the ranks of precaution for safety are divided as "DANGER" and "CAUTION".

◇! DANGER

: The case which dangerous situation may happen, fear of death or receiving a serious injury is assumed when user makes a mistake in handling.

⚠ CAUTION

: The case which dangerous situation may happen, fear of receiving a middling injury and a slight injury is assumed and occurrence of only material injury is assumed when user makes a mistake in handling.

Moreover, the matters described in ⚠ CAUTION also may be combined with serious result in some situation.

As both items state a important contents, please follow it certainly.

The explanations of picture display for inhibition and compulsion are described as follows.

🛇 : It displays inhibition (a matter they must not do).
For example in case of Fire Strictly Prohibited, it is 🛇.

❗ : It displays compulsion (a matter they must do certainly).
For example in case of Earth, it is ⏚ .

1. About installation

⚠ CAUTION
◦ Use it in the environment described in the catalog and manual. If you use it in the environment which high temperature , high humidity , dust, corrosive gas, vibration, and shock exist, these may become a cause of electric shock , fire , and wrong movement.
◦ Install it according to manual. If installation is defective , it may become a cause of fall, breakdown and wrong movement.
◦ Don’t enter an alien substance such as wiring fall. It may become a cause of fire , breakdown , and wrong movement.

2. About wiring

COMPULSION

- Carry out earth certainly.
  If earth is not carried out, a fear of electric shock and wrong movement exist.

⚠ CAUTION

- Connect a power supply matching to rate.
  If a power supply mismatching to rate is connected, it may become a cause of fire.
- As for wiring work, specialist owning a qualification should carry out it.
  If wiring is mistaken, a fear of fire, breakdown, and electric shock exist.

3. Caution in using

◇ DANGER

- Exercise care not to touch the terminal during conduction.
  A fear of electric shock exist.
- Construct emergency stop circuit, interlock circuit and so on outside of PC.
  Because of a breakdown of PC, a fear of machine damage and accident exist.

⚠ CAUTION

- Operations such as program change in running, forced output, RUN, and STOP must be carried out in adequate confirmation of safety. Because of mistake in operation, a fear of machine damage and accident exist.
- Turn on according to a order of turning on power supply.
  Because of wrong movement, a fear of machine damage and accident exist.

4. About maintenance

DANGER

- Exercise care not to connect a battery in reverse of + , — , charge it , decomposition it , heat it , throw it into fire , short it.
  A fear of explosion and catching fire exist.

INHIBITION

- Exercise care not to break up and reconstruct PC.
  It becomes a cause of fire , breakdown , and wrong movement.

CAUTION

- Mounting and dismounting module / unit must be carried out after turning off power.
  It may become a cause of electric shock , wrong movement , and breakdown.
- As to fuse , exchange to appointed one.
  It becomes a cause of fire and breakdown.

The history of revision

No.	page	Items	Contents	number of manual
1	—	There are missing pages from3－208 to 3－278	correct	96. 11 NJI－168A(X)

INTRODUCTION

We appreciate that you have selected the H-series H-250/252 of the Hitachi programmable controller . This instruction manual outlines each module focusing to the H-series H-250/252 CPU modules of the Hitachi programmable controller(hereinafter abbreviated to PC). The manual consists of the following five parts according to contents. Carefully read the manual to familiarize yourself with the procedures respectively of installation, operation, and maintenance and inspection.

The manual consists of the following two separate volumes.

Hardware Edition

Part I	Outline of H-series H-250/252 P C: Concept on H-series H-250/252
Part II	System Device Specification: Specification of each module of H-series H-250/252
Part III	Installation, Mounting, Wiring, and Preparation for Running: From installation to trial run of H-series H-250/252
Part IV	Maintenance, Inspection, and Error Recovery Processing: Maintenance, etc. of H-series H-250/252
Appendix	List of special internal outputs

Software Edition

Chapter 1	Input/Output signal: Allocation of I/O signals of H-series H-250/252 and special internal output
Chapter 2	Processing Method and Concept on Scanning: Processing method of H-series H-250/252 and concepts on scanning
Chapter 3	Ladder/Command: H-series H-250/252 Ladder/Command programming language
Chapter 4	Communication with Host Computer: Communication function of the CPU module of H-series
Appendix	List of error code, error recovery processing, and special internal output

For H-300/700/2000 and H-302/702/2002,H- 100M, H-200, or HB-700 of the H-series CPU module, refer to the dedicated manual.

Caution

In this manual, descriptions relating to H-252 are for the series name H-252B.

Note:

(1) No part of this manual may be reproduced without the prior permission of Hitachi.

(2) Parts of this manual may be changed without notice in the future.

(3) If any doubtful or omitted parts are found in this manual, please inform Hitachi.

BEFORE USE

This product has been carefully manufactured, but you are kindly advised to make the following checks immediately after purchase. (Carefully handle the product, at the occasion of unpacking, not to apply impact or vibration to the product.)

(1) Check if the type meets the order specification.
(2) Check if there is any portion which was damaged during transport.
(3) Check for looseness of screws and for attachment of foreign matters.
(4) Check for any missing article among the packaged articles.

PACKAGED ARTICLES

Check in accordance with the following table if all the articles are complete in the package.

List of package articles of CPU module

No.	Description	Quantity	Remarks
1	CPU module (main unit)	1	
2	Key switch	2	
3	Instruction manual Hardware edition Software edition	1 copy per each	

CONTENTS

Chapter 1 I/O Signals ....1 - 1

Chapter 2 Concept of processing system and scanning system ....2 - 1

Chapter 3 Ladder / Command ....3 - 1

Chapter 1 I/O Signals

1.1 About I/O Signals

There are three types of I/O signals

1. External I/O area

   The PC transfers signals to and from external devices via I/O modules.

2. Internal I/O area

   Data is transferred in registers in the PC.

3. CPU link area

   Data is transferred to and from another CPU in CPU link.

   For these three signals, data is handled in bits, words (16 bits), or double words (32 bits).

Table 1.1 H-series PC I/O Table

		Bit data	Word data	Double-word data
External I/O area	Input	X□□□□□	WX□□□□	DX□□□□
	Output	Y□□□□□	WY□□□□	DY□□□□
Internal I/O area		R□□□□	——	——
		——	WR□□□□	DR□□□□
		M□□□□	WM□□□	DM□□□
CPU link area		L□□□□□	WL□□□□	DL□□□□

- An I/O number is assigned to □□□□ in the above table. For external I/O assignment (including remote external I/O assignment), see Section 1.5 "Rules for External I/O assignment".
- Bit data, word data, and double- word data on the same line in the above table have mutual relationship.
- The area for X, WX, and DX is shared by bit data and word data. This is also true for the area for Y, WY, and DY, the area for M, WM, and DM, and the area for L, WL and DL.

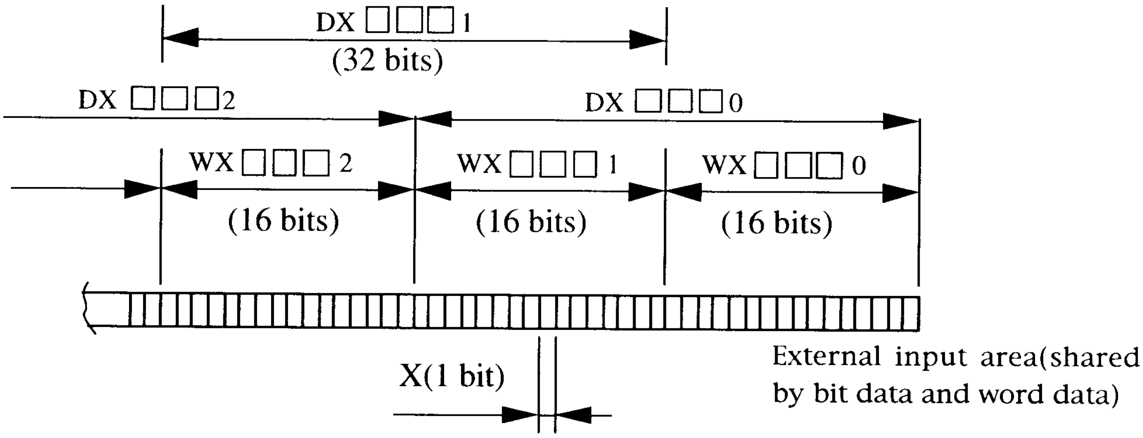


Figure 1.1 Relationship between X, WX, and DX (This figure also applies to relationship between Y, WY, and DY, relationship between M, WM, and DM, and relationship between L, WL, and DL.)

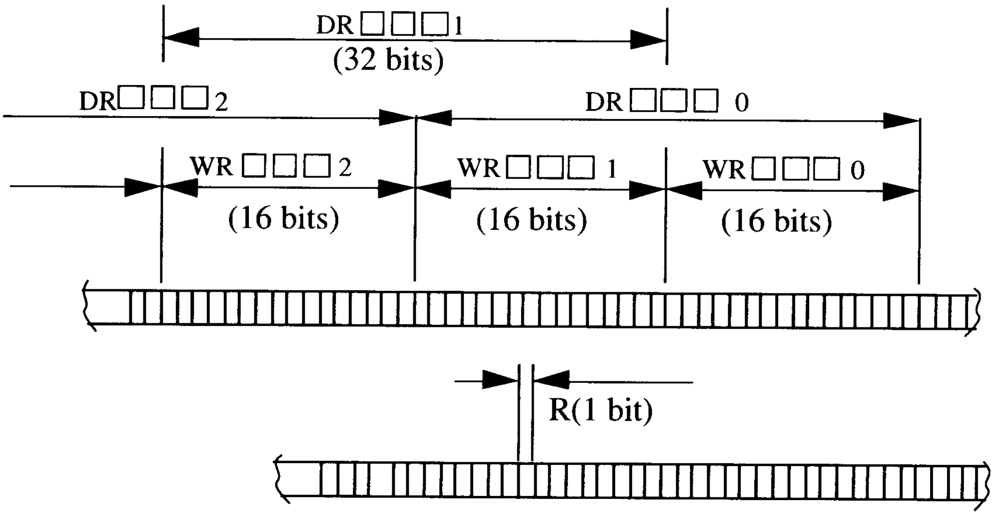


Figure 1.2 Relationship between R, WR, and DR

- R is allocated in the area separated from that for the WR and DR. In the area for WR, bit wise operation is not allowed.
- To perform bit wise operation of the internal output, use the area for M, WM, DM that can be used both for bit data and word data.
- In a system without the CPU link module (see Section 2.4 "CPU Link System" in the hardware manual) , the CPU link area (L, WL, and DL) can be used as the internal output area that is shared by bit data and word data.

1.2 I/O Assignment

Explanation of columns in the table below

Size column:	B : Bit W : Word (16 points) bit D : Double-word (32 points)
Point count column:	This column indicates the maximum number of points that can be used, when only bits are used. In word notation, the column indicates the maximum number of words that can be used when only words are used.
I/O assignment:	This column indicates the assignable range. For external I/O assignment, see Section 1.5 " Rules for External I/O Assignment."

Table 1.2 I/O Assignment

Function		symbol	Size	Name	Point count	I/O Assignment	Remarks
External I/O	External I/O	X	B	External bit input	928 points (58 words) (For H-250, 512 points (32 words))	X 0 u s b	u : Unit number (0 to 2) (H-250: 0 to 1) s : Slot number (0 to 9) in hexadecimal (H-250: 0 to 8) b : Intra-module bit number (00 to 95) in decimal m : Intra-module word number (0 to 7. For DX and DY, however, 0 to 6) Storage is not possible during power outage.
		Y	B	External bit output		Y 0 u s b	
		WX	W	External word input		WY 0 u s m	
		WY	W	External word output		WY 0 u s m	
		DX	D	External double-word input		DX 0 u s m	
		DY	D	External double-word output		DY 0 u s m	
	Remote external I/O	X	B	Remote external bit input	512 points (32 words) (Up to 128 points for one host station)	X r St s b	r : Remote host station number (1 to 4) St : Remote local station number (0 to 7) s : Slot number (0 to 8) b : Intra-module bit number (00 to 95) in decimal m : Intra-module word number (0 to 7. For DX and DY, 0 to 6) Storage is not possible during power outage.
		Y	B	Remote external bit output		Y r St s b	
		WX	W	Remote external word input		WY r St s m	
		WY	W	Remote external word output		WY r St s m	
		DX	D	Remote external double-word input		DX r St s m	
		DY	D	Remote external double- word output		DY r St s m	

Table 1.2 I/O Assignment(continued)

Function		symbol	Size	Name	Point count	I/O Assignment	Remarks
CPU link area		L	B	Bit CPU link area 1	16,384 points (1.024 words)	L 0 - 3FFF	Numbers are in hexadecimal. Storage is not possible during power outage. When the CPU is operated from STOP to RUN, however, storage will not be cleared.
		WL	W	Word CPU link area 1		WL 0 - 3FF	
		DL	D	Double-word CPU link area 1		DL 0 - 3FE	
		L	B	Bit CPU link area 2	16,384 points (1.024 words)	L 10000 - 13FFF	
		WL	W	Word CPU link area 2		WL 1000 - 13FF	
		DL	D	Double-word CPU link area 2		DL 1000 - 13FE	
Internal output	Internal bit output	R	B	Internal bit output	1,984 points	R0 - 7BF	Storage is possible during power outage. Numbers are in hexadecimal.
		R	B	Internal special bit output	64 points	R7C0 - 7FF	Storage is always possible during power outage. Numbers are in hexadecimal.
	Internal word output	WR	W	Internal word output	a. 1,024 words b 17,408 words	WR 0 - 3FF DR 0 - 3FE WR 0 - 43FF DR 0 - 43FE	a : H-250 b: H-252 *Note
		DR	D	Internal double-word output			Numbers are in hexadecimal. Storage is possible during power outage. If a range exceeding the capacity on the left is read, 0 is assumed.
		WR	W	Internal special word output	512 words	WR F000 - F1FF	Storage is always possible during power outage. Numbers are in hexadecimal.
		DR	D	Internal special double-word output		DR F000 - F1FE	
	Data area	M	B	Bit data area	16,384 points (1,024 words)	M 0 - 3FF	Storage is possible during power outage. Numbers are in hexadecimal.
		WM	W	Word data area		WM 0 - 3FF	
		DM	D	Double-word data area		DM 0 - 3FE	
Others	Edge detection	DIF	B	Rising edge detection	512 points	DIF 0 to 511	Storage is possible during power outage. Numbers are in decimal. Duplicated numbers cannot be used.
		DFN	B	Falling edge detection	512 points	DFN 0 to 511	Storage is possible during power outage. Numbers are in decimal. Duplicated numbers cannot be used.

*Note: These are divided into the two discontinuous areas WR0~3FF and WR400~43FF.

Table 1.2 I/O Assignment(continued)

Function		symbol	Size	Name	Point count	I/O Assignment	Remarks
Others	Master control	MCS	B	Master control set	50 points	MCS 0 - 49	Numbers are in decimal.
		MCR	B	Master control reset		MCR 0 - 49	
	Timer counter	TD	B	On-delay timer	512 points (256 points or less for the timer)	TD 0 - 255	Storage is possible during power outage. Numbers are in decimal. Use CTU, CTD, and CT up/down counters in combination. The same timer counter number must not be used twice or more. Only timers 0 to 63 can use 0.01 as the time base
		SS	B	Single-shot timer		SS 0 - 255	
		WDT	B	Watchdog timer		WDT 0 - 255	
		MS	B	Monostable timer		MS 0 - 255	
		TMR	B	Accumulation timer		TMR 0 - 255	
		CU	B	Up counter		CU 0 - 511	
		RCU	B	Ring counter		RCU 0 - 511	
		CTU	B	Up/down counter increment		CTU 0 - 511	
		CTD	B	Up/down counter decrement		CTD 0 - 511	
		CT	B	Up/down counter output		CT 0 - 511	
		CL	B	Elapsed count clear	512 points	CL 0 - 511	
		TC	W	Timer counter elapsed time	512 words	TC 0 - 511	
	Constant	-	W	Decimal word constant		0 - 65 535	Signed numbers from - 32,768 to 32,767
		-	D	Decimal double-word constant		0 - 4 294 967 295	Signed numbers from - 2,147,483,648 to 2,147,483,647
		H	W	Hexadecimal word constant		H0000 - HFFFF	
		H	D	Hexadecimal double-word constant		H00000000 - HFFFFFFFF	
		-	B	Bit constant		0 or 1	

1.3 I/O Storage during Power Outage

Usually, internal outputs are cleared when the PC starts operation. The internal outputs listed in Table 1.3 are retained unless the maximum number of points stored during power outage is exceeded. For operations, see the instruction manual for LADDER EDITOR SOFTWARE, and PGM peripherals.

Table 1.3 I/O Storage during Power Outage

I/O type	Maximum number of points stored during power outage	I/O range	Remarks
Internal bit output	1,984 points	R000 - 7BF	
Internal word output	1,024 words	WR000 ~ 3FF	H-250
	17,408 words	WR0000 - 43FF	H-252
Bit and word shared internal output	16,384 points (1,024 words)	M0000 - 3FFF (WM000 - 3FF)	
Rising edge detection	512 points	DIF 0 - 511	
Falling edge detection	512 points	DFN 0 - 511	
Timer/counter	512 points	TD 0 - 511	Elapsed time indicated by the timer/counter. TCO to TC511 are also retained.

Note) For TD, the following I/O are included
Timers : TD, SS, WDT, MS, TMR
Counters : CU, RCU, CTU, CTD, CT

1.4 Internal Special Outputs

(1) Functions

Internal special outputs are comprised of internal special bit outputs (R7C0 to R7FF) and internal special word outputs (WRF000 to WRF1FF). They have the following functions and purposes:

Function	Purpose	Example
[1] Control the system status.	Permit execution of CPU functions.	Remote run enabled (R7C3) Debug enabled (R7C5)
	Set conditions to start and stop CPU operation.	Continued processing during cycle time over (R7C0)
[2] Record the system status	Record the system status	Self-diagnostic error (R7DB) Self-diagnostic error details (WRF000)

In item [1] above, the CPU always checks the contents of internal special outputs during operation. For this reason, the user can forcibly set or reset internal special output data to permit execution of CPU functions, set operation conditions, and perform other functions.

In item [2] above, the CPU always records the system operation status in internal special outputs. The user can check the system status by monitoring the appropriate internal special output.

Notes;

1) Check the set and reset conditions before setting data in internal special outputs. Particularly, do not set data in the areas where the system sets data.

2) Do not use the internal special bit outputs (R7C0 to R7FF) and internal special word output (WRF000 to WRF1FF) as normal outputs (coils and arithmetic boxes).

3) If the I/O number exceeds usefull range in H-250/252 then the I/O number will be transfered as bellow.

H-250			H-252		
Bit	→	L13FFF	Bit	→	R7FF
Word	→	WL13FF	Word	→	WRF019
Double word	→	DL13FE	Double word	→	DRF018

(2) Internal special outputs

I/O No.	Contents	ON	OFF
R7C0	Continue processing during cycle time over (normal scan)	U	U
R7C1	Continue processing during cycle time over (periodic scan)		
R7C2	(Undefined)	-	-
R7C3	Remote Run enabled	U	U
R7C4	Remote Stop enabled		
R7C5	Debug enabled		
R7C6	(Undefined)	-	-
R7C7	Enabled modification during RUN	U	U
R7C8	Serious failure	S	
* R7C9	Sequence processor error		
R7CA	User memory error		
R7CB	(Undefined)	-	-
R7CC	Memory size over	S	U
R7CD	(Undefined)	-	-
R7CE	(Undefined)		
R7CF	(Undefined)		
R7D0	(Undefined)		
R7D1	Congestion error (normal scan)	S	U
R7D2	Congestion error (periodic scan)		
R7D3	(Undefined)	-	-
R7D4	Syntax or assemble error	S	U
R7D5	(Undefined)	-	-
* R7D6	Excessive number of assignment points in I/O module	S	U
* R7D7	Communication module error		
R7D8	(Undefined)	-	-
R7D9	Battery error	S	S
R7DA	(Undefined)	-	-
R7DB	Self-diagnostic error	S	U
R7DC	(Undefined)	-	-
R7DD	(Undefined)		
* R7DE	Link module error	S	U
R7DF	(Undefined)	-	-
R7E0	Mode key switch in STOP position	S	S
R7E1	Dip switch in the position "4"		
R7E2	Mode key switch in RUN position		
R7E3	Single scan ON after RUN		
R7E4	Always ON		X
R7E5	0.02 - s clock		S
R7E6	0.1 - s clock		
R7E7	1 - s clock		
R7E8	Occupancy flag		
R7E9	Disabled RUN		
R7EA	Modification during RUN		
R7EB	Error LED display clear	U	
R7EC	Internal special erroneous output clear		
R7ED	(Undefined)	-	-
R7EE	(Undefined)		
R7EF	(Undefined)		
R7F0	Carry flag (CY)	S	S
R7F1	Overflow (V)		
R7F2	Shift data (SD)	U	U
R7F3	Calculation error (ERR)	S	S
R7F4	Data error (DER)		
R7F5	(Undefined)	-	-
R7F6	(Undefined)		
R7F7	(Undefined)		
R7F8	Request to read calendar clock	U	S
R7F9	Request to set calendar clock		
R7FA	Request to adjust calendar clock		
R7FB	Incorrect calendar clock setting	S	
* R7FC	Trigger condition matching flag		
* R7FD	Trace monitor flag		
R7FE	(Undefined)	-	-
* R7FF	Converted I/O number in the case of bit I/O No. error		

The user must operate U.
S is controlled by systems. The user must not control S and X.
* : Effective only for H-252.

I/O No.	Contents	ON	OFF
WRF000	Self-diagnostic error code	S	U
WRF001	Syntax and assembler error details		
WRF002	(Undefined)	-	-
WRF003	(Undefined)		
WRF004	Invalid communication module slot number	S	U
WRF005	(Undefined)	-	-
WRF006	(Undefined)		
WRF007	Invalid link module slot number	S	U
WRF008	Error circuit number		
WRF009	(Undefined)	-	-
WRF00A	(Undefined)		
WRF00B	Calendar clock reading data (year)	S	S
WRF00C	Calendar clock reading data (month, day)		
WRF00D	Calendar clock reading data (a day of the week)		
WRF00E	Calendar clock reading data (hour, minute)		
WRF00F	Calendar clock reading data (second)		
WRF010	Maximum scan time		
WRF011	Current scan time		
WRF012	Minimum scan time		
WRF013	CPU status		
WRF014	Internal word output capacity		
WRF015	Calculation error code		U
WRF016	Residue register (lower order)		S
WRF017	Residue register (upper order)		
* WRF018	Converted I/O number in the case of double word I/O No. error		
* WRF019	Converted I/O number in the case of word I/O No. error		
WRF01A	(Undefined)	-	-
WRF01B	Calendar clock reading or setting data (year)	U/S	S
WRF01C	Calendar clock reading or setting data (month, day)		
WRF01D	Calendar clock reading or setting data (a day of the week)		
WRF01E	Calendar clock reading or setting data (hour, minute)		
WRF01F	Calendar clock reading or setting data (second)		
WRF020 WRF021	Status, communication module slot 0 status	S	
WRF022 to WRF031	Status, communication module slot 1 to 8 status		
WRF032 WRF033	Status, communication module slot 9 status		
WRF034 to WRF03F	(Undefined)	-	-
WRF040 WRF041 WRF042	Member registration area 1	S	S
WRF043 WRF044 WRF045	Member registration area 2		
WRF046 WRF047 WRF048	Member registration area 3		
WRF049 WRF04A WRF04B	Member registration area 4		
WRF04C WRF04D WRF04E	Debug registration area		
WRF04F to WRF0DF	(Undefined)	-	-
WRF0E0 to WRF13F	Link - 1 error flag	S	S
WRF140 to WRF19F	Link - 2 error flag		
WRF1A0 to WRF1FF	(Undefined)	-	-

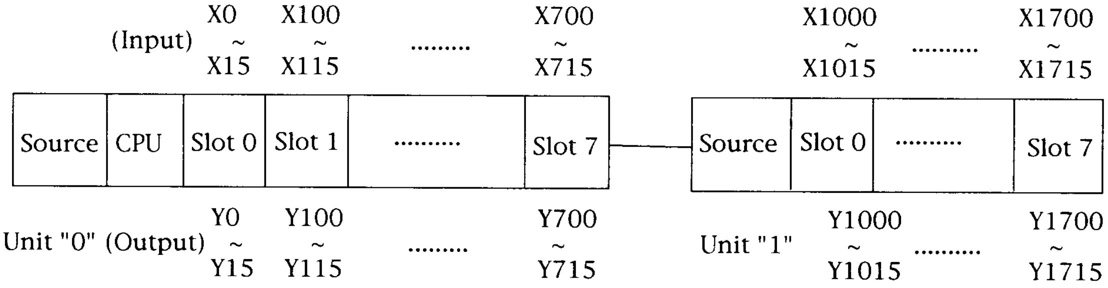
* and WRF020 to WRF1FF are effective only for H-252.

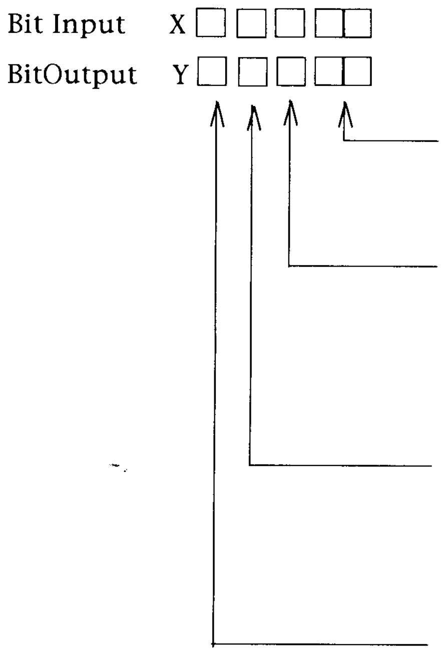
1.5 Rules for External I/O Assignment

Assignment of external inputs and outputs is determined depending on I/O classification, discrimination among basic, expansion and remote I/O, mounting slot position, and bit word number in module

(1) Assignment of the bit modules

Assignment of the bit modules is shown bellow.
X0~X15 for the case the 16-point input is assigned to the slot "0".
Y100~Y115 for the casethe 16-point input is assigned to the slot "1".





Bit numbers in the I/O module:
0 to 95 (decimal)
16-point module: 00 to 15
32-point module: 00 to 31

Slot position on the base
(hexadecimal)
- Basic base: 0 to 8
- Expansion base
  In remote mode: 0 to 7
  (BSH-3, 5, 7 & 10 are not usable) *1
  In non-remote mode : 0 to 9(0 to 7 for H-250)

Unit number or remote local station number
- Basic base: 0
- Expansion base
  In remote mode: 0 to 7
  In non-remote mode: 1to 2 (1 for H-250)

Remote host station number
In remote mode: 1 to 4
In non-remote mode: 0

*1. Local station RIOH-TL can not be used in the sophisticated function base.

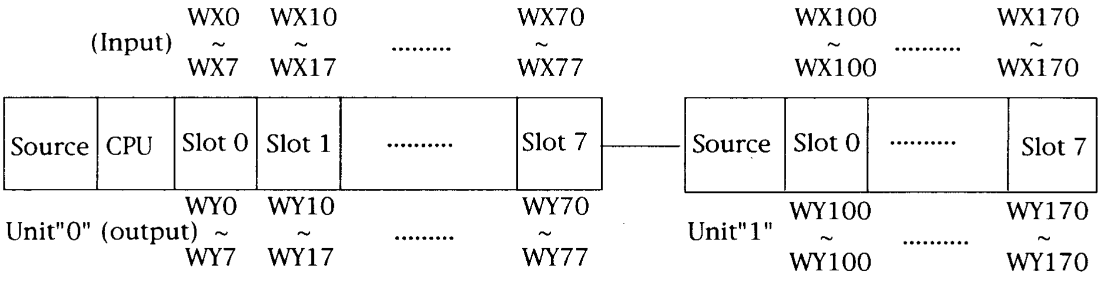
Caution

Assignment of the mixed I/O modules are as follows.

Type	Input number	Output number
PHH-DT	8 input points X0~X7	8 output points Y24~Y31
PHM-TT	16 input points X0~X15	16 output points Y16~Y31

(2) Assignment of the word modules

Assignment of the word(16-bit) modules is shown bellow.
WX0~WX7 for the case the word input module is assigned to the slot "0".
WY10~WY17 for the case the word input module is assigned to the slot "1".



Word Input WX □□□□
Word Output WY □□□□

- Word number: 0 to 7
- Slot position on the base (hexadecimal)
  - Basic base: 0 to 8
  - Expansion base
    In remote mode: 0 to 7 *1
    (BSH-3, 5, 7 & 10 are not usable)
    In non-remote mode : 0 to 9 (0 to 7 for H-250)
- Unit number or remote local station number
  - Basic base: 0
  - Expansion base
    In remote mode: 0 to 7
    In non-remote mode: 1 to 2 (1 for H-250)
- Remote host station number
  In remote mode: 1 to 4
  In non-remote mode: 0

Bit number in the I/O module	Word number
00 - 15	0
16 - 31	1
32 - 47	2
48 - 63	3
64 - 79	4
80 - 95	5
[96 - 111]*	6
[112 - 127]*	7

* Bits 96 to127 cannot be accessed in bit units. They can be accessed only in word units.

*1 Local station RIOH-TL can not be used in the sophisticated function base.

•Word I/O modules of H-252/H-250 are shown in the table below. Word I/O modules can be installed on the basic base and the expansion base, but can not be installed on the remote end.

Part name	Type	Input(WX)	Output(WY)	Specification
Analog input	AGH-I	WX0~WX7	-	8-quantity, 4~20mA, 8 bits
	AGH-IV			8-quantity, 0~10V, 8 bits
	AGH-IV2			8-quantity, 4~20mA, 0~10V, 12 bits
Analog output	AGH-O	-	WY0~WY3	4-quantity, 4~20mA, 8 bits
	AGH-OV			4-quantity, 0~10V, 8 bits
	AGH-OD	-	WY0, WY1	2-quantity, 4~20mA, 8 bits
	AGH-ODV			2-quantity, 0~10V, 8 bits
Counter	CTH	WX0, WX1	WY2~WY7	2-phase, 10kHz
32-point input	PIH-DM	WX0, WX1	-	DC input
32-point output	POH-TM	-	WY0, WY1	Transistor output
Thermo-couple input	THH	WX0~WX7	-	8-quantity, Type J, K

•WX and WY can be used also for bundling or outputting data of 16 bits in the bit module.

(3) Double- word I/O number

Double- wordInput DX □□□□
Double- word Output DY □□□□

Word number: 0 to 6

Slot position on the base (hexadecimal)
- Basic base: 0 to 8
- Expansion base
  In remote mode: 0 to 7 *1
  (BSH-3, 5, 7 & 10 are not usable)
  In remote mode :
  0 to 9(0 to 7 for H-250)

Unit number or remote local station number
- Basic base: 0
- Expansion base
  In remote mode: 0 to 7
  In non-remote mode: 1

Remote host station number
In remote mode: 1 to 4
In non-remote mode: 0

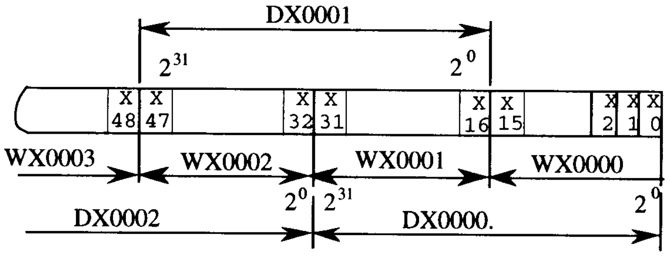
Bit number	Word number	Double-word number
00 - 15	0	0
16 - 31	1	1
32 - 47	2	2
48 - 63	3	3
64 - 79	4	4
80 - 95	5	5
[96 - 111]*	6	6
[112 - 127]*	7	

*1 Local station RIOH-TL can not be used in the sophisticated function base.

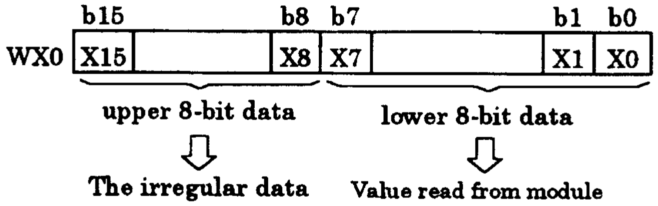
* Bits 96 to127 cannot be accessed in bit units. They can be accessed only in word units.

- Relation between bits, words, and double- words (areas shared by bits and words)

In the following example, X, WX, and DX are used.



When the word data is read from eight points input module (PIM-D, PIM-A), 16 points I/O mixture module (PHH-DT), and 8-bit analog input module (AGH-IV, AGH-I), upper 8-bit (for instance, X8-X15 in case of 0 slot) becomes irregular as follows. Do upper 8-bit of the program example which shows in the following in masking and give upper 8-bit as all "0".

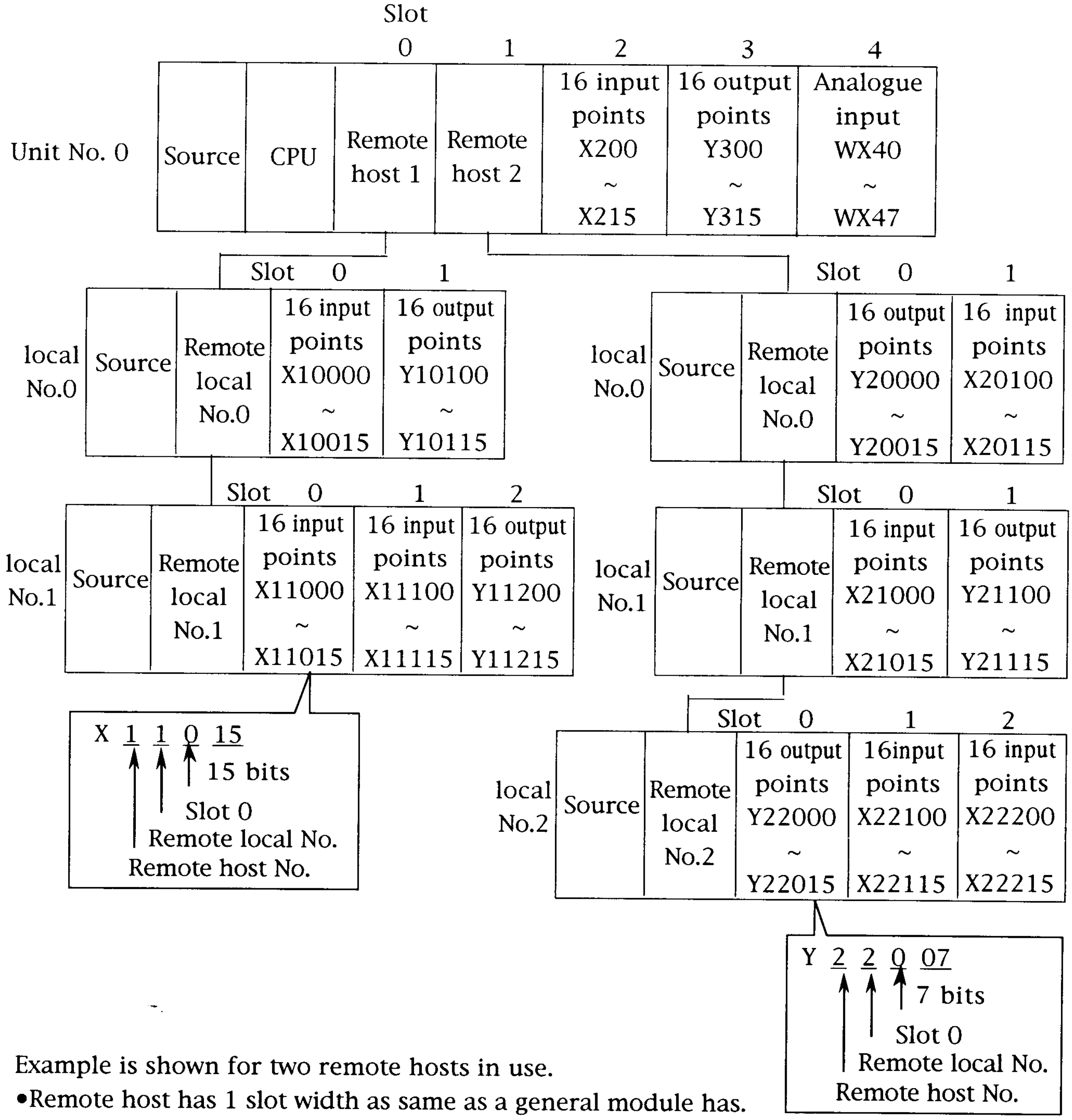


[Program example]

```
WR0 = WX0 AND H00FF
```

···· By masking Upper 8-bit is assumed to be all "0".

1.6 Example of Assignment of Remote I/Os and Analog ModulesRules



Example is shown for two remote hosts in use.

- Remote host has 1 slot width as same as a general module has.
- 4 remote hosts(Remote host No.1~4) can be installed.
- Analogue inputs are installed in the slot No.4 of the basic base(Unit No.0).
  Their input numbers are WX40~WX47. Analog input module has 8-bit configuration. Upper 8 bits in the register are all "0". The register configuration is shown bellow.

	b15							b8	b7 b0
Channel 0	0	0	0	0	0	0	0	0	Data WX0
1	0	0	0	0	0	0	0	0	Data WX1
2	0	0	0	0	0	0	0	0	Data WX2
3	0	0	0	0	0	0	0	0	Data WX3

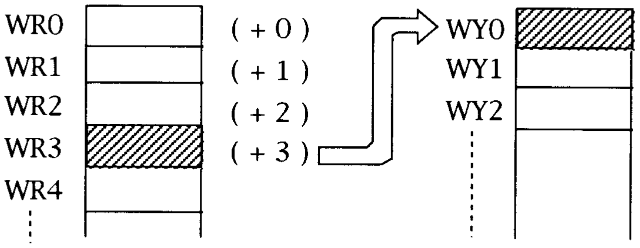
1.7 How to Use Array Variables

An array variable specifies I/O numbers by adding a constant or word I/O (WR, WM, and WL) as an index to I/O (R, M, L, WR, WM, WL, TC, DR, DM, DL). Array variables can be used only for commands expressed by substitution formulas.

(1) Expression of array variables.

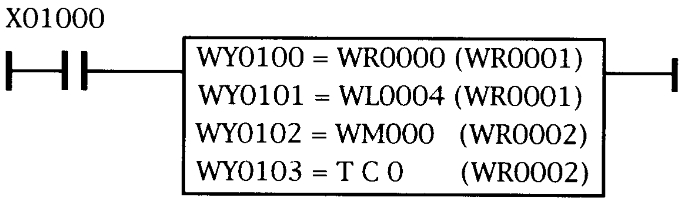
An array variable is expressed by the form ☐a (b). ☐a is called "I/O of an array variable", and " b " in the parentheses is called "contents of index". When " b" is a constant, ☐a(b) means " ☐a + b ".

An example expressing " WY0 = WR0(3) " is shown below.



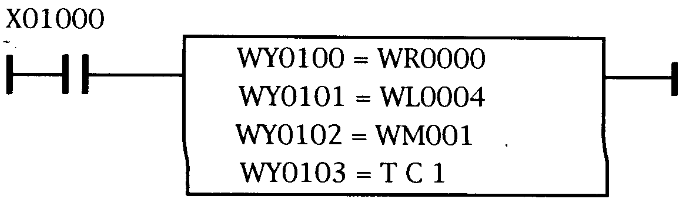
Explanation on WY0= WR0(3)

(2) When WR, WL, WM and TC are used for I/O of array variables



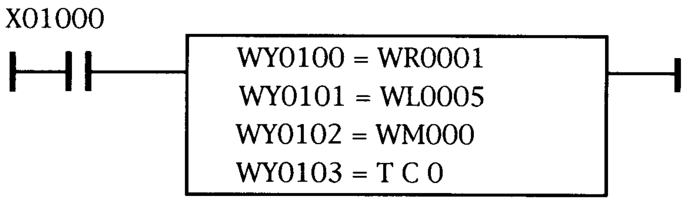
```
WY0100 = WR0000 (WR0001)
WY0101 = WL0004 (WR0001)
WY0102 = WM000  (WR0002)
WY0103 = T C 0  (WR0002)
```

(a) Equivalent program for WR0001 = 0, WR0002 = 1



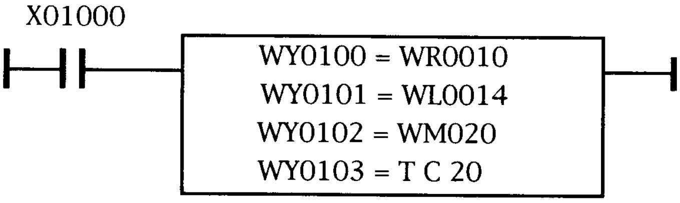
```
WY0100 = WR0000
WY0101 = WL0004
WY0102 = WM001
WY0103 = T C 1
```

(b) Equivalent program for WR0001 = 1, WR0002 = 0



```
WY0100 = WR0001
WY0101 = WL0005
WY0102 = WM000
WY0103 = T C 0
```

(c) Equivalent program for WR0001 = 10, WR0002 = 20



```
WY0100 = WR0010
WY0101 = WL0014
WY0102 = WM020
WY0103 = T C 20
```

Array variables using the above WR, WL, WM and TC specify I/O numbers obtained by adding the contents of index (value in parentheses) to the directly specified I/O numbers.

Chapter 2 Concept of processing system and scanning system

2.1 I/O Processing system

The direct mode and the refreshing mode are prepared for I/O processing system of H-250/H-252. Operate a controller with the dip switch set in either of the two modes.

Caution

① I/O processing for word modules (32 points I/O, Analogue, Counter, Link, Remote, Positioning, Serial I/O, Thermo-couple) will be performed in the simultaneous refreshing mode, even when the dip switch is set in the direct mode.

The following description explains the differences between the refreshing mode and the direct mode.

Refresh Processing System

Before computational processing, the ON/OFF status of external input is fetched in the image memory. Even if the status of external input changes during computational processing, the input status in the image memory remains unchanged. Change in the external input is to be fetched at the input processing of the next scan. Along with computational processing, each status of external output, internal output, etc. changes sequentially on the image memory.

After computational processing, the ON/OFF status of external output on the image memory is output to the output circuit.

Therefore, if input X0 turns ON immediately after input processing, it is fetched at the next processing (point a in the figure below) and output Y100 turns ON at the output processing (point b). This signifies that a time period for 2 scans at maximum is taken between input and output.

Direct Processing System

At every computation, the ON/OFF status of external input is fetched and the ON/OFF status of external output is output to the output circuit.

On execution of an instruction (at point c), the ON status of input X0 is fetched and output Y100 turns ON. Therefore, a time period for only 1 scan is needed from input to output.

As compared above, the direct system can be said to provide a better response than the refresh system.

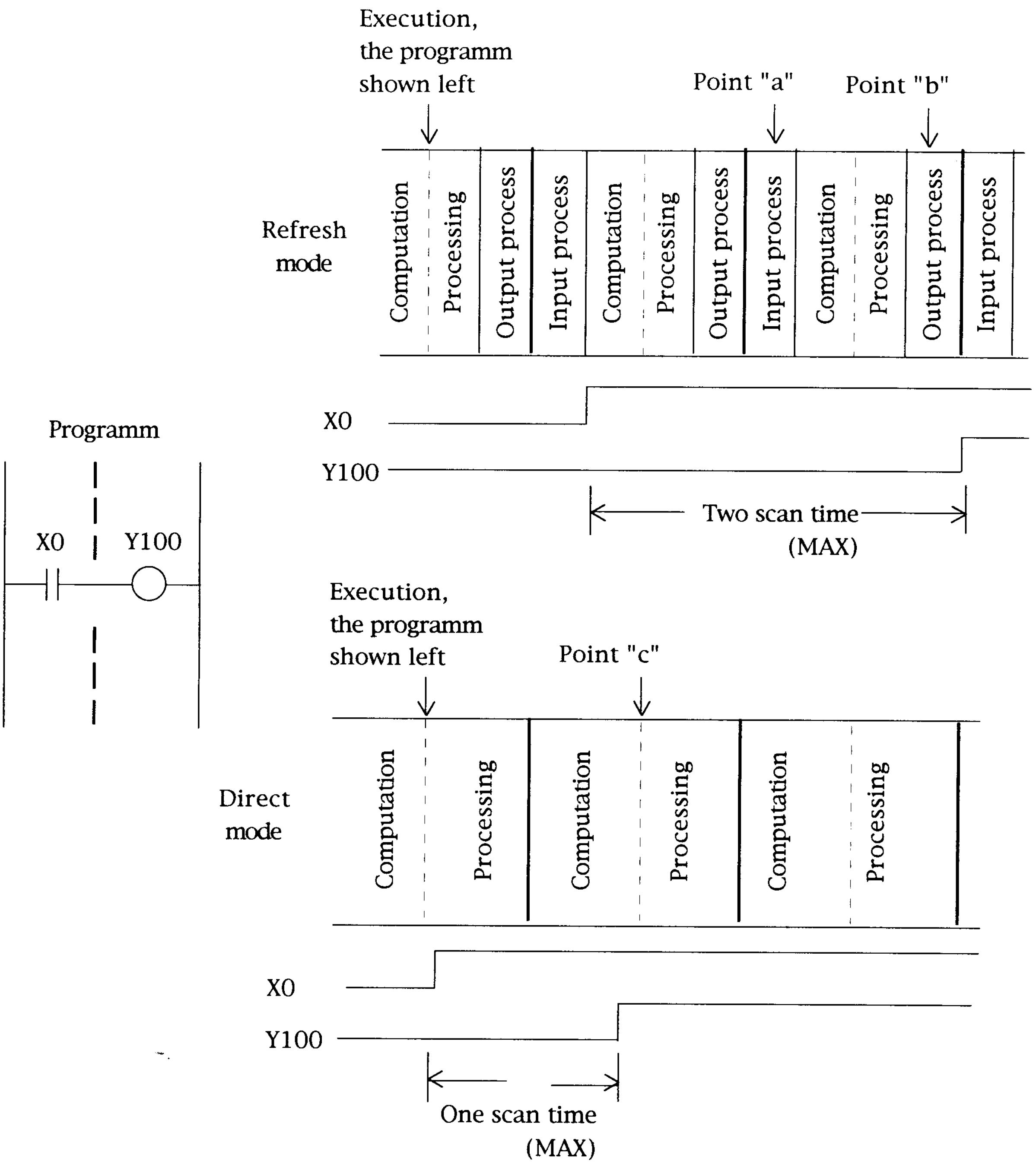


Figure 2.1 Differences between the two modes

2.2 Concept of Scan

(1) Definition and Function

H-252/250 is provided with the two types of scan operations, the normal scan and the periodic scan .

(a) Normal scan

The normal scan is a group of executions of a Ladder or command program (excluding interruption scan program), and a scan END processing initiated from an END command.

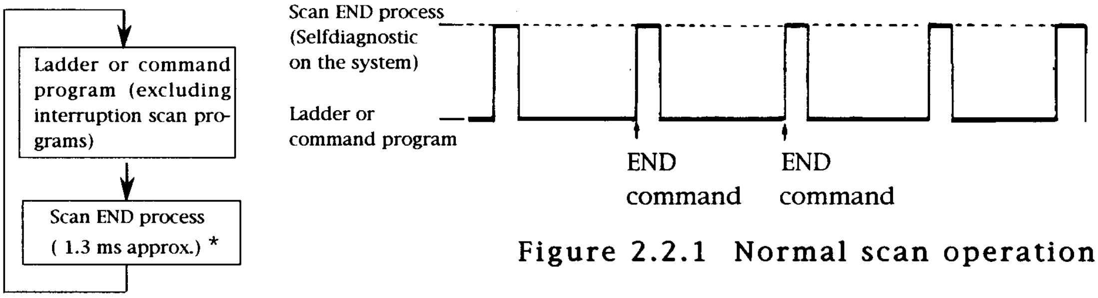


Figure 2.2.1 Normal scan operation

* This is the time in the refresh mode. (In the direct mode, the time of H-252 is 32µs approximately, the time of H-250 is 180µs approximately.)

(b) The periodic scan

The periodic scan executes interruption programs (periodic scan program) at the cycles designated by the user (10ms, 20 ms, 40ms) during PC is in operation.

Locate the periodic scan program between commands INT 0 and RTI in case of 10ms periodic application, and between INT 1 and RTI in case of 20ms application.

The periodic processing of the system is executed at 10ms, irrespective of existence of other periodic scan programs.

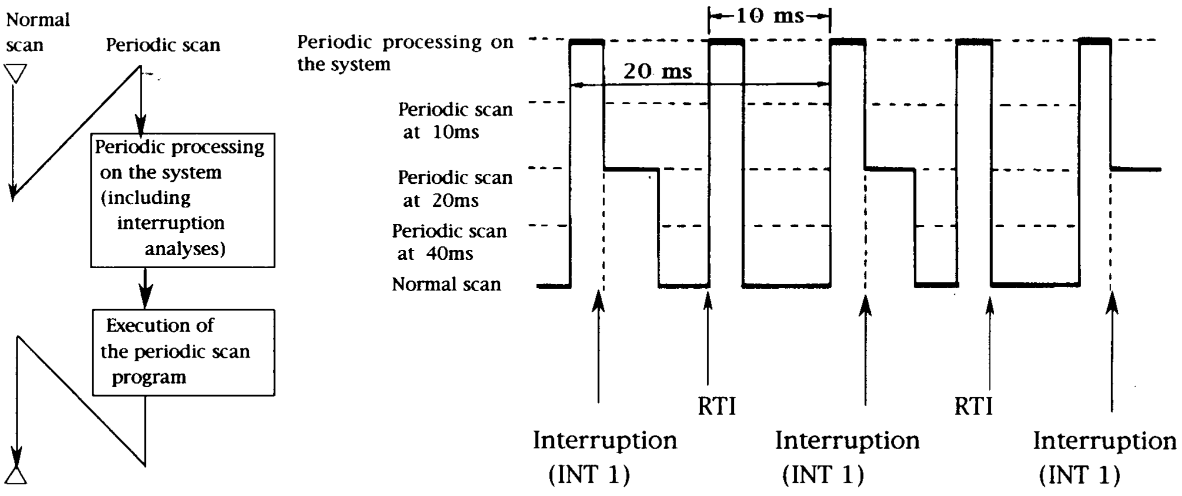


Figure 2.2.2 Periodic scan operation (20ms cycle specified for example)

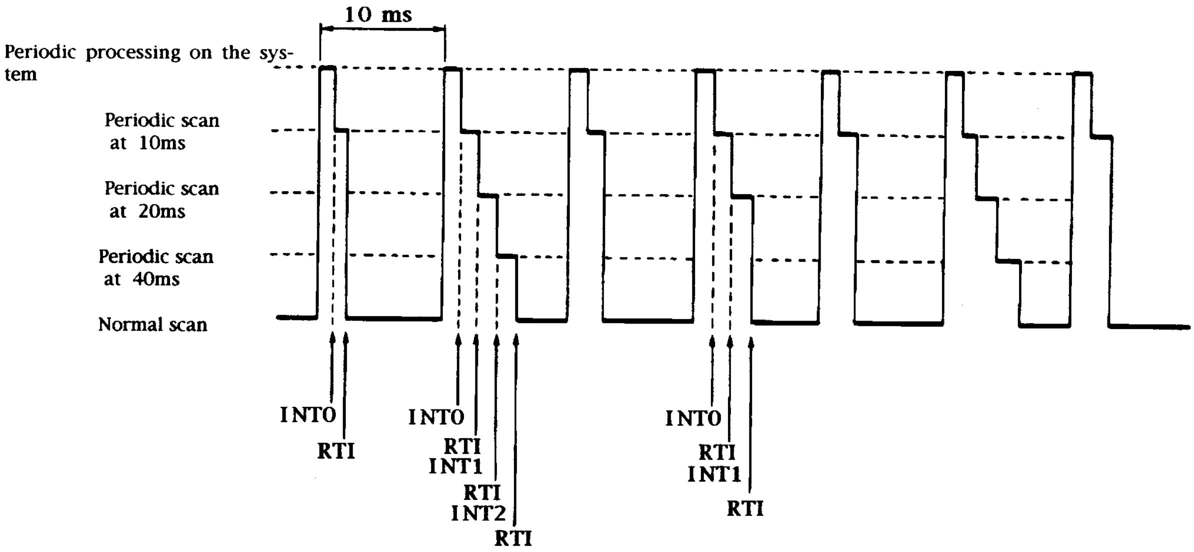
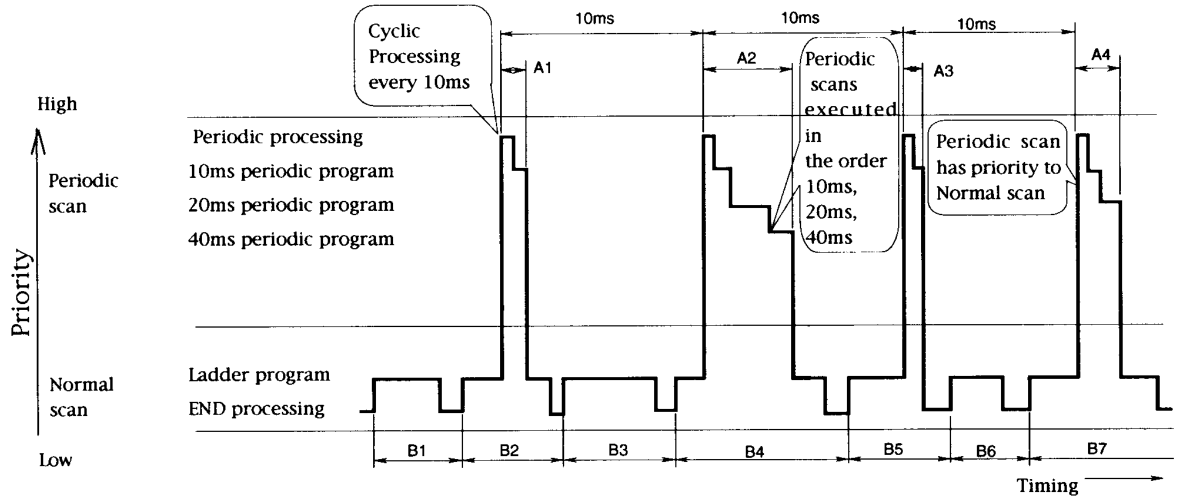


Figure 2.2.3 Periodic scan operation (combination of various periodic scan)

(2) Scan and Interruption

CPU modules H-250/H-252 are operated in the two scan modes, the normal scan and the periodic scan. Progression and contents of scan executions are illustrated in Figure 2.2 bellow.



An : Time of the periodic scan   Bn : Time of the normal scan

Figure 2.2.4 Relationship between various scans

Point

Scan as defined here means the execution of a program, from a start to an end.

A time period taken for each scan is termed a scan time. In Fig. 2.2.4 suppose that the periodic scan time indicated by A2 is longer than 10 msec (interval of periodic processing). In this case, the next periodic scan starts before the end of the first periodic scan processing. This phenomenon is called a periodic scan congestion error. With regard to normal scan, the number of periodic processings is counted within a time period from one normal scan to the next normal scan. In case the time determined by the periodic processing count × periodic processing interval (10 msec) exceeds the specified time (100 msec if unspecified), a normal scan congestion error occurs. If a congestion error occurs on either periodic scan or normal scan, the CPU module stops program execution (operation stops).

Next, the relation between user program and each scan is shown in Fig. 2.2.5

Point

- ○ Time period for congestion error check of normal scan: 100 msec (specifiable within 10 to 2550 msec with programming device).
- ○ Time period for congestion error check of periodic scan equals the interval of the shortest periodic scan under use. (An error occurs if the total execution time of all periodic scans exceeds the interval of the shortest periodic scan.)

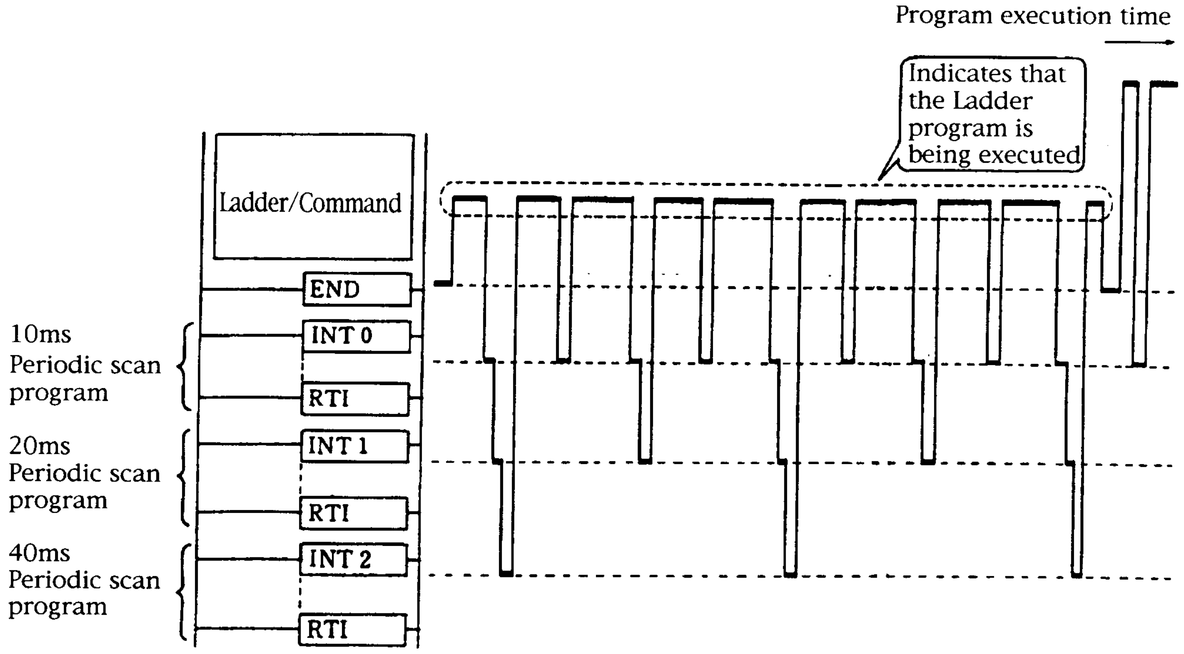


Figure 2.2.5 Relationship between user's program and various scans

In Figure 2.2.5, the structure of the user's program memory is shown as a case that utilize all of the functions, Ladder/Command and periodic program. In actual operation, a user will use various arrangements depending on various operational requirements.

(3) PID control

H-252 is able to execute PID processing and the sequence processing in parallel, as shown in FIGURE 2.2.6.

H-252 processes PID operation once in each 20ms cyclic scan.

In the operation, the CPU ,

① Reads the processed value (PV) and the set value(SV),

② Executes PID operation,

③ Makes the output (MV) as the result of the operation,

Executes the normal scan during 20ms scan intervals,

④ Executes the parallel operation of sequense controls.

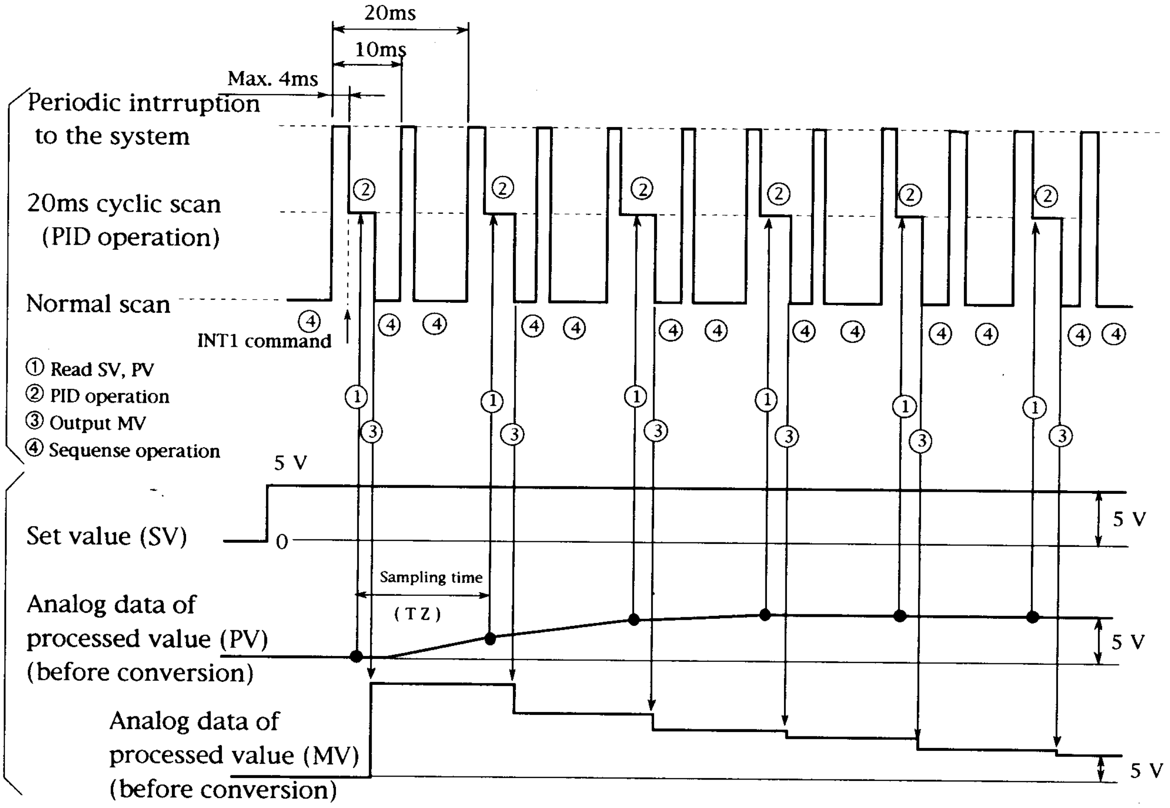


Figure 2.2.6 20ms cyclic scan operation (a) by H-252, and state of variations in outputs caused by PID operation.

As shown above, the subject processing value of PID control is sampled at (PV), and the output (MV) is reset, every 20ms. The sampling interval is called " sampling time" (TZ). This time setting is adjustable from the minimum of 20ms in H-252 as shown in Figure 2.5, to the maximum of 4s at every 20ms, by means of the PID group management command as will be explained in the later part of this edition.

(Notes) In Figure 2.2.6 ① and ② (read and write of analog signals) are assumed without any time lag for simplicity of explanations. The following time laggings will exist in actual systems.

Maximum 5ms: When the analog modules in use for PID are equipped on the basic board and the expansion board.

(4) Supervising the congestion

The congestion error is such that the actual execution time of a sequence control system exceeds a pre-set value (100ms, as the default value), as supervised by the actual execution time metering device. In this status, the PC will be stopped or continued to operate depending on settings of the special internal output (R7C0 ~ R7C).

The supervision of a congestion is performed at every scan (the normal scan, the periodic scan).

① Supervising the normal scan.

When the execution time of one scan cycle exceeds the congestion check time as pre-set by parameters, it will be assumed as a congestion error, and the special internal output R7D1 will be ON.

When the internal special output R7C0 is OFF, PC will be stopped.

(Caution) In the refresh mode avoid assembling a lengthy loop program. In case the loop is lengthy, the last circuit (END command) will not be executed, which will cause the I/O processing to be disabled and therefore the response time of the I/O will be prolonged. Therefore, set the congestion check time in the refresh mode whithin a half of the required response time.

② Supervising the periodic scan.

When the execution time of the periodic scan (the time from the periodic interruption to RT1) exceeds the shortest cycle time (10ms when INT0 exists, 40ms when INT2 alone is used), it is handled as a congestion error, and the special internal output R7D2 will be ON.

When the internal special output R7C1 is OFF, PC will be stopped.

(Notes) If, during the congestion error status of the periodic scan , operation will be continued, the periodic scan processing will be halted and re-started from the beginning of the periodic scan.

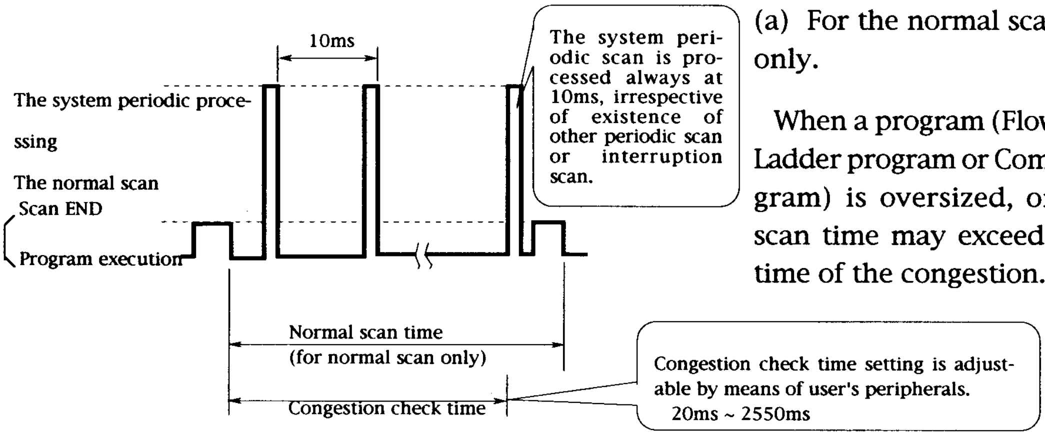
If such status is repeated, the normal scan will not be executed and the congestion error will be occurred in the normal scan.

(5) Causes of congestion errors

① In the normal scan.

The following three cases are considered . Cautions should be made in designing a program, to make the sum of scan time

within the check time of the congestion, in case simultaneous executions is designed for a periodic scan .

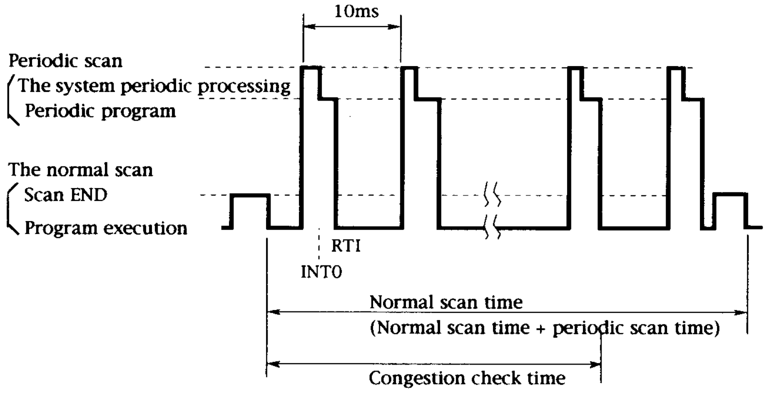


The system periodic scan is processed always at 10ms, irrespective of existence of other periodic scan or interruption scan.

Congestion check time setting is adjustable by means of user's peripherals.
20ms ~ 2550ms

(a) For the normal scan program only.

When a program (Flow program, Ladder program or Command program) is oversized, one normal scan time may exceed the check time of the congestion.



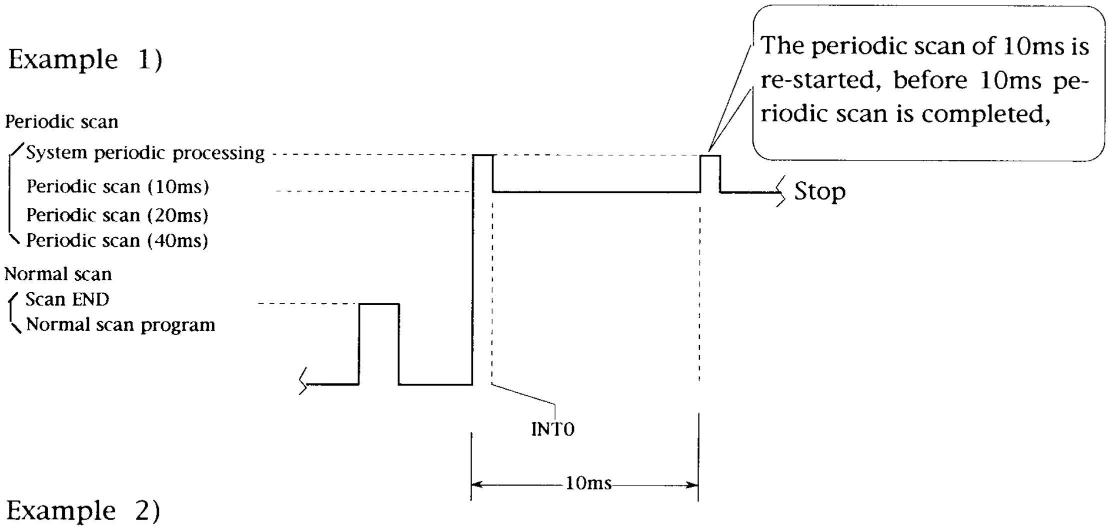
(b) For the normal scan program + the periodic scan program.

The normal scan time has a chance to be prolonged caused by an execution of the periodic scan program.

② In the periodic scan.

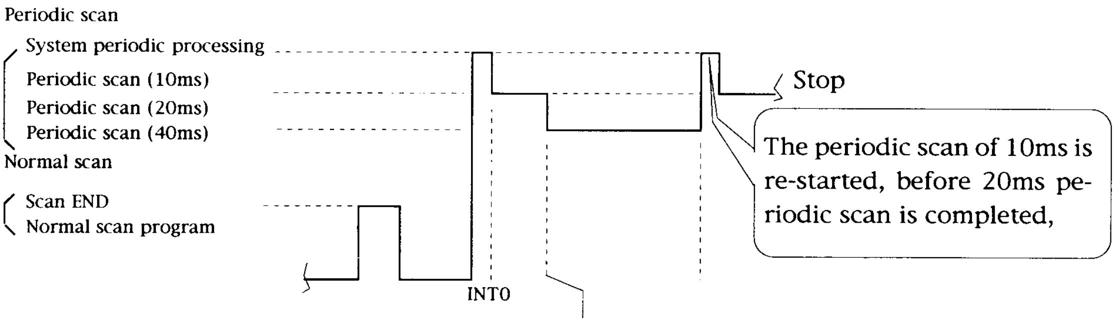
During three periodic scans, of 10ms, 20ms and 40ms, are in parallel, the scanning operation will be stopped , if 10ms scan is re-started before all the three scans are completed.

Example 1)



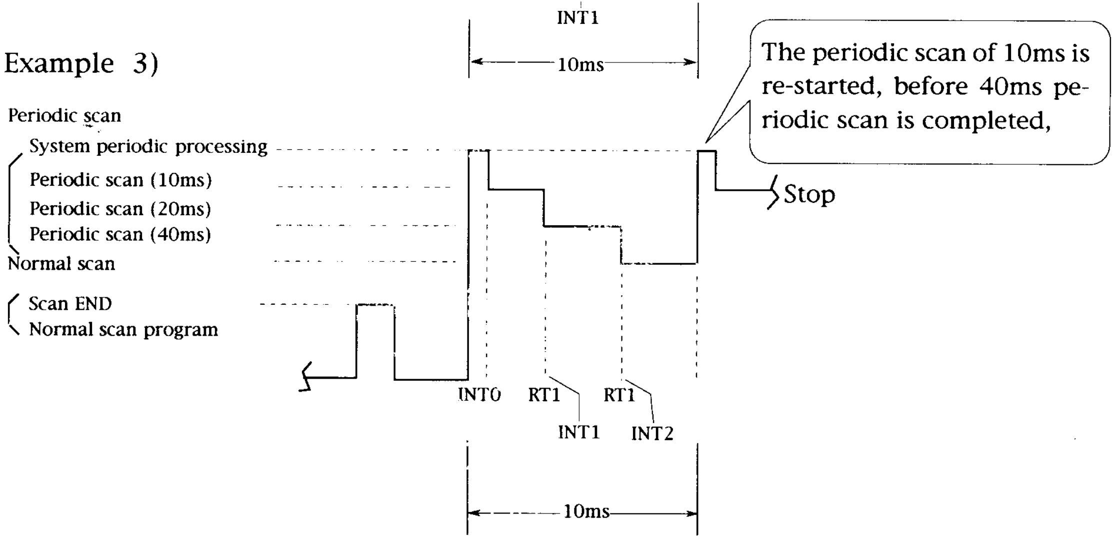
The periodic scan of 10ms is re-started, before 10ms periodic scan is completed,

Example 2)

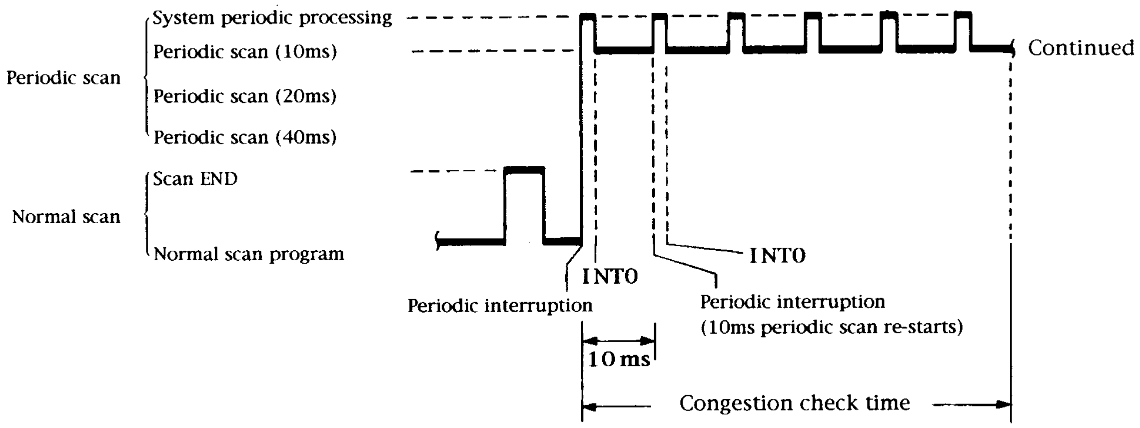


The periodic scan of 10ms is re-started, before 20ms periodic scan is completed,

Example 3)



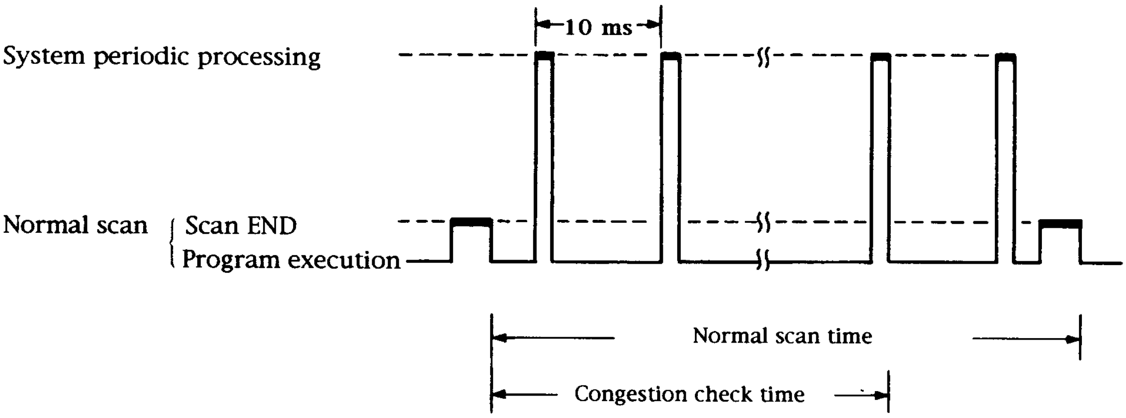
The periodic scan of 10ms is re-started, before 40ms periodic scan is completed, In another case in which the internal bit output R7C0 is ON, only the periodic scan will continue execution, with a congestion error containing in the periodic scan. It should be noted that the normal scan cannot be executed in such case.



(6) Continuation of the operation after a congestion

① In the normal scan congestion.

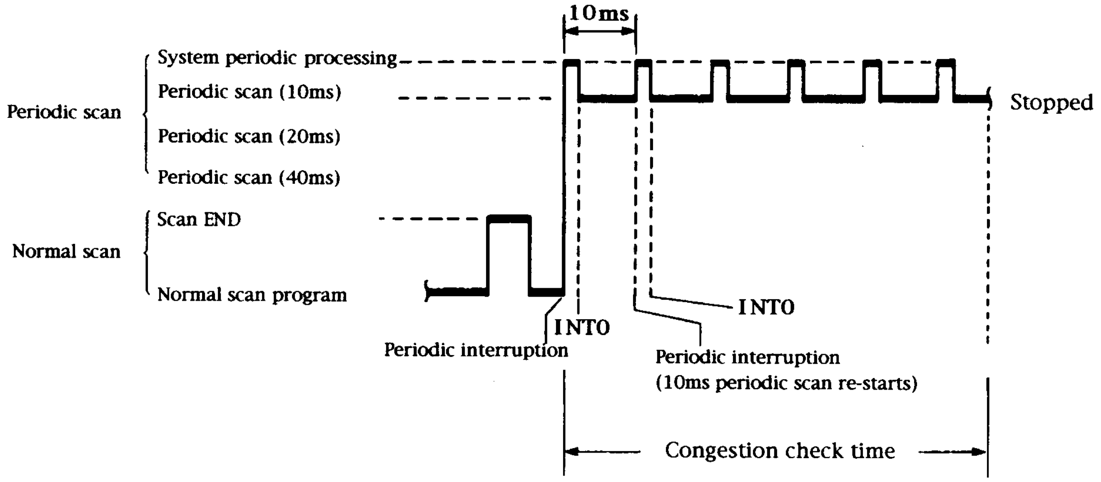
When the normal scan congestion error occurs, and at the same time, the special internal bit output R7C0 is ON to continue the processing after the error, the normal scan will complete execution to END irrespective of the congestion check time and will again start the normal scan at the top of the program.



It should be noted that the normal scan will not be stopped by the congestion error (the scan END processing will not be executed), even when an endless loop is formed by JMP command in the normal scan.

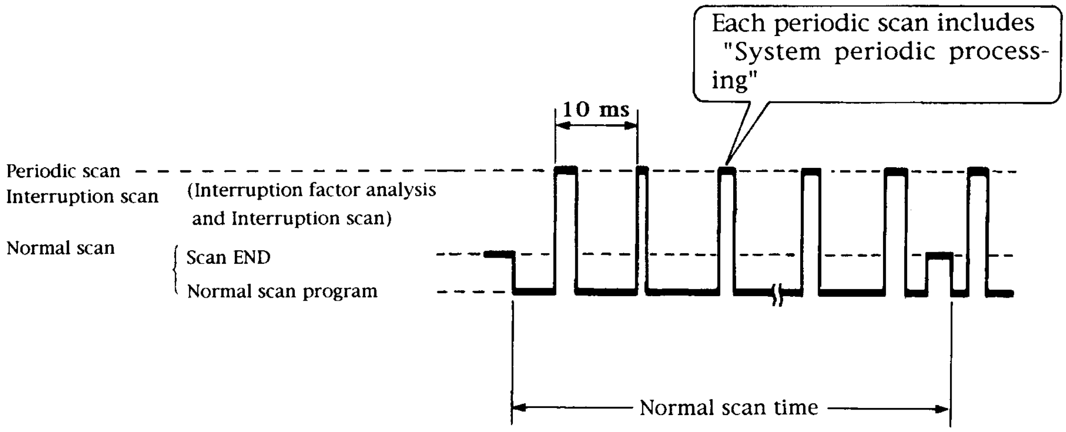
② In the periodic scan.

When the periodic scan congestion error occurs, and at the same time, the special internal bit output R7C1 is OFF to continue the processing after the error, the periodic scan will be stopped and be re-started at the top. In this case, when the signal of process continuation is OFF, this scan will be stopped because of an periodic congestion error.



(7) About the normal scan time

① Definition of the normal scan time.



The normal scan time is defined as the time required to execute one scan which starts from the beginning of the normal scan program and ends at its END operation. This time also includes the time required to execute an interruptive system periodic scan. The system periodic scan is included in the periodic scan and is continuously, at every 10ms irrespective of the periodic scan execution, applying interruptions that will perform interruption factor analyses, timer renewals, I/O re-write, etc. Accordingly, the time of normal scan includes a combination of times for the system periodic processing, even when no periodic scan is provided.

② Details of various scan times.

Table 2.2 Scan type vs Processing time

Scan type / CPU type	Normal scan		Periodic scan	
	Scan END processing	Normal scan program	System periodic processing	Periodic scan
H-252	1.3 ms * approx.	Processing time of Command, Ladder program	3ms max.	Processing time of INTn ~ RTI (n = 0, 1, 2)
H-250	1.3 ms * approx.		3ms max.	

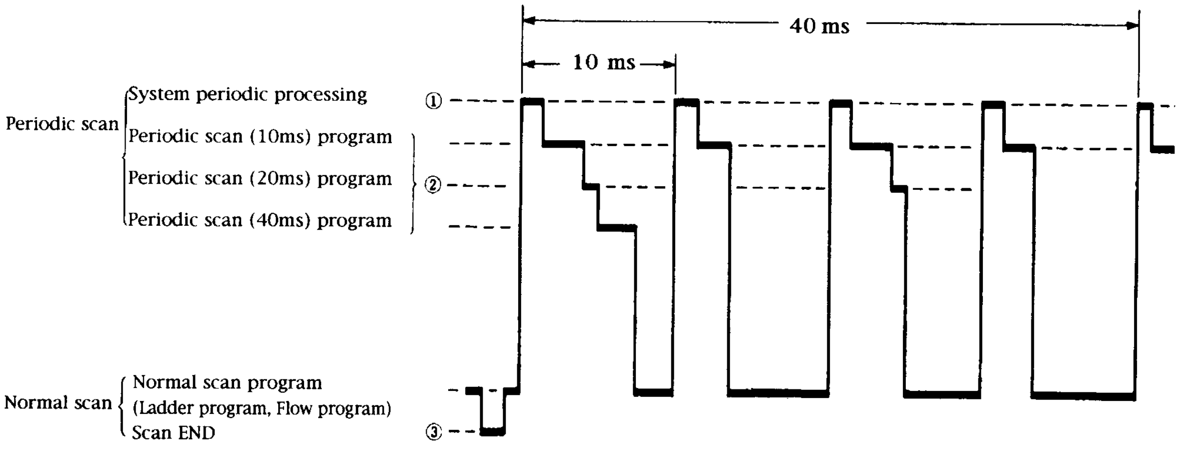
* This is the time in the refresh mode. (In the direct mode, the time of H-252 is 32μs approximately, the time of H-250 is 180μs approximately.)

(a) Monitoring the normal scan time.

To know the maximum time, the in-process time and the minimum time of the normal scan, monitor the special internal outputs WRF010 ~ WRF012.

(8) Calculation scan times

A periodic scan interrupts a normal scan in order to perform the processing. Therefor, when scan time of the periodic scan becomes longer, the normal scan time would be unnecessarily prolonged, resulting in a disable status of the normal scan. When a normal scan is required to be completed in a pre- determined value, the processing should properly be allocated in each scan, in consideration on each load to be processed.



Example of scans

① System periodic processing(processing time per 10ms)

H-252 3ms maximum

H-250 3ms maximum

When the trace monitor function is used in H-252, (Sampling points:50 points,

Trigger requirement:16 points)

2.3ms maximum

longer time would be necessary depending on situations.

② Periodic scan

Execution time of a 10ms periodic scan program (from INT0 to RTI)

Execution time of a 20ms periodic scan program (from INT1 to RTI)

Execution time of a 40ms periodic scan program (from INT2 to RTI)

③ Scan END processing time

H-252 1.3ms

H-250 1.3ms

One scan time (X) is divided into the following four portions.

a) Normal scan time

b) Scan END processing time

c) Total periodic scan time.

" c)" is divided into the following three portions.

c-1) 10ms periodic scan processing time (one scan time /10ms × periodic scan (10ms) processing time).

c-2) 20ms periodic scan processing time (one scan time /20ms × periodic scan (20ms) processing time).

c-3) 40ms periodic scan processing time (one scan time /40ms × periodic scan (40ms) processing time).

Resultant X is expressed below.

$$X(\text{ms}) = (\text{Normal scan time}) + \text{END processing time}$$

$$+ X \frac{\text{System periodic processing time} + \text{Periodic scan (10ms) execution time}}{\text{10ms}}$$

$$+ \frac{\text{Periodic scan (20ms) execution time}}{\text{20ms}} + \frac{\text{Periodic scan (40ms) execution time}}{\text{40ms}}$$

The above equation is transformed;

$$X = \frac{\text{Normal scan time} + \text{END processing time}}{1 - \frac{\left(\begin{array}{c}\text{System periodic processing time}\\ + \text{Periodic scan (10ms) execution time}\end{array}\right)}{\text{10ms}} + \frac{\left(\begin{array}{l}\text{Periodic scan (20ms)}\\ \text{execution time}\end{array}\right)}{\text{20ms}} + \frac{\left(\begin{array}{l}\text{Periodic scan (40ms)}\\ \text{execution time}\end{array}\right)}{\text{40ms}}}$$

Each scan time can be calculated from the Tables of Commands attached

Chapter 3 Ladder / Command

This chapter explains the function of each command.

3.1 Classification of commands

(1) The following table shows the classification of the commands used in the H - series PC H-250/252.

Table 3.1 Command table

No.	Command classification	Description	Types
1	Basic command	Sequence	21
		Timer/Counter	10
		Relational box	8
2	Arithmetic command	Arithmetic expression (array variable)	1
		Arithmetic operation	10
		Logical operation	3
		Relational operation	8
3	Application command	Bit operation	3
		Shift / Rotate	8
		Transfer	7
		Negation / Complement of "2" / Sign	5
		Conversion	5
		Application (Square root, Bit count, Conversion, FIFO, Unite, Depart, *I/O conversion)	9
4	Control command	END, JMP, CAL, FOR, NEXT, RTS, RTI, LBL, SB, INT, CEND, CJMP	12
5	Transfer command for sophisticated function module	*TRNS1, *TRNS4	2
6	F U N command	*Trigonometric function, *Data function, *Data research, *Square root, Others *Comment, *Trace monitor	38
7	P I D function	*PID operation	3

⋆ Effective for H-252.

Caution

The internal output area of WR is discontinuous in WR0~3FF and WR400~43FF in H-252. Therefore, please note the following points.

(1) Double word I/O No."DR03FF" which spreads over these two areas can not be executed correctly and please pay attention not to use "DR03FF" in the programming.

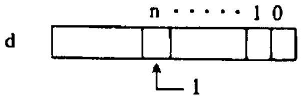
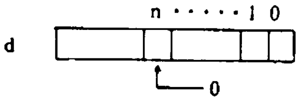
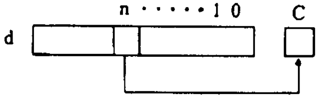
(2) By the instruction when two areas extend and are used
- The processing is executed to WR3FF and makes DER=1 (WSHR, MOV, and DECO, etc.).
- The processing is disabled and DER=1 (FUN and TRNS, etc.).

Please refer to precaution of each instruction for details.

3.2 List of Commands

This section explains how to refer to the tables showing details of each command.

Table 3.2 Format of Command list tables

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C	Processing speed (μ s)		Steps	Remarks
														H-252	H-250		
Bit Operation	1		BSET (d, n)	Bit Set	 d: Word n: 0 ~ 15 d: Double word n: 0 ~ 31	Word D Word	d n d n	[Word] d: WY, WR, WL, WM, TC n: WX, WY, WR, WL,WM, TC, Constants [Double word] d: DY, DR, DL, DM n: WX, WY, WR, WL,WM, TC, Constants	●	●	●	●	●			3 3	
	2		BRES (d, n)	Bit Reset	 d: Word n: 0 ~ 15 d: Double word n: 0 ~ 31	Word D Word	d n d n		●	●	●	●	●			3 3	
	3		BTS (d, n)	Bit Test	 d: Word n: 0 ~ 15 d: Double word n: 0 ~ 31	Word D Word	d n d n		●	●	●	●	↕			3 3	

[1][2] [3] [4] [5] [6] [7] [8] [9] [10] [11]

[1] Category..................................Category of Command by processing purposes.

[2] No. ....Serial No. of Command in each category.

[3] Ladder Symbol ....Expression format to be used in writing Ladder diagram Program.

Command symbol...................Expression format to be used in writing Command Program.

(a) Ladder symbol and Command symbol are different in each Basic command, while they are the same in each command of other types (Arithmetic, Application, Control, Transfer, FUN, PID).

(b) The notations d, S, p and n in the parentheses ( ) are numeric values to be used for arguments or I/O for various types of processing.

d (Destination): Contents of I/O will be changed by the processing.

S (Source) : Contents of I/O will be referred, but will not be changed.

p (Pointer) : Gives the top location of FIFO buffer.

n (number) : Gives the size, the bit location and the label number.

These notations will differ from the above in some types of programming devices

[4] Command name.......................Name of each command.

[5] Description of Processing.......Outline of the processing.

[6] Size...I/O size usable in the command.

[7] Type.......................................I/O type usable in the command.

[8] Condition Code

C................................Carry (Special internal output R7F0)

Indicates the contents of carry by addition, borrow by subtraction, or shift.

V................................Overflow (Special internal output R7F1)

Indicates that an overflow is occurred in signed data operation, exceeding out of the range of the signed data.

Example: The range of signed word data: -32 768 ~ 32 767

WR0001 = -32 768

WR0002 = 1

WR0000 = WR0001 - WR0002

WR0000 is exceeding out of the range -32 768, resulting in an overflow.

For details, refer to the commands Binary Addition and Binary Subtraction.

SD..............................Shift data (Special internal output R7F2)

The contents of SD will be shifted by the command SHR or SHL.

ERR............................Error (Special internal output R7F3)

This will be set to "1" when an error occurs in the execution of a control commands or a special command. The corresponding error code will be set in WRF015. When no error is occurred, the last status will be maintained.

DER............................Data error (Special internal output R7F4)

This will be set to "1" as a data error, when I/O number exceeds the permissible value or BCD is found to be abnormal. When no data error is occurred, this will be set to "0".

↕..................................Changes according to the operation result.

•..................................The last status will be maintained.

①..................................Will be set to "1" when the operation result has an error. Will be maintained in the last status in other cases.

[9] Processing speed (μs)..............Time required to execute the command

Shown as "the minimum value~ the maximum value" when an execution has been normally ended. These values will differ depending on types of commands and number of parameters and data. For details, refer to the detailed description sheet for each command.

[10] Steps......................................Number of steps of the commands.

Number of steps will differ even in the same command depending on their sizes and types.

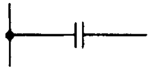
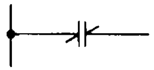
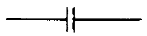
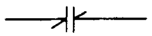
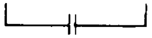
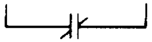
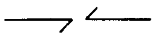
[11] Remarks.................................Remarks for each command if any.

S..................................Signed data.

Signed data will be displayed with the sign in the circuit monitoring, while the sign will not be displayed in I/O monitoring or Overview list monitoring.

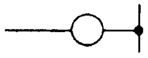
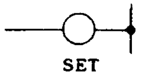
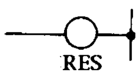
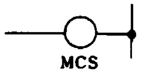
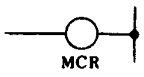
The command lists are shown on the following pages.

(1) Basic commands (1/6)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)		Steps	Remarks
									DER	ERR	SD	V	C	H-252	H-250		
Sequence commands	1		LD	Start logical operation	The start of "contact a" operation of relay function, type X, Y, R, L, M, TD, SS, WDT, MS, TMR, CU, CT, DIF, DFN.	Bit	-	X O U S □□ X R St S □□ Y O U S □□ Y R St S □□ Decimal R0 ~ R7FF L0 ~ L3FFF L10000 ~ L13FFF M0 ~ M3FFF TD0 ~ TD255 SS0 ~ SS255 WDT0 ~ WDT255 MS0 ~ MS255 TMR0 ~ TMR255 CU0 ~ CU511 RCU0 ~ RCU255 CT0 ~ CT511 DIF0 ~ DIF511 DFN0 ~ DFN511	●	●	●	●	●	0.2 (0.5)	0.6 (0.9)	1	U: Unit S: Slot R: Remote host St: Remote local *1
	2		LDI	Start logical NOT operation	The start of "contact b" operation of relay function, type X, Y, R, L, M, TD, SS, WDT, MS, TMR, CU, CT, DIF, DFN.	Bit	-										
	3		AND	Serial AND connection	The "contact a" AND connection of relay function, type X, Y, R, L, M, TD, SS, WDT, MS, TMR, CU, CT, DIF, DFN.	Bit	-										
	4		ANI	Serial AND/NOT connection	The "contact b" AND connection of relay function, type X, Y, R, L, M, TD, SS, WDT, MS, TMR, CU, CT, DIF, DFN.	Bit											
	5		OR	Parallel OR connection	The "contact a" OR connection of relay function, type X, Y, R, L, M, TD, SS, WDT, MS, TMR, CU, CT, DIF, DFN.	Bit	-		●	●	●	●	●	0.4 (0.7)	1.2 (1.5)	2	
	6		ORI	Parallel OR/NOT connection	The "contact b" OR connection of relay function, type X, Y, R, L, M, TD, SS, WDT, MS, TMR, CU, CT, DIF, DFN.	Bit											
	7		NOT	Logical NOT	Inverts the previous operational result.	Bit	-	———	●	●	●	●	●	0.4	1.8	2	
	8		AND DIF	Rising edge detection	Detects the rising edge of a signal	Bit		DIF0 ~ DIF511 Decimal	●	●	●	●	●	0.6 (1.2)	3.0 (3.6)	3	Avoid double numbering. *1
			OR DIF											0.8 (1.4)	3.6 (4.2)	4	
	9		AND DFN	Falling edge detection	Detects the falling edge of a signal	Bit	-	DFN0 ~ DFN511 Decimal	●	●	●	●	●	0.6 (1.2)	3.0 (3.6)	3	Avoid double numbering. *1
			OR DFN											0.8 (1.4)	3.6 (4.2)	4	

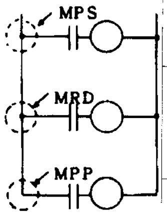
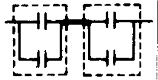
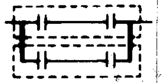
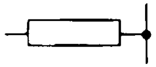
*1. These are the processing speed when the internal outputs are used in the direct mode and the refresh mode. Speeds in ( ) are for the external I/O in the direct mode.

Basic commands (2/6)

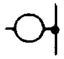
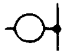
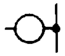
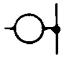
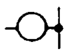
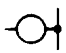
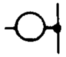
Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C	Processing speed (µ s) H-252	 H-250	Steps	Remarks
Sequence commands	10		OUT	I/O Output	Output coil of a relay function type Y, R, L, M, TD, SS, WDT, MS, TMR, CU, RCU, CTU, CTD, CL.	Bit	-	**Decimal** **Y 0US□□** **Y RStS□□** **R0~R7FF** **L0~L3FFF** **L10000~L13FFF** **M0~M3FFF** **TD0~TD255** **SS0~SS255** **WDT0~WDT255** **MS0~MS255** **TMR0~TMR255** **CU0~CU511** **RCU0~RCU511** **CTU0~CTU511** **CTD0~CTD511** **CL0~CL511**	●	●	●	●	●	0.4 (0.5)	0.8 (0.9)	1	U: Unit S: Slot R: Remote host St: Remote local *1
	11		SET	I/O setting	Set output of a relay function type Y, R, L, M.	Bit	-	**Decimal** **Y 0US□□** **Y RStS□□** **R0~R7FF** **L0~L3FFF** **L10000~L13FFF** **M0~M3FFF**	●	●	●	●	●	0.4 (0.5)	0.8 (0.9)	1	U: Unit S: Slot R: Remote host St: Remote local *1
	12		RES	I/O resetting	Reset output of a relay function type Y, R, L, M.												
	13		MCS	Start Master Control	Setting operation of Master control.	Bit	-	**MCS0~MCS49**	●	●	●	●	●	0.6	3.0	3	Avoid double numbering.
	14		MCR	Reset Master Control	Resetting operation of Master control.	Bit	-	**MCR0~MCR49**	●	●	●	●	●	0.4	1.8	2	Avoid double numbering.

*1. These are the processing speed when the internal outputs are used in the direct mode and the refresh mode. Speeds in (   ) are for the external I/O in the direct mode.

Basic commands (3/6)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size	Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)		Steps	Remarks
								DER	ERR	SD	V	C	H-252	H-250		
Sequence commands	15		MPS	Push Operational Result	Stores the operation result of the last MPS command.	-	-	●	●	●	●	●	-	-	0	
	16		MRD	Read Operatinal Result	Reads the operation result stored by the MPS command and continues the operation in response to a next command.											
	17		MPP	Pull Operational Result	Reads the operation result stored by the MPS command and continues operation in response to a next command. Clears the operation result stored by the MPS command.											
	18		ANB	Logical block Serial Connection	Serial connection of two logical blocks.	-	-	●	●	●	●	●	-	-	0	
	19		ORB	Logical block Parallel Connection	Parallel connection of two logical blocks.								0.2	0.6	1	
	20		[ ]	Start and End, Processing Box	Programs an arithmetic command, an application command, a special command or a control command, between the parentheses [(Start of an operation) and] (End of an operation).	-	-	●	●	●	●	●	0.6	1.8	3	
	21		( )	Start and End, Relational Box	Programs an relational box command, between the parentheses [(Start of an operation) and] (End of an operation).	-	-	●	●	●	●	●	0.4	1.2	0	

Basic commands (4/6)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size	Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)		Steps	Remarks
								DER	ERR	SD	V	C	H-252	H-250		
Timer	1	 TD	OUT TD	On-delay Timer	Operation of On-delay Timer	-	TD0 ~ TD255. Available 0 ~ 63 for 0.01s scale	●	●	●	●	●	1.8	6.3	5	Avoid double numbering
	2	 SS	OUT SS	Single shot	Operation of Single shot	-	SS0 ~ SS255. Available 0 ~ 63 for 0.01s scale	●	●	●	●	●	1.8	6.3	5	
	3	 MS	OUT MS	Mono-stable Timer	Operation of Mono-stable Timer	-	MS0 ~ MS255. Available 0 ~ 63 for 0.01s scale	●	●	●	●	●	1.8	6.3	5	
	4	 TMR	OUT TMR	Integral Timer	Operation of Integral Timer	-	TMR0 ~ TMR255. Available 0 ~ 63 for 0.01s scale	●	●	●	●	●	1.8	6.3	5	
	5	 WDT	OUT WDT	Watch Dog Timer	Operation of Watch Dog Timer	-	WDT0 ~ WDT255. Available 0 ~ 63 for 0.01s scale	●	●	●	●	●	48.9~68.6	78.2~107.6	7	
Counter	6	 CU	OUT CU	Counter	Operation of Counter	-	CU0 ~ CU511	●	●	●	●	●	2.0	6.5	5	
	7	 RCU	OUT RCU	Ring Counter	Operation of Ring Counter	-	RCU0 ~ RCU511	●	●	●	●	●	2.0	6.5	5	
	8	 CTU	OUT CTU	Up/Down Counter Up	Operation of Up/Down Counter Up. Use together with Up/ Down Counter Down in pairs.	-	CTU0 ~ CTU511	●	●	●	●	●	2.0	6.5	5	
	9	 CTD	OUT CTD	Up/Down Counter Down	Operation of Up/Down Counter Down. Use together with Up/ Down Counter Up in pairs.	-	CTD0 ~ CTD511	●	●	●	●	●	1.6	5.2	3	
	10	 CL	OUT CL	Counter Clear	Operation of Counter Clear (CU, RCU, CTU, CTD) and Watch Dog Timer Clear.	-	CL0 ~ CL511	●	●	●	●	●	0.4	0.8	1	

Basic commands (5/6)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C	Processing speed (μ s) H-252	Processing speed (μ s) H-250	Steps	Remarks
Relational Box	1	S_1 == S_2	LD(S_1==S_2)	= Relational Box	ON when S1 = S2. OFF when S1 < > S2.	Word	S1	[Words] WX O U S m WX r St S m WY O U S m WY r St S m *3 WR0~WR43FF WL0~WL3FF WL1000~WL13FF WM0~WM3FF TC0~TC511 Constants 0~65535 H0~HFFFF	●	●	●	●	●	1.0~17.3	4.6~20.8	5	*1 *2 Upper columns for Words. Lower columns for DoubleWords.
		S_1 == S_2	AND (S_1==S_2)				S2									6	
		S_1 == S_2	OR(S_1==S_2)			Double Word	S1							30.2~54.3	50.2~86.2	7	
							S2									8	
	2	S_1 S== S_2	LD(S_1S==S_2)	Signed = Relational Box	ON when S1 = S2. OFF when S1 < > S2. S1 and S2 are compared as signed 32-bit binary numbers.	Double Word	S1		●	●	●	●	●	30.2~54.3	50.2~86.2	5	*2
		S_1 S== S_2	AND (S_1S==S_2)													6	
		S_1 S== S_2	OR(S_1S==S_2)				S2									7	
																8	
	3	S_1 <> S_2	LD (S_1 < > S_2)	<> Relational Box	ON when S1 < > S2. OFF when S1 = S2	Word	S1	[Double Words] DX O U S m DX r St S m DY O U S m DY r St S m *3 DR0~DR43FE DL0~DL3FE DL1000~DL13FE DM0~DM3FE Constants 0~4294967295 H0~HFFFF FFFF	●	●	●	●	●	1.0~17.3	4.6~20.8	5	*1 *2 Upper columns for Words. Lower columns for DoubleWords.
		S_1 <> S_2	AND (S_1 < > S_2)				S2									6	
		S_1 <> S_2	OR (S_1 < > S_2)			Double Word	S1							32.6~49.3	53.8~78.7	7	
							S2									8	
	4	S_1 S<> S_2	LD (S_1 S< > S_2)	Signed < > Relational Box	ON when S1 < > S2. OFF when S1 = S2. S1 and S2 are compared as signed 32-bit binary numbers.	Double Word	S1		●	●	●	●	●	32.6~49.3	53.8~78.7	5	*2
		S_1 S<> S_2	AND (S_1 S< > S_2)													6	
		S_1 S<> S_2	OR (S_1 S< > S_2)				S2									7	
																8	

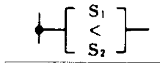
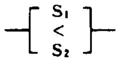
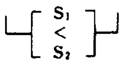
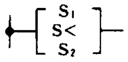
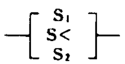
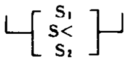
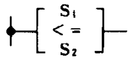
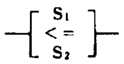
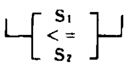
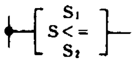
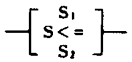
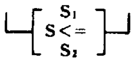
*1) In Word size, LD (S1 ☐ S2) and AND (S1 ☐ S2) have 5 steps, and OR (S1 ☐ S2) has 6 steps.

*2) In Double Word size, LD (S1 ☐ S2) and AND (S1☐ S2) have 5 steps when both S1 and S2 are I/O. They have 6 steps when S1 and S2 are I/O and a real number, a real number and I/O, or when they have 7 steps when both Sl and S2 are constants. In OR (S1 ☐ S2), one step is added to each of the above cases.

Type in the above table: Each number is assigned to; U: Unit S: Slot m: Word inside module R: Remote host St: Remote local

*3) H-250 : WR0 to WR3FF or DR0 to DR3FE.

Basic commands (6/6)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)		Steps	Remarks
									DER	ERR	SD	V	C	H-252	H-250		
Relational Box	5	  	LD(S₁<S₂) AND(S₁<S₂) OR(S₁<S₂)	< Relational Box	ON when S1 < S2. OFF when S1 > = S2.	Word Word Double Word Double Word	S1 S2 S1 S2	[Words] S₁, S₂ WX $\boxed{O}\boxed{U}\boxed{S}\boxed{m}$ WX $\boxed{r}\boxed{St}\boxed{S}\boxed{m}$ WY $\boxed{O}\boxed{U}\boxed{S}\boxed{m}$ WY $\boxed{r}\boxed{St}\boxed{S}\boxed{m}$ WR0~WR43FF *3 WL0~WL3FF WL1000~WL13FF WM0~WM3FF TC0~TC511 Constants 0~65535 H0~HFFFF	●	●	●	●	●	1.0~17.3 37.1~48.2	4.6~20.8 60.5~77.1	5 6 7 8	*1 *2 Upper columns for Words. Lower columns for DoubleWords.
	6	  	LD (S₁ S< S₂) AND (S₁ S< S₂) OR (S₁ S< S₂)	Signed < Relational Box	ON when S1 < S2. OFF when S1 > = S2. S1 and S2 are compared as signed 32-bit binary numbers.	Double Word Double Word	S1 S2		●	●	●	●	●	29.6~56.2	49.4~89.2	5 6 7 8	*2
	7	  	LD (S₁ <= S₂) AND (S₁ <= S₂) OR (S₁ <= S₂)	< = Relational Box	ON when S1 < = S2. OFF when S1 > S2	Word Word Double Word Double Word	S1 S2 S1 S2	[Double Words] S₁, S₂ DX $\boxed{O}\boxed{U}\boxed{S}\boxed{m}$ DX $\boxed{r}\boxed{St}\boxed{S}\boxed{m}$ DY $\boxed{O}\boxed{U}\boxed{S}\boxed{m}$ DY $\boxed{r}\boxed{St}\boxed{S}\boxed{m}$ DR0~DR43FE *3 DL0~DL3FE DL1000~DL13FE DM0~DM3FE Constants 0~4294967295 H0~HFFFF FFFF	●	●	●	●	●	1.0~17.3 38.4~49.2	4.6~20.8 62.4~78.6	5 6 7 8	*1 *2 Upper columns for Words. Lower columns for DoubleWords.
	8	  	LD (S₁ S<= S₂) AND (S₁ S<= S₂) OR (S₁ S<= S₂)	Signed < = Relational Box	ON when S1 < = S2. OFF when S1 > S2. S1 and S2 are compared as signed 32-bit binary numbers.	Double Word Double Word	S1 S2		●	●	●	●	●	30.6~57.6	50.9~91.1	5 6 7 8	*2

*1) In Word size, LD (S1 ☐ S2) and AND (S1 ☐ S2) have 5 steps, and OR (S1 ☐ S2) has 6 steps.

*2) In Double Word size, LD (S1 ☐ S2) and AND (S1☐ S2) have 5 steps when both S1 and S2 are I/O. They have 6 steps when S1 and S2 are I/O and a real number, a real number and I/O, or when they have 7 steps when both Sl and S2 are constants. In OR (S1 ☐ S2), one step is added to each of the above cases.

Type in the above table: Each number is assigned to; U: Unit S: Slot m: Word inside module R: Remote host St: Remote local

*3) H-250 : WR0 to WR3FF or DR0 to DR3FE.

(2) Arithmetic commands (1/3)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)		Steps	Remarks
									DER	ERR	SD	V	C	H-252	H-250		
Assignment	1	d = S		Assignment Statement	Assignment S to d Array variables can be used for d and S. External I/O can not be used for I/O of array variables.	Bit	d	Y, R, L, M	↕	●	●	●	●	0.8 ~ 201.4	3.8 ~ 306.3	3 4	I/O : I/O I/O : Array
							S	X, Y, R, L, M, Constants								4 5	Array : I/O Array : Array
						Word	d	WY, WR, WL,WM, TC	↕	●	●	●	●	0.9 ~ 233.0	3.8 ~ 353.5	3 4	I/O : I/O I/O : Array
							S	WX, WY, WR, WL, WM, TC, Constants								4 5	Array : I/O Array : Array
						Double Word	d	DY, DR, DL, DM	↕	●	●	●	●	1.6 ~ 242.3	5.0 ~ 367.4	4 4	I/O : I/O I/O : Array
							S	DX, DY, DR, DL, DM, Constants								5 5	Array : I/O Array : I/O
Arithmetic Operation	1	d = S1 + S2		Binary Addition	S1 + S2 to d	Word	d S1 S2	[Word] d: WY, WR, WL, WM S1, S2: WX, WY, WR, WL, WM, TC, Constants [Double Word] d: DY, DR, DL, DM S1, S2: DX, DY, DR, DL, DM, Constants	●	●	●	↕	↕	42.4 ~ 42.8	68.5 ~ 69.1	4	
						D Word	d S1 S2							52.8 ~53.2	83.9 ~ 84.6	6	
	2	d = S1 B + S2		BCD Addition	S1 + S2 to d	Word	d S1 S2		↕	●	●	●	↕	53.3	84.8	4	
						D Word	d S1 S2							84.5	131.4	6	
	3	d = S1 - S2		Binary Subtraction	S1 - S2 to d	Word	d S1 S2		●	●	●	↕	↕	40.4	65.5	4	
						D Word	d S1 S2							49.7	79.4	6	
	4	d = S1 B - S2		BCD Subtraction	S1 - S2 to d	Word	d S1 S2		↕	●	●	●	↕	53.7	85.4	4	
						D Word	d S1 S2							81.3	126.7	6	
	5	d = S1 × S2		Binary Multiplication	S1 × S2 to d	Word	d S1 S2		↕	●	●	●	●	70.2 ~ 99.0	110.0 ~ 153.1	4	
						D Word	d S1 S2							162.2~1086.5	247.6~1629.8	6	
	6	d = S1 B × S2		BCD Multiplication	S1 × S2 to d	Word	d S1 S2		↕	●	●	●	●	209.4 ~ 344.7	318.2 ~ 520.5	4	
						D Word	d S1 S2							434.7~1889.0	655.1~2829.8	6	

Input Constants for BCD operations in hexadecimal numbers.

Arithmetic commands (2/3)

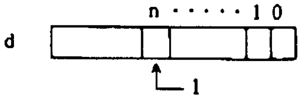
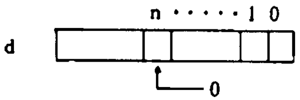
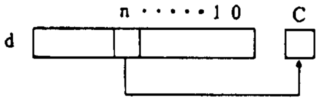
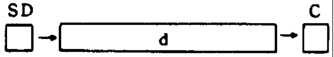
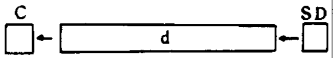
Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)		Steps	Remarks
									DER	ERR	SD	V	C	H-252	H-250		
Arithmetic Operation	7		d = S1 S * S2	Signed Binary Multiplication	S1 * S2 to d	D Word	d S1 S2	[Word] d: WY, WR, WL, WM S1, S2: WX, WY, WR, WL, WM, TC, Constants [Double Word] d: DY, DR, DL, DM S1, S2: DX, DY, DR, DL, DM, Constants	↕	●	●	●	●	172.4~1128.8	262.9~1693.1	6	
	8		d = S1 / S2	Binary Division	(1)Word size S1 / S2 to d S1 mod S2 to WRF016 (2)Double Word size S1 / S2 to d S1 mod S2 to DRF016	Word	d S1 S2		↕	●	●	●	●	104.1	160.7	4	
						D Word	d S1 S2							1494.1~1673.6	2239.3~2527.2	6	
	9		d = S1 B / S2	BCD Division		Word	d S1 S2							222.5~286.1	337.7~432.9	4	
						D Word	d S1 S2							1417.7~2271.3	2125.0~3401.6	6	
	10		d = S1 S / S2	Signed Binary Division		D Word	d S1 S2		↕	●	●	↕	●	1142.0~1731.5	1712.8~2594.3	6	
Logical Operation	1		d = S1 OR S2	OR	S1 + S2 to d	Bit	d S1 S2	[Bit] d: Y, R, L, M S1, S2: X, Y, R, L, M [Word] d: WY, WR, WL, WM, TC S1, S2: WX, WY, WR, WL, WM, TC, Constants [Double Word] d: DY, DR, DL, DM S1, S2: DX, DY, DR, DM, Constants	●	●	●	●	●	32.2~33.7	53.1~55.4	4	
						Word	d S1 S2							25.9	43.8	4	
						D Word	d S1 S2							36.7	60.0	6	
	2		d = S1 AND S2	AND	S1 • S2 to d	Bit	d S1 S2		●	●	●	●	●	32.2~33.7	53.1~55.4	4	
						Word	d S1 S2							25.9	43.8	4	
						D Word	d S1 S2							36.7	60.0	6	
	3		d = S1 XOR S2	Exclusive OR	S1 ⊕ S2 to d	Bit	d S1 S2		●	●	●	●	●	31.0~32.5	51.3~53.6	4	
						Word	d S1 S2							25.9	43.8	4	
						D Word	d S1 S2							36.7	60.0	6	

Input Constants for BCD operations in hexadecimal numbers.

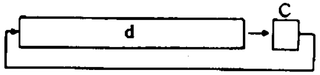
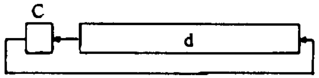
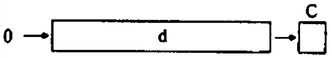
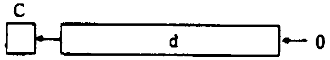
Arithmetic commands (3/3)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C	Processing speed (μ s) H-252	 H-250	Steps	Remarks
Relational Operation	1		d = S1 = = S2	= Relational Expression	If S1 = S2 then 1 to d else 0 to d	Word	d S1 S2	[Word] d: Y, R, L, M S1, S2: WX, WY, WR, WL,WM, TC, Constants [Double Word] d: Y, R, L, M S1, S2: DX, DY, DR, DL, DM, Constants	●	●	●	●	●	1.2~26.1	4.6~28.9	4	
						D Word	d S1 S2							33.5~40.8	55.1~66.0	6	
	2		d = S1 S = = S2	Signed = Relational Expression	If S1 = S2 then 1 to d else 0 to d S1 and S2 are compared as signed numbers.	D Word	d S1 S2							33.5~40.8	55.1~66.0	6	
	3		d = S1 < > S2	< > Relational Expression	If S1 < > S2 then 1 to d else 0 to d	Word	d S1 S2		●	●	●	●	●	1.2~26.1	4.6~28.9	4	
						D Word	d S1 S2							34.1~40.8	56.1~66.0	6	
	4		d = S1 S < > S2	Signed < > Relational Expression	If S1 < > S2 then 1 to d else 0 to d S1 and S2 are compared as signed numbers.	D Word	d S1 S2							34.1~40.8	56.1~66.0	6	
	5		d = S1 < S2	< Relational Expression	If S1 < S2 then 1 to d else 0 to d	Word	d S1 S2		●	●	●	●	●	1.2~26.1	4.6~28.9	4	
						D Word	d S1 S2							38.2~40.1	62.1~65.0	6	
	6		d = S1 S < S2	Signed < Relational Expression	If S1 < S2 then 1 to d else 0 to d S1 and S2 are comparea as signed numbers.	D Word	d S1 S2							34.3~46.2	56.4~74.2	6	
	7		d = S1 < = S2	< = Relational Expression	If S1 < = S2 then 1 to d else 0 to d	Word	d S1 S2		●	●	●	●	●	1.2~26.1	4.6~28.9	4	
						D Word	d S1 S2							37.9~39.5	61.8~64.1	6	
	8		d = S1 S < = S2	Signed < = Relational Expression	If S1 < = S2 then 1 to d else 0 to d S1 and S2 are compared as signed numbers.	D Word	d S1 S2							42.8~43.7	69.1~70.4	6	

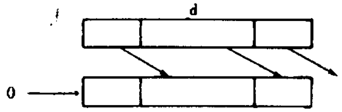
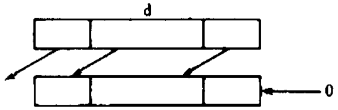
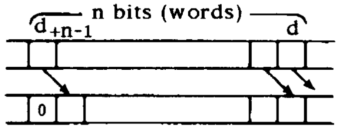
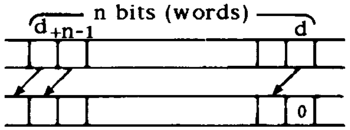
(3) Application commands (1/8)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C	Processing speed (μ s) H-252	 H-250	Steps	Remarks
Bit Operation	1	BSET (d, n)		Bit Set	 d: Word n: 0 ~ 15 d: Double word n: 0 ~ 31	Word	d n	[Word] d: WY, WR, WL, WM, TC n: WX, WY, WR, WL,WM, TC, Constants [Double word] d: DY, DR, DL, DM n: WX, WY, WR, WL,WM, TC, Constants	●	●	●	●	●	31.5~83.8	52.2~130.4	3	
						D Word	d n							32.7~86.1	54.0~133.8	3	
	2	BRES (d, n)		Bit Reset	 d: Word n: 0 ~ 15 d: Double word n: 0 ~ 31	Word	d n		●	●	●	●	●	34.3~86.7	56.4~134.6	3	
						D Word	d n							35.6~89.3	58.4~138.6	3	
	3	BTS (d, n)		Bit Test	 d: Word n: 0 ~ 15 d: Double word n: 0 ~ 31	Word	d n		●	●	●	●	↕	32.6~86.9	53.8~135.0	3	
						D Word	d n							35.4~88.9	58.0~138.1	3	
Shift / Rotate	1	SHR (d, n)		Shift Right	 Shifting right by n bits. d: Word n: 0 ~ 15 d: Double word n: 0 ~ 31	Word	d n	[Word] d: WY, WR, WL, WM, TC n: WX, WY, WR, WL,WM, TC, Constants [Double word] d: DY, DR, DL, DM n: WX, WY, WR, WL,WM, TC, Constants	●	●	●	●	↕	41.4~93.1	67.0~144.3	3	
						D Word	d n							53.3~209.9	84.3~319.0	3	
	2	SHL (d, n)		Shift Left	 Shifting left by n bits. d: Word n: 0 ~ 15 d: Double word n: 0 ~ 31	Word	d n		●	●	●	●	↕	41.2~93.1	66.7~144.3	3	
						D Word	d n							54.3~211.2	86.2~320.9	3	

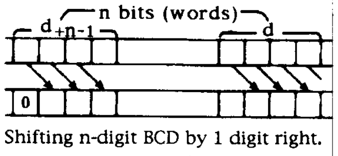
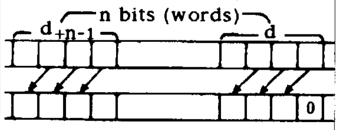
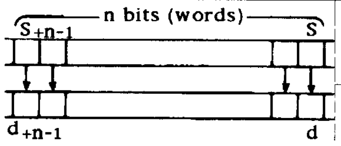
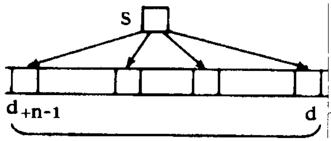
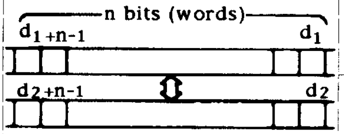
Application commands (2/8)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)		Steps	Remarks
									DER	ERR	SD	V	C	H-252	H-250		
Shift / Rotate	3		ROR (d, n)	Rotate Right	 Rotating right by n bits. d: Word n: 0 ~ 15 d: Double word n: 0 ~ 31	Word	d n	[Word] d: WY, WR, WL, WM, TC n: WX, WY, WR, WL, WM, TC, Constants [Double word] d: DY, DR, DL, DM n: WX, WY, WR, WL, WM, TC, Constants	●	●	●	●	↕	41.2~76.3	66.7~119.2	3	
						D Word	d n							54.0~174.9	85.7~266.7	3	
	4		ROL (d, n)	Rotate Left	 Rotating left by n bits. d: Word n: 0 ~ 15 d: Double word n: 0 ~ 31	Word	d n		●	●	●	●	↕	41.2~76.3	66.7~119.2	3	
						D Word	d n							54.0~174.9	85.7~266.7	3	
	5		LSR (d, n)	Logical shift Right	 Shifting right by n bits. d: Word n: 0 ~ 15 d: Double word n: 0 ~ 31	Word	d n		●	●	●	●	↕	35.0~70.1	57.4~109.9	3	
						D Word	d n							47.7~168.7	76.4~257.4	3	
	6		LSL (d, n)	Logical shift Left	 Shifting left by n bits. d: Word n: 0 ~ 15 d: Double word n: 0 ~ 31	Word	d n		●	●	●	●	↕	35.0~70.1	57.4~109.9	3	
						D Word	d n							47.7~168.7	76.4~257.4	3	

Application commands (3/8)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size	Type						Processing speed (μ s)			Remarks
													H-252	H-250		
Shift / Rotate	7	BSR (d, n)		BCD Shift Right	 Shifting BCD right by n digits. d: Word n: 0 ~ 3 d: Double word n: 0 ~ 7	d n	[Word] d: WY, WR, WL, WM, TC n: WX, WY, WR, WL,WM, TC, Constants	●	●	●	●	●	34.3~49.2	56.4~78.6	3	
						d n							51.7~132.8	82.3~203.6	3	
	8	BSL (d, n)		BCD Shift Left	 Shifting BCD left by n digits. d: Word n: 0 ~ 3 d: Double word n: 0 ~ 7	d n	[Double word] d: DY, DR, DL, DM n: WX, WY, WR, WL,WM, TC, Constants	●	●	●	●	●	34.3~49.2	56.4~78.6	3	
						d n							51.7~132.8	82.3~203.6	3	
Transfer	1	WSHR (d, n)		Shift Right Block	 Shifting a n-bit (Word) block right by one bit (Word). n: 0 ~ 255	d n	d: (Note) R, L, M n: WX, WY, WR, WL,WM, TC, Constants	↕	●	●	●	●	75.1	117.4	3	
						d n							73.5	114.9	3	
	2	WSHL (d, n)		Shift Left Block	 Shifting a n-bit (Word) block left by one bit (Word). n: 0 ~ 255	d n	[Word] d: WR, WL, WM n: WX, WY, WR, WL,WM, TC, Constants	↕	●	●	●	●	71.0	111.2	3	
						d n							81.3	126.7	3	

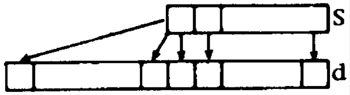
Application commands (4/8)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)		Steps	Remarks
									DER	ERR	SD	V	C	H-252	H-250		
Transfer	3		WBSR (d, n)	(BCD SHIFT RIGHT BLOCK)	 Shifting n-digit BCD by 1 digit right. n: 0 ~ 255	Word	d n	[Bit] d: R, L, M n: WX, WY, WR, WL,WM, TC, Constants	↕	●	●	●	●	80.0	124.7	3	*1
	4		WBSL (d, n)	(BCD SHIFT LEFTBLOCK)	 Shifting n-digit BCD by 1 digit left. n: 0 ~ 255	Word	d n	[Word] d: WR, WL, WM n: WX, WY, WR, WL,WM, TC, Constants	↕	●	●	●	●	81.3	126.7	3	
	5		MOV (d, S, n)	(MOVE)	 n : 0 ~ 255	Bit	d S	R, L, M	↕	●	●	●	●	134.9	206.8	4	*1
						Bit	n	WX, WY, WR, WL, WM, TC, Constants									
						Word	d S	WR, WL, WM						98.8	152.7	4	
						Word	n	WX, WY, WR, WL, WM, TC, Constants									
	6		COPY (d, S, n)	(COPY)	 n : 0 ~ 255	Bit	d	R, M, L	↕	●	●	●	●	82.4	128.3	4	
						Bit	S	X, Y, R, L, M, Constants									
						Bit	n	WX, WY, WR, WL, WM, TC, Constants									
						Word	d	WR, WL, WM						71.1	111.3	4	
						Word	S n	WX, WY, WR, WL, WM, TC, Constants									
	7		XCG (d1, d2, n)	(EXCHANGE)	 n : 0 ~ 255	Bit	d1 d2	R, L, M	↕	●	●	●	●	165.0	251.8	4	*1
						Bit	n	WX, WY, WR, WL, WM, TC, Constants									
						Word	d1 d2	WR, WL, WM						160.3	244.8	4	
						Word	n	WX, WY, WR, WL, WM, TC, Constants									

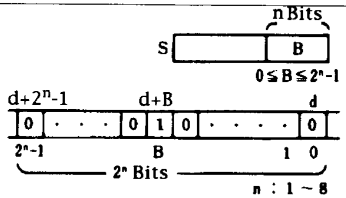
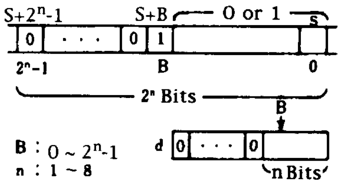
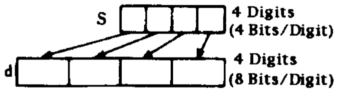
---

* 1) Processing speeds are for n = 1.

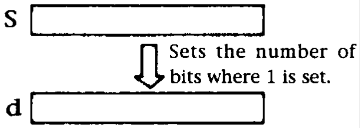
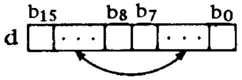
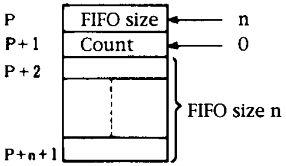
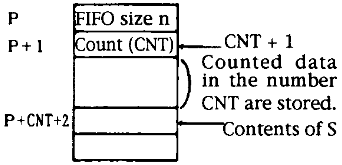
Application commands (5/8)

<table>
<tr><th rowspan="2">Category</th><th rowspan="2">No.</th><th rowspan="2">Ladder symbol</th><th rowspan="2">Command symbol</th><th rowspan="2">Command name</th><th rowspan="2">Description of processing</th><th rowspan="2" colspan="2">Size</th><th rowspan="2">Type</th><th>R7F4</th><th>R7F3</th><th>R7F2</th><th>R7F1</th><th>R7F0</th><th colspan="2">Processing speed (μ s)</th><th rowspan="2">Steps</th><th rowspan="2">Remarks</th></tr>
<tr><th>DER</th><th>ERR</th><th>SD</th><th>V</th><th>C</th><th>H-252</th><th>H-250</th></tr>
<tr><td rowspan="14">Sign / Two's Complement / Negation</td><td rowspan="3">1</td><td rowspan="3" colspan="2">NOT (d)</td><td rowspan="3">Inversion
(NOT)</td><td rowspan="3">$\overline{d}$ to d</td><td>Bit</td><td>d</td><td>Y, R, L, M</td><td rowspan="3">●</td><td rowspan="3">●</td><td rowspan="3">●</td><td rowspan="3">●</td><td rowspan="3">●</td><td>29.1~30.6</td><td>48.6~50.9</td><td>2</td><td rowspan="3"></td></tr>
<tr><td>Word</td><td>d</td><td>WY, WR, WL, WM</td><td>25.5</td><td>43.2</td><td>2</td></tr>
<tr><td>D Word</td><td>d</td><td>DY, DR, DL, DM</td><td>35.9</td><td>58.7</td><td>2</td></tr>
<tr><td rowspan="2">2</td><td rowspan="2" colspan="2">NEG (d)</td><td rowspan="2">Two's Complement
(NEGATE)</td><td rowspan="2">-d to d</td><td>Word</td><td>d</td><td>WY, WR, WL, WM</td><td rowspan="2">●</td><td rowspan="2">●</td><td rowspan="2">●</td><td rowspan="2">●</td><td rowspan="2">●</td><td>26.3</td><td>44.3</td><td>2</td><td rowspan="2"></td></tr>
<tr><td>D Word</td><td>d</td><td>DY, DR, DL, DM</td><td>36.6</td><td>59.8</td><td>2</td></tr>
<tr><td rowspan="4">3</td><td rowspan="4" colspan="2">ABS (d, S)</td><td rowspan="4">Absolute Value
(ABSOLUTE)</td><td rowspan="4">|S| to d
Sign of S to c: 1: Positive
0: Negative</td><td rowspan="2">Word</td><td>d</td><td rowspan="4">[Word]
d: WY, WR, WL, WM
S: WX, WY, WR, WL, WM, TC, Constants.</td><td rowspan="4">●</td><td rowspan="4">●</td><td rowspan="4">●</td><td rowspan="4">●</td><td rowspan="4">↕</td><td rowspan="2">29.6~32.3</td><td rowspan="2">49.4~53.3</td><td rowspan="2">3</td><td rowspan="4"></td></tr>
<tr><td>S</td></tr>
<tr><td rowspan="2">D Word</td><td>d</td><td rowspan="2">36.6~41.5</td><td rowspan="2">59.8~67.2</td><td rowspan="2">4</td></tr>
<tr><td>S</td></tr>
<tr><td rowspan="4">4</td><td rowspan="4" colspan="2">SGET (d, S)</td><td rowspan="4">Sign
(Addition)</td><td rowspan="4">If C = 1 then -S to d
else S to d</td><td rowspan="2">Word</td><td>d</td><td rowspan="4">[Double word]
d: DY, DR, DL, DM
S: DX, DY, DR, DL, DM, Constants</td><td rowspan="4">●</td><td rowspan="4">●</td><td rowspan="4">●</td><td rowspan="4">●</td><td rowspan="4">●</td><td rowspan="2">27.6~30.2</td><td rowspan="2">46.3~50.2</td><td rowspan="2">3</td><td rowspan="4"></td></tr>
<tr><td>S</td></tr>
<tr><td rowspan="2">D Word</td><td>d</td><td rowspan="2">32.8~37.8</td><td rowspan="2">54.1~61.6</td><td rowspan="2">4</td></tr>
<tr><td>S</td></tr>
<tr><td rowspan="2">5</td><td rowspan="2" colspan="2">EXT (d, S)</td><td rowspan="2">Sign Extension
(EXTEND)</td><td rowspan="2"> </td><td rowspan="2">Double Word</td><td>d</td><td>DY, DR, DL, DM</td><td rowspan="2">●</td><td rowspan="2">●</td><td rowspan="2">●</td><td rowspan="2">●</td><td rowspan="2">●</td><td rowspan="2">27.5~27.9</td><td rowspan="2">46.1~46.8</td><td rowspan="2">3</td><td rowspan="2"></td></tr>
<tr><td>S</td><td>WX, WY, WR, WL, WM, TC, Constants.</td></tr>
</table>

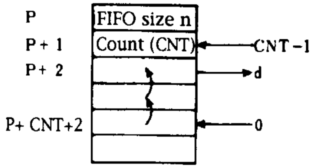
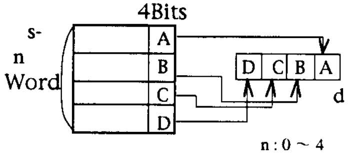
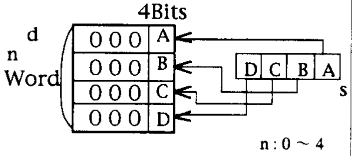
Application commands (6/8)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)		Steps	Remarks
									DER	ERR	SD	V	C	H-252	H-250		
Conversion	1		BCD (d, S)	Binary to BCD Conversion (BCD)	Converts the contents of S into BCD and sets it in d. Word: If S > = H2710 then DER = 1 else DER = 0. D. word: If S > = H5F5 E100 then DER = 1 else DER = 0.	Word	d S	[Word] d: WY, WR, WL, WM S: WX, WY, WR, WL, WM, TC, Constants.	↕	●	●	●	●	60.3~64.5	95.2~101.5	3	
						D Word	d S							109.8~142.0	169.2~217.4	4	
	2		BIN (d, S)	BCD to Binary Conversion (BINARY)	Converts the contents of S into Binary and sets it in d. When BCD is abnormal, DER = 1 When BCD is normal, DER = 0	Word	d S	[Double word] d: DY, DR, DL, DM S: DX, DY, DR, DL, DM, Constants	↕	●	●	●	●	65.7~78.6	103.3~122.6	3	
						D Word	d S							97.0~157.0	150.1~239.8	4	
	3		DECO (d, S, n)	(DECODE)		Bit	d	R, L, M	↕	●	●	●	●	108.7	167.6	4	n = 1
							S	WX, WY, WR, WL, WM, TC, Constants.									
							n	Constants: 1 ~ 8									
	4		ENCO (d, S, n)	(ENCODE)		Bit	d	WY, WR, WL, WM	↕	●	●	●	↕	101.2	156.3	4	n = 1
							S	R, L, M									
							n	Constants: 1 ~ 8									
	5		SEG (d, S)	7-segment Decode (SEGMENT)		Word	d	DY, DR, DL, DM	●	●	●	●	●	73.4	114.8	3	
							S	WX, WY, WR, WL, WM, TC, Constants.									

Application commands (7/8)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)		Steps	Remarks
									DER	ERR	SD	V	C	H-252	H-250		
Application	1	SQR (d, S)		(SQUARE ROOT)	$d \leftarrow \sqrt{S}$ **d** : 4-Digit **BCD** **S** : 8-Digit **BCD**	Word	d	WY, WR, WL, WM	↕	●	●	●	●	1270.9~1572.0	1905.6~2355.8	4	
							S	DX, DY, DR, DL, DM, Constants									
	2	BCU (d, S)		(BIT COUNT)	 Sets the number of bits where 1 is set.	Word	d	WY, WR, WL, WM	●	●	●	●	●	77.1~80.6	120.3~125.5	3	
							S	WX, WY, WR, WL, WM, TC, Constants									
						D Word	d	WY, WR, WL, WM						133.2~140.2	204.2~214.7	4	
							S	DX, DY, DR, DL, DM, Constants									
	3	SWAP (d)		(SWAP)		Word	d	WY, WR, WL, WM	●	●	●	●	●	24.7	42.1	2	
	4	FIFIT (P, n)		(FIFO INITIALIZE)		Word	P	WR, WL, WM	↕	●	●	●	●	116.1	178.6	3	
							n	Constants: 0 ~ 255									
	5	FIFWR (P, S)		(FIFO WRITE)		Word	P	WR, WL, WM	↕	●	●	●	●	134.9	206.8	3	
							S	WX, WY, WR, WL, WM, TC, Constants									

Application commands (8/8)

Category	No.	Ladder symbol	Command symbol	Command name	description of processing	Size		Type	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C	Processing speed (μ s) H-252	H-250	Steps	Remarks
Application	6		FIFRD (P, d)	(FIFO READ)		Word	P	WR, WL, WM	↕	●	●	●	●	115.5	177.8	3	＊2
							d	WY, WR, WL, WM, TC									
	7		UNIT (d, S, n)	Connection (UNIT)		Word	d	WY, WR, WL, WM	↕	●	●	●	●	82.7~111.0	128.8~171.0	4	
							S	WR, WL, WM									
							n	Constants: 0 ~ 4									
	8		DIST (d, S, n)	Separation (DISTRIBUTE)			d	WR, WL, WM	↕	●	●	●	●	80.2~107.4	125.0~165.6	4	
						Word	S	WY, WR, WL, WM, TC, Constants									
							n	Constants: 0 ~ 4									
	9		＊1 ADRIO (d, S)	I/O address Conversion	Actual address of I/O to d. I/O address that is specified by S is set to d.	Bit	S	X, Y, R, L, M	●	●	●	●	●	20.7	-	3	
						Word	d	WY, WR, WL, WM									
							S	WX,WY, WR, WL, WM									

＊1) This command is effective for H-252.

＊2) These processing speeds are for the case where the magnitude of FIFO is n=1.

(4) Control commands (1/2)

Category	No.	Ladder symbol	Command symbol	Command name	description of processing	Size		Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)		Steps	Remarks
									DER	ERR	SD	V	C	H-252	H-250		
Control	1		END	Ending Normal Scan (END)	Indicates the end of a normal scan. The control will be returned to the top of the program to continue scanning.	-	-	-	●	●	●	●	●	32	180~188	1	∗ 1
														1300	1332~1340		∗ 2
	2		CEND (S)	Conditional End of Scan (CONDITIO-NAL END)	When S equals 1, the control will be returned to the top of the program to continue scanning. When S equals 0, the next statement will be executed.	Bit	S	X, Y, R, L, M	●	●	●	●	●	16.6	29.8	2	∗ 3
														(Processing speed of END command) + 2.8	(Processing speed of END command) + 5.5		∗ 4
	3		JMP n	Unconditio-nal Jump (JUMP)	The control will be jumped to LBLn, where n is the same number n of JMPn.	Word	n	Constants: 0 ~ 255	●	①	●	●	●	69.2~175.2	103.5~261.9	2	
	4		CJMP n (S)	Conditional Jump (CONDITIO-NAL JUMP)	If S =1 then Jump to LBLn else not Jump.	Word	n	Constants: 0 ~ 255	●	①	●	●	●	15.6	28.4	3	∗ 3
							S	X, Y, R, L, M						78.9~184.9	108.4~266.8		∗ 4
	5		LBL n (S)	Label (LABEL)	Identifies the targets for Jump by the commands JMP, CJMP, having the same code numbers "n".	Word	n	Constants: 0 ~ 255	●	●	●	●	●	0.2	0.6	1	

∗ 1 In the direct mode

∗ 2 In the refreshing mode.

∗ 3 When the condition is not fulfilled.

∗ 4 When the condition is fulfilled.

Control commands (2/2)

Category	No.	Ladder symbol	Command symbol	Command name	description of processing	Size		Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (µ s)		Steps	Remarks
									DER	ERR	SD	V	C	H-252	H-250		
Control	6	FOR n (S)		FOR	When S equals 0, the control will jump to the step that is located next to NEXTn, where n is the same number n of FORn. When S is not equal to 0, the step next to this command will be executed.	Word	n S	Constants: 0 ~ 49 WY, WR, WL, WM	●	①	●	●	●	46.5~111.7	60.0~157.5	3	
	7	NEXT n		NEXT	Subtract 1 from the repetition count S. The control will be jumped to FORn (S), where n is the same number n of NEXTn.	Word	n	Constants: 0 ~ 49	●	①	●	●	●	59.7~89.2	79.7~123.7	2	
	8	CAL n		Call Subroutine (CALL)	Calls the subroutine SBn, where n is the same number n of CALn.	Word	n	Constants: 0 ~ 99	●	①	●	●	●	57.7~72.9	76.6~99.4	2	
	9	SB n		Start Subroutine	Starts the subroutine SBn, where n is the code number.	Word	n	Constants: 0 ~ 99	●	●	●	●	●	0.2	0.6	1	
	10	RTS		RETURN SUBROUTINE	Returns from the subroutine.	-	-	-	●	●	●	●	●	40.2~55.7	50.5~73.7	1	
	11	INT n		Start Interruption Scan (INTERRUPT)	Starts Interruption Scan of the code number n.	Word	n	Constants: 0 ~ 2	●	●	●	●	●	173~229	540~560	1	
	12	RTI		RETURN INTERRUPT	Returns from Interruption Scan.	Word	-	-	●	●	●	●	●	74~162	360~370	1	

(5) Transfer commands for Sophisticated Function Module *1

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C	Processing speed (μs) H-252	Steps	Remarks
Transfer Commands for Sophisticated Function Module	1		TRNS 1 (d, s, t)	Communication Command for SIH	Communication between CPU module and SIH.	Bit	t	R, L, M	↕	●	●	●	●	271.0~378.9	5	During communication
						Word	d	WY						806.5~931.8		At a start of communication
						Word	s	WR, WL, WM								
	2		TRNS 4 (d, s, t)	Communication Command for POSH	Communication between CPU module and POSH.	Bit	t	R, L, M	↕	●	●	●	●	286.9~315.8	5	During communication
						Word	d	WY						690.1~990.8		At a start of communication
						Word	s	WR, WL, WM								

*1) These commands are effective for H-252.

(6) FUN commands *1 (1/4)

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C	Processing speed (μ s) H-252	Steps	Remarks
FUN Commands	1	FUN 10 (S) (SIN (S))		SIN Function	Trigonometric function, SIN operation.	Word	S	WR, WM, WL	↕	●	●	●	●	117.9~208.6	3	
	2	FUN 11 (S) (COS (S))		COS Function	Trigonometric function, COS operation.	Word	S	WR, WM, WL	↕	●	●	●	●	121.0~207.9	3	
	3	FUN 12 (S) (TAN (S))		TAN Function	Trigonometric function, TAN operation.	Word	S	WR, WM, WL	↕	●	●	●	●	117.8~206.7	3	
	4	FUN 13 (S) (ASIN (S))		ARC SIN Function	Trigonometric function, ARC SIN operation.	Word	S	WR, WM, WL	↕	●	●	●	●	472.7~487.8	3	
	5	FUN 14 (S) (ACOS (S))		ARC COS Function	Trigonometric function, ARC COS operation.	Word	S	WR, WM, WL	↕	●	●	●	●	474.3~486.0	3	
	6	FUN 15 (S) (ATAN (S))		ARC TAN Function	Trigonometric function, ARC TAN operation.	Word	S	WR, WM, WL	↕	●	●	●	●	402.5~494.9	3	
	7	FUN 20 (S) (DSRCH (S))		Data search	Searches the data string to pick up the specified data, and sets the location and the number of these data.	Word	S	WR, WM, WL	↕	●	●	●	●	310.0	3	2 words out of 10 match each other
	8	FUN 21 (S) (TSRCH (S))		Table search	Picks up the specified data tables out of the data string, and stores these tables in the specified location.	Word	S	WR, WM, WL	↕	●	●	●	●	330.3	3	10th block with 2 words/block
	9	FUN 30 (S) (BINDA (S))		Binary to ASCII Conversion (16-bit)	Converts unsigned 16-bit BIN data into decimal ASCII code, and stores them.	Word	S	WR, WM, WL	↕	●	●	●	●	199.1~291.1	3	
	10	FUN 31 (S) (DBINDA (S))		Binary to ASCII Conversion (32-bit)	Converts signed 32-bit BIN data into decimal ASCII code, and stores them.	Word	S	WR, WM, WL	↕	●	●	●	●	281.1~425.1	3	
	11	FUN 32 (S) (BINHA (S))		Binary to ASCII Conversion (16-bit)	Converts unsigned 16-bit BIN data into hexadecimal ASCII code, and stores them.	Word	S	WR, WM, WL	↕	●	●	●	●	125.5~203.9	3	
	12	FUN 33 (S) (DBINHA (S))		Binary to ASCII Conversion (32-bit)	Converts unsigned 32-bit BIN data into hexadecimal ASCII code, and stores them.	Word	S	WR, WM, WL	↕	●	●	●	●	164.2~246.1	3	

---

*1) These commands are effective for H-252.

FUN commands (2/4) *1

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4 DER	R7F3 ERR	R7F2 SD	R7F1 V	R7F0 C	Processing speed (μ s) H-252	Steps	Remarks
FUN Commands	13	FUN 34 (S) (BCDDA (S))		BCD to ASCII Conversion (16-bit)	Converts 16-bit BCD (BCD 4-digit) data into decimal ASCII code, and stores the result.	Word	S	WR, WM, WL	↕	●	●	●	●	130.8~208.8	3	
	14	FUN 35 (S) (DBCDDA (S))		BCD to ASCII Conversion (32-bit)	Converts 32-bit BCD (BCD 8-digit) data into decimal ASCII code, and stores the result.	Word	S	WR, WM, WL	↕	●	●	●	●	175.3~286.2	3	
	15	FUN 36 (S) (DABIN (S))		ASCII to Binary Conversion (16-bit)	Converts 5-digit unsigned decimal ASCII data into 16-bit BIN data, and stores the result.	Word	S	WR, WM, WL	↕	●	●	●	●	179.8~286.1	3	
	16	FUN 37 (S) (DDABIN (S))		ASCII to Binary Conversion (32-bit)	Converts 10-digit signed decimal ASCII data into 32-bit BIN data, and stores the result.	Word	S	WR, WM, WL	↕	●	●	●	●	270.6~367.4	3	
	17	FUN 38 (S) (HABIN (S))		ASCII to Binary Conversion (16-bit)	Converts 4-digit hexadecimal ASCII code into 16-bit BIN data, and stores the result.	Word	S	WR, WM, WL	↕	●	●	●	●	168.6~272.6	3	
	18	FUN 39 (S) (DHABIN (S))		ASCII to Binary Conversion (32-bit)	Converts 8-digit hexadecimal ASCII code into 32-bit BIN data, and stores the result.	Word	S	WR, WM, WL	↕	●	●	●	●	279.5~407.4	3	
	19	FUN 40 (S) (DABCD (S))		ASCII to BCD Conversion (16-bit)	Converts 4-digit decimal ASCII code into 4-digit BCD data, and stores the result.	Word	S	WR, WM, WL	↕	●	●	●	●	157.3~246.7	3	
	20	FUN 41 (S) (DDABCD (S))		ASCII to BCD Conversion (32-bit)	Converts 8-digit decimalASCII code into 8-digit BCD data, and stores the result.	Word	S	WR, WM, WL	↕	●	●	●	●	256.8~352.3	3	
	21	FUN 42 (S) (ASC (S))		Binary to ASCII Conversion (Specified)	Converts BIN data into ASCII code in a specified number of characters, and stores the result.	Word	S	WR, WM, WL	↕	●	●	●	●	456.2~471.5	3	
	22	FUN 43 (S) (HEX (S))		ASCII to Binary Conversion (Specified)	Converts ASCII code having a specified number of characters into BIN data, and stores the result.	Word	S	WR, WM, WL	↕	●	●	●	●	684.7~705.4	3	10 characters
	23	FUN 44 (S) (SADO (S))		Character String Concatenation	Concatenates specified character strings (~ NULL), and stores the result in a specified location of I/O.	Word	S	WR, WM, WL	↕	●	●	●	●	668.5~859.0	3	10 + 10 characters
	24	FUN 45 (S) (SCMP (S))		Character String Comparison	Compares specified character strings (~ NULL) each other, and stores the result.	Word	S	WR, WM, WL	↕	●	●	●	●	501.1~616.6	3	Comparison in10 characters

*1) These commands are effective for H-252.

FUN commands (3/4) *1

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)	Steps	Remarks
									DER	ERR	SD	V	C	H-252		
FUN Commands	25	FUN 46 (S)	(WTOB (S))	Word to Byte Conversion	Splits and converts 16-bit word data into 8-bit byte data, and stores the result.	Word	S	WR, WM, WL	↕	●	●	●	●	543.9~768.9	3	5 words to 10 bytes
	26	FUN 47 (S)	(BTOW (S))	Byte to Word Conversion	Combines and converts 8-bit byte data into 16-bit word data, and stores the result.	Word	S	WR, WM, WL	↕	●	●	●	●	370.7~595.7	3	10 bytes to 5 words
	27	FUN 48 (S)	(BSHR (S))	Byte Right Shift	Shift right a specified data string by a specified bytes (8 bits * n).	Word	S	WR, WM, WL	↕	●	●	●	●	266.4~416.4	3	10 characters
	28	FUN 49 (S)	(BSHL (S))	Byte Left Shift	Shift left a specified data string by a specified bytes (8 bits * n).	Word	S	WR, WM, WL	↕	●	●	●	●	240.2~390.2	3	10 characters
	29	FUN 50 (S)	(TRSET (S))	Sampling Trace Set	To be used in a trace monitor function.	Word	S	WR, WM, WL	↕	●	●	●	●	153.4~163.7	3	
	30	FUN 51 (S)	(TRASE (S))	Sampling Trace Execution	To be used in a trace monitor function.	Word	S	WR, WM, WL	↕	●	●	●	●	120.6~130.9	3	
	31	FUN 52 (S)	(TRRES (S))	Sampling Trace Reset	To be used in a trace monitor function.	Word	S	WR, WM, WL	↕	●	●	●	●	–	3	
	32	FUN 60 (S)	(BSQR (S))	Binary Square Root	Square root of BIN data is given.	Word	S	WR, WM, WL	↕	●	●	●	●	1212.7~1511.0	3	
	33	FUN 61 (S)	(PGEN (S))	Scan Pulse Generation	ON and OFF operation of a specified bit in a specified number of the scan.	Word	S	WR, WM, WL	↕	●	●	●	●	286.5	3	
	34	FUN 80 (S)		I/O refresh (All points)	Performs I/O refresh of all data of external I/O.	Word	S	WR, WM, WL	●	●	●	●	●	1303	3	
	35	FUN 81 (S)		I/O refresh (I/O specified)	Performs refreshing of input modules only or output modules only.	Word	S	WR, WM, WL	↕	●	●	●	●	1898	3	When number of assignments is 9
	36	FUN 82 (S)		I/O refresh (Arbitray slot))	Performs refreshing of specified number of modules so specified.	Word	S	WR, WM, WL	↕	●	●	●	●	433	3	*3

*1) These commands are effective for H-252.

*2) FUN 50 ~ 52 are usable in LADDER EDITOR.

*3) Processing speed in case of refresh only in a 16-point input module (X16) that is mounted in the slot 0 of the unit 0.

FUN commands (4/4) *

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)		Steps	Remarks
									DER	ERR	SD	V	C	H-252	H-250		
FUN Commands	37	FUN 254 (S)	(BOXC (S))	Box Comment	Comments can be inputted in the box. No execution is provided with this command.	Word	S	WR, WM, WL	●	●	●	●	●	15.9	28.9	3	
	38	FUN 255 (S)	(MEMC (S))	Memo Comments	Memo can be inputted in the box. No execution is provided with this command.	Word	S	WR, WM, WL	●	●	●	●	●	15.9	28.9	3	

*) FUN 254 and FUN255 are usable in LADDER EDITOR.

PID functions *

Category	No.	Ladder symbol	Command symbol	Command name	Description of processing	Size		Type	R7F4	R7F3	R7F2	R7F1	R7F0	Processing speed (μ s)	Steps	Remarks
									DER	ERR	SD	V	C	H-252		
PID Commands	1	FUN 0 (S)	(PIDIT (S))	PID Initialization	Initializes the area for PID operation.	Word	S	WR, WM, WL	●	●	●	●	●	3310.5	3	loop = 1
	2	FUN 1 (S)	(PIDOP (S))	PID Execution Management	Manages PID operations.	Word	S	WR, WM, WL	●	●	●	●	●	881.4	3	
	3	FUN 2 (S)	(PIDCL (S))	PID Calculation	Executes PID operations.	Word	S	WR, WM, WL	●	●	●	●	●	4075.6	3	loop = 1

*) PID commands are effective for H-252.

3.3 Details of Commands

(1) Basic Commands
(2) Arithmetic Commands
(3) Application Commands
(4) Control Commands
(5) Transfer Commands for Sophisticated Function Module
(6) FUN Commands

Name	Start Logical Operation (LD,LDI)

Ladder format	Condition code					Processing time (μ s)		Remarks
n ─┤ ├─ n ─┤/├─	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	•In the refresh mode. •When an internal output is used in the direct mode.
	DER	ERR	SD	V	C	0.2	0.6	
	●	●	●	●	●			
Command format	No. of steps							
LD n LDI n	Conditions			Step		0.5	0.9	When an external I/O is used in the direct mode.
	—			1				

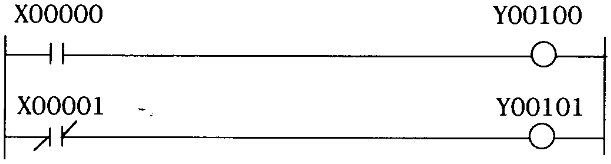
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	I/O No.	○	○	○	○									

[Function]

- n ┝┤├─ LD n — Denotes the start of "Contact a" operation of relay function. The result is "1" when the input is ON.

- n ┝┤/├─ LDI n — Denotes the start of "Contact b" operation of relay function. The result is "1" when the input is OFF.

[Program example]



```
LD   X00000
OUT  Y00100

LDI  X00001
OUT  Y00101
```

[Explanations]

- When input X0000 is ON, output Y00100 is ON. When input X0000 is OFF, output Y00100 is OFF.
- When input X0001 is OFF, output Y00101 is ON. When input X0001 is ON, output Y00101 is OFF

[Precautions]

- L and WL become internal outputs when the link module is not used.
- Edge detections (DIF and DFN) can not be used for LD1. (Can not be inputted).

```
LDI  DIF 0 .... NG
AND  R000
OUT  Y00100
```

Name	Serial Connection of Contact (AND, ANI)

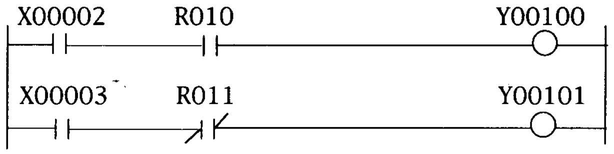
<table>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">n
—| |—
n
—|/|—</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">•In the refresh mode.
•When an internal output is used in the direct mode.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">0.2</td><td rowspan="3">0.6</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">AND n
ANI n</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">0.5</td><td rowspan="2">0.9</td><td rowspan="2">When an external I/O is used in the direct mode.</td></tr>
<tr><td colspan="3">—</td><td colspan="2">1</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	I/O No.	○	○	○	○									

[Function]

- n —| |— AND n   Denotes the " Contact a" AND connection with the previous result.
- n —|/|— ANI n   Denotes the " Contact b" AND connection with the previous result.

[Program example]



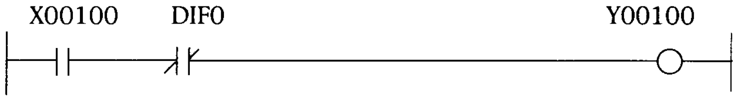
```
LD   X00002
AND  R010
OUT  Y00100

LD   X00003
ANI  R011
OUT  Y00101
```

[Explanations]

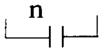
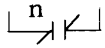
- When both inputs X00002 and R010 are ON, output Y00100 is ON and others are OFF.
- When input X00003 is ON and R011 is OFF, output Y00101 is ON and others are OFF.

[Precautions]

- L and WL become internal outputs when the link module is not used.
- Edge detections (DIF and DFN) can not be used for LD1. (Can not be inputted).

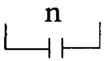


```
LD   X00100
ANI  DIF0.............NG
OUT  Y00100
```

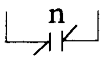
<table>
<tr><td>Name</td><td colspan="8">Parallel Connection of Contact (OR, ORI)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">n

n
</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">•In the refresh mode.
•When an internal output is used in the direct mode.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">0.4</td><td rowspan="3">1.2</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">OR n
ORI n</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">0.7</td><td rowspan="2">1.5</td><td rowspan="2">When an external I/O is used in the direct mode.</td></tr>
<tr><td colspan="3">—</td><td colspan="2">2</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	I/O No.	○	○	○	○									

[Function]

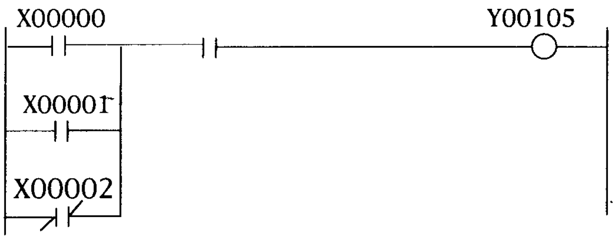
- 

Denotes the " Contact a" OR connection with the previous result.

- 

Denotes the " Contact b" OR connection with the previous result.

[Program example]



```
LD    X00000
OR    X00001
ORI   X00002
OUT   Y00105
```

[Explanations]

Operation result is "1" which causes Y00105 ON, when X00000 is ON, or X00001 is ON, or X00002 is OFF.

[Precautions]

L and WL become equivalent to internal I/O when the link module is not used.
Edge detections (DIF and DFN) can not be used for ORI. (Can not be inputted).

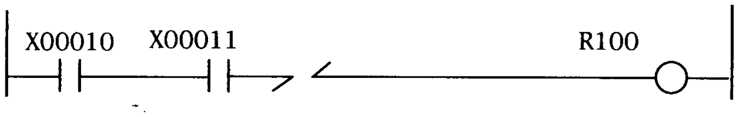
<table>
<tr><td>Name</td><td colspan="8">Negation (NOT)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"></td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">0.4</td><td rowspan="5">1.8</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">NOT</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">2</td></tr>
</table>

<table>
<tr><td rowspan="2" colspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R,
L,
M</td><td>TD, SS, WDT,
MS, TMR, CU,
RCU, CT</td><td>WX</td><td>WY</td><td>WR,
WL,
WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR,
DL,
DM</td></tr>
<tr><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></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></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]

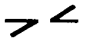
- Inverts the previous operational result.

[Program example]



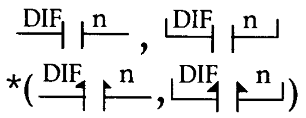
```
LD   X00010
AND  X00011
NOT
OUT  R100
```

[Explanations]

- When both input X00010 and X00011 are ON, the result becomes "1".
  This turns "0" by , and R100 is OFF.

- Other than the above, R100 becomes ON.

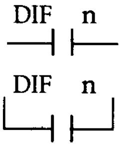
[Precautions]

<table>
<tr><td>Name</td><td colspan="8">Rising Edge Detection (AND DIF, OR DIF)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"> </td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper : *
AND DIF
Lower : *
OR DIF</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">0.6
(1.2)</td><td rowspan="3">3.0
(3.6)</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">AND DIF n
OR DIF n</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">0.8
(1.4)</td><td rowspan="3">3.6
(4.2)</td></tr>
<tr><td colspan="3">AND DIF n</td><td colspan="2">3</td></tr>
<tr><td colspan="3">OR DIF n</td><td colspan="2">4</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	No.												○	0 ~ 511 specified (Decimal number)

[Function]

* When an internal output is used in the refresh mode and in the direct mode. (Values in ( ) are for the cases an external I/O is used in the direct mode.)



- Detects the rising edge of signal. Operation is held in one scan.

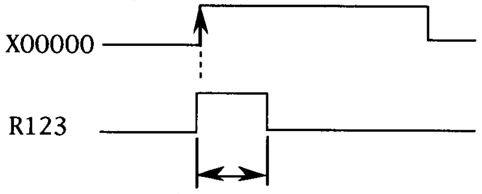
*( ): Expression in these parentheses for LADDER EDITOR.

[Program example]

```
LD   X00000
AND  DFN0
OUT  R123
```

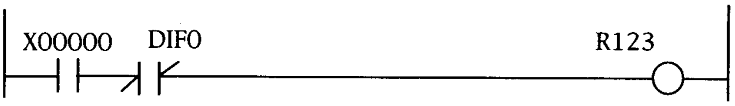
[Explanations]

Time chart

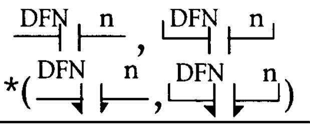


- R123 is ON only in one scan caused by the rising edge of signal X00000.
- In case X00000 is "contact b", the operationa is equivalent to the DFN operation with "contact b".

[Precautions]



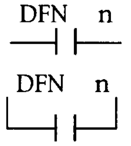
- Double numbering of DIF is unacceptable. If doubled, program execution will be impaired though it is not treated as an error. Be sure to check programs to prevent the double definition.
- DIF can not be used with "contact b" (ANI). (Can not be inputted).

<table>
<tr><td>Name</td><td colspan="8">Falling Edge Detection (AND DFN, OR DFN)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"> </td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper : *
AND DIF
Lower : *
OR DIF</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">0.6
(1.2)</td><td rowspan="3">3.0
(3.6)</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">AND DFN n
OR DFN n</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">0.8
(1.4)</td><td rowspan="3">3.6
(4.2)</td></tr>
<tr><td colspan="3">AND DFN n</td><td colspan="2">3</td></tr>
<tr><td colspan="3">OR DFN n</td><td colspan="2">4</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R,
L,
M</td><td>TD, SS, WDT,
MS, TMR, CU,
RCU, CT</td><td>WX</td><td>WY</td><td>WR,
WL,
WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR,
DL,
DM</td></tr>
<tr><td>n</td><td>No.</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>0 ~ 511 specified
(Decimal number)</td></tr>
<tr><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></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>

* When an internal output is used in the refresh mode and in the direct mode. (Values in ( ) are for the cases an external I/O is used in the direct mode.)

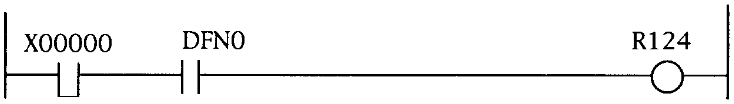
[Function]



- Detects the falling edge of signal. Operation is held in one scan.

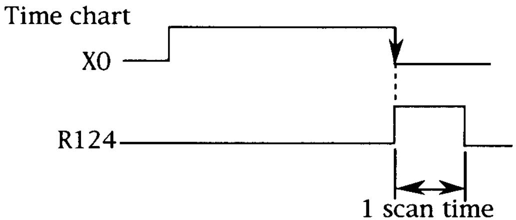
*( ): Expression in these parentheses for LADDER EDITOR.

[Program example]



```
LD    X00000
AND   DFN0
OUT   R124
```

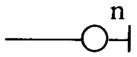
[Explanations]



- R123 is ON only in one scan caused by the falling edge of signal X00000.
- In case X00000 is "contact b", the operation is equivalent to the DIF operation with "contact a".

[Precautions]

- Double numbering of DFN is unacceptable. If doubled, program execution will be impaired though it is not treated as an error. Be sure to check programs to prevent the double definition.
- DFN can not be used with "contact b" (ANI). (Can not be inputted).

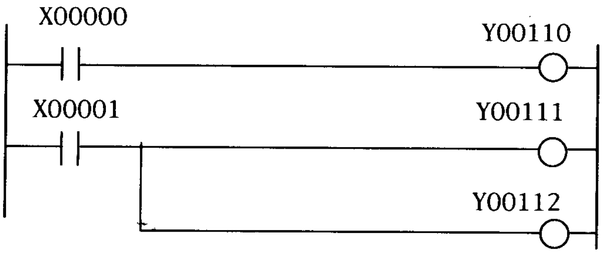
Name	Output to Coil (OUT)							
Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
	DER	ERR	SD	V	C			
	●	●	●	●	●	0.4	0.8	•In the refresh mode. •When an internal output is used in the direct mode.
Command format	No. of steps							
	Conditions			Step		0.5	0.9	When an external I/O is used in the direct mode.
OUT n	—			1				

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	I/O No.		○	○	○									

[Function]

- Makes coil circuit ON when the previous operational result is "1".
- Makes coil circuit OFF when the previous operational result is "0".

[Program example]



```
LD    X00000
OUT   Y00110

LD    X00001
OUT   Y00111
OUT   Y00112
```

[Explanations]

- When input X00000 is ON the result becomes "1" causing Y00100 ON.
- When input X00001 is ON the result becomes "1" causing Y00111 and Y00112 ON.

[Precautions]

- L and WL are equivalent to internal output when the link module is not used.

<table>
<tr><td>Name</td><td colspan="8">Set and Reset of Output to Coil (SET, RES)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">n —O┤ SET, n —O┤RES
*(n —Ⓢ┤ , n —Ⓡ┤)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">•In the refresh mode.
•When an internal output is used in the direct mode.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">0.4</td><td rowspan="3">0.8</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">SET n
RES n</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">0.5</td><td rowspan="2">0.9</td><td rowspan="2">When an external I/O is used in the direct mode.</td></tr>
<tr><td colspan="3">—</td><td colspan="2">1</td></tr>
</table>

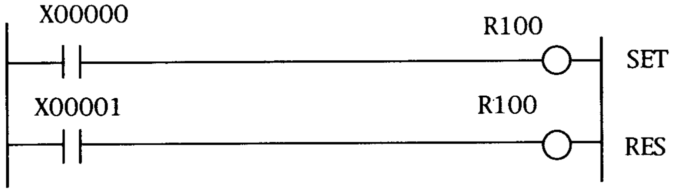
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	I/O No.		○	○										

[Function]

- —O┤ SET (SET n) Makes a device set ON when the previous operational result is "1". Even when the operational result turns to "0", the device will not be turned to OFF.
- —O┤ RES (RES n) Makes coil circuit OFF when the previous operational result is "1".

*( ): Expression in these parentheses is for LADDER EDITOR.

[Program example]



```
LD    X00000
SET   R100
LD    X00001
RES   R100
```

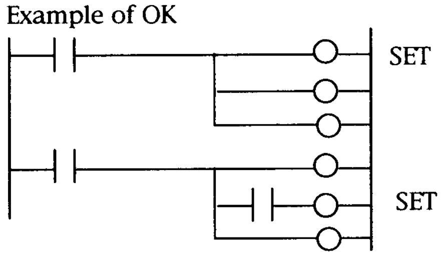
[Explanations]

- When input X0000 is ON, output R100 is ON. R100 is kept ON even when X00000 is OFF.
- When input X00001 is ON, output R100 is OFF.
- When both input X00000 and X00001 are ON, only the latest execution in the program is valid.

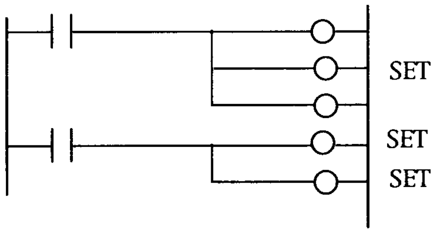
[Precautions]

- When "Set/Reset coil" is used in tandem coil group, locate this group on the top of the circuit or provide a contact (R7E4 or a normally ON or the like) just before the group.

Example of OK



Example of NG



(In the case of NG, an input is disabled caused by the Ladder error.)

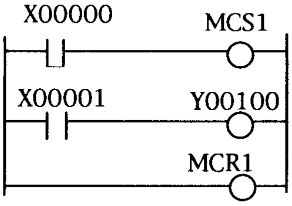
<table>
<tr><td>Name</td><td colspan="9">Master Control, Set and Reset (MCS, MCR)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">MCS n, MCR n
*(MCS n, MCR n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">Upper values: for MCS n</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">0.6</td><td rowspan="3">3.0</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">MCS n
MCR n</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">0.4</td><td rowspan="3">1.8</td><td rowspan="3">Lower values: for MCR n</td></tr>
<tr><td colspan="3">MCS n</td><td colspan="2">3</td></tr>
<tr><td colspan="3">MCR n</td><td colspan="2">2</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	No.												○	0 ~ 49 specified (Decimal number)

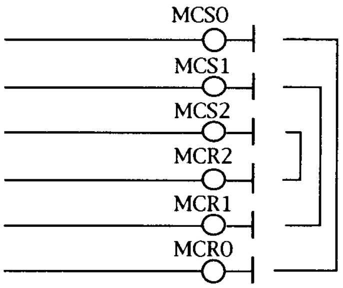
[Function]

- Controls inputs to circuits inserted between Set (MCS n) and Reset (MCR n) of master control. ("AND" operation is performed by each input and MCS).
- Master controls can be piled up to eight layers. (H-250 : seven layers).
  *( ): Expression in these parentheses is for LADDER EDITOR.

[Program example]



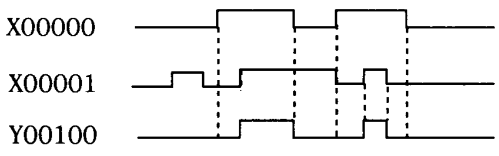
```
LD  X00000
MCS1
LD  X00001
MCR1
```



Eight layers maximum. (H-250 : seven layers)

[Explanations]

Time chart



- When input X00000 is ON, the output Y00100 in the circuit inserted between MCS and MCR is made ON/OFF in response to the input X00001.
- When the input X00000 is OFF, the output Y00100 in the circuit inserted between MCS and MCR is OFF irrespective of the input X00001.

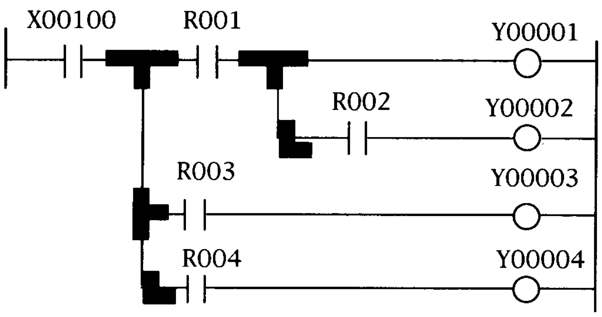
[Precautions]

- Numbers of Master control are 0. ~ 49 (Decimal). Double numbering is acceptable.
- Master controls can be piled up to eight layers. (H-250 : seven layers).
- MCS and MCR of Master control must be used as pairs.

<table>
<tr><th>Name</th><th colspan="8">Store, Read or Clear Operation Result (Ladder Branch)</th></tr>
<tr><th>Ladder format</th><th colspan="5">Condition code</th><th colspan="2">Processing time (μ s)</th><th>Remarks</th></tr>
<tr><td rowspan="3">┬ Store
┼ Read
└ Clear</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">—</td><td rowspan="5">—</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">MPS Store
MRD Read
MPP Clear</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">0</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		

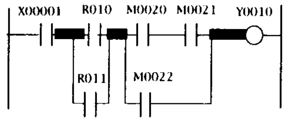
[Function]



```
LD  X00100
MPS
AND R001
MPS
OUT Y00001
MPP
AND R002
OUT Y00002
MRD
AND R003
OUT Y00003
MPP
AND R004
OUT Y00004
```

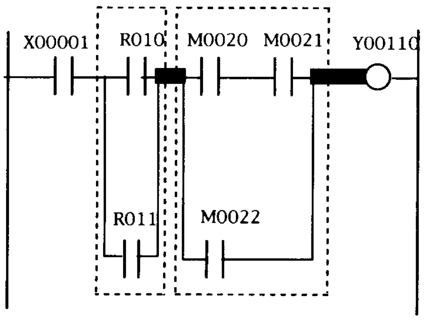
- MPS stores the last operation result before MPS command (Push).
- MRD reads the operation result that is stored by the command MPS and continues operation in response to a next command.
- MPP reads the operation result stored by the last MPS command before MPP and continues operation in response to a next command, and clears the operation result after completes the operation (Pull).

[Precautions]

<table>
<tr><td>Name</td><td colspan="9">Logical Block Serial Connection (ANB)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"> </td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">—</td><td rowspan="5">—</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">ANB</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">0</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		

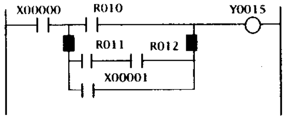
[Function]



LD X00001
LD R010
OR R011
ANB
LD M0020
AND M0021
OR M0022
ANB
OUT Y00110

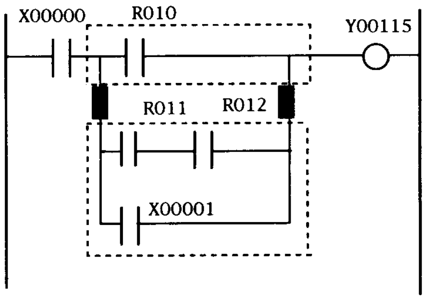
Performs AND connection of the logical operation blocks of Ladder diagram expression as shown by broken lines.

[Precautions]

<table>
<tr><td>Name</td><td colspan="8">Logical Block Parallel Connection (ORB)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"> </td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">0.2</td><td rowspan="5">0.6</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">ORB</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">1</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><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></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></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]



```
LD   X00000
LD   R010
LD   R011
AND  R012
ORB
OR   X00001
ANB
OUT  Y00115
```

Performs OR connection of the logical operation blocks of Ladder diagram expression as shown by broken lines.

[Precautions]

<table>
<tr><td>Name</td><td colspan="9">On - Delay Timer (ON DELAY TIMER)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">TD n t
——○┤
× s</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">1.8</td><td rowspan="5">6.3</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">OUT TD n t s</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">5</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>n</td><td>Timer No.</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>0 ~ 255 Specified (Decimal number)</td></tr>
<tr><td>t</td><td>Time base</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>.01s, .1s, 1s</td></tr>
<tr><td>s</td><td>Set value</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>1 ~ 65,535 Specified (Decimal number)</td></tr>
</table>

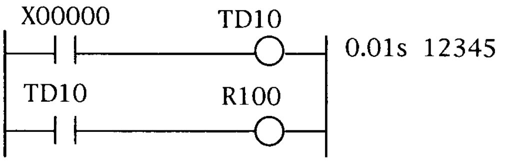
[Function]

- Elapsed time will be reset and counted while the condition for start is ON. Coil will be ON when the elapsed time >= the pre-set value.
- Elapsed time will be cleared when the condition for start is OFF, and the coil will be OFF immediately.
- The elapsed time of the on delay timer is stored in TCn. Its maximum value is 65 535 (Decimal).
- Input 0 ~ 255 (Decimal) for timer numbers. Double numbering is not acceptable.
- To change the elapsed time use arithmetic commands, the forced set or the like. When the change is made during RUN, the operation will follow the new value at that moment.
- In the case of the change prior to or during an execution of a program, set the values on WX, WY, WR, WL, WM. The on delay timer operation will result in reading the newly set value at each scan.
- When the change is made on the time base and/or the set value by a modification of the program during RUN, the operation will follow the new value of the time base and/or the set value at that moment.

[Precautions]

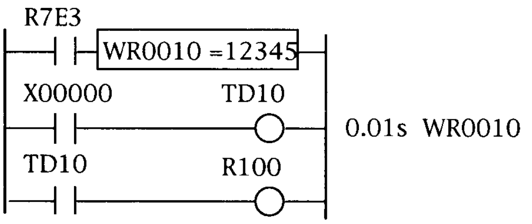
- The time base 0.01s can be used only for the timer numbers 0 ~ 63 (64 points).
- The time bases 0.1s and 1s can be used for all numbers 0 ~ 255 (Decimal).
- The timers can be used on up to 256 points for the total of TD, SS, MS, TMR, WDT. Double numbering is not acceptable. (For instance TD1 and SS1 can not be used simultaneously).

[Program example]



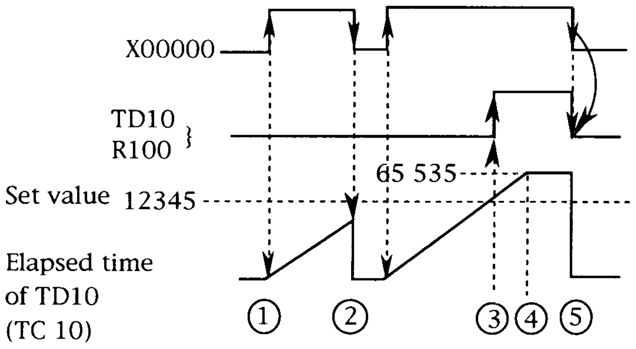
```
LD   X00000
OUT  TD10  0.01S  12345
LD   TD10
OUT  R100
```

Example in which the word I/O is used for the settig value in the above circuit.



```
LD   R7E3
[
WR0010 =12345
]
LD   X00000
OUT  TD10  0.01S  WR0010
LD   TD10
OUT  R100
```

[Explanations]



① When input X00000 is ON, the elapsed time of TD is reset and counted.

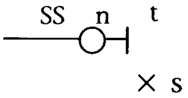
② When input X00000 is OFF, the elapsed time of TD is cleared.

③ TD10 is ON, when the elapsed time >= the pre-set value.

④ While X00000 is ON, the elapsed time increases up to 65 535 maximum.

⑤ When X00000 is OFF, TD10 is also OFF and the elapsed time also is cleared.

- Example of the word I/O used as the set value
  The set value is set on the word I/O at RUN start.
  As another solution specify the power failure registration function on the value of the word I/O.

Name	Single Shot (SINGLE SHOT)							
Ladder format	Condition code					Processing time (μ s)		Remarks
 SS n t × s	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
	DER	ERR	SD	V	C	1.8	6.3	
	●	●	●	●	●			
Command format	No. of steps							
	Conditions				Step			
OUT SS n t s	—				5			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	Timer No.												○	0 ~ 255 Specified (Decimal number)
t	Time base													.01s, .1s, 1s
s	Set value					○	○	○					○	1 ~ 65,535 Specified (Decimal number)

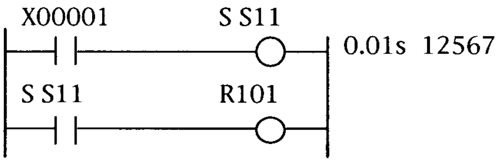
[Function]

- Detects the rising edge of starting condition, starts reset of the elapsed time, and makes the coil ON.
- The coil will be OFF when the elapsed time >= a pre-set value. When a rising edge is repeatedly detected while the elapsed time < the pre-set value, the elapsed time will be returned to zero and the time counting will be re-started (re-triggered).
- The elapsed time of the on delay timer is stored in TCn. Its maximum value is 65 535 (Decimal). Input 0 ~ 255 (Decimal) for timer numbers. Double numbering is not acceptable.
- To change the elapsed time use arithmetic commands, the forced set or the like. When the change is made during RUN, the operation will follow the new value at that moment.
- In the case of the change prior to or during an execution of a program, set the values on WX, WY, WR, WL, WM. The on delay timer operation will result in reading the newly set value at each scan.
- When the change is made on the time base and/or the set value by a modification of the program during RUN, the operation will follow the new value of the time base and/or the set value at that moment.

[Precautions]

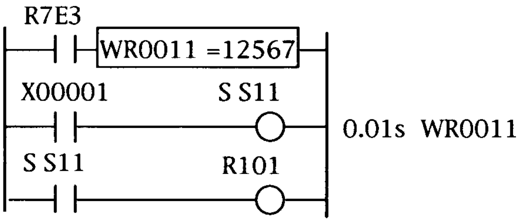
- The time base 0.01s can be used only for the timer numbers 0 ~ 63 (64 points).
- The time bases 0.1s and 1s can be used for all numbers 0 ~ 255 (Decimal).
- The timers can be used on up to 256 points for the total of TD, SS, MS, TMR, WDT. Double numbering is not acceptable. (For instance TD1 and SS1 can not be used simultaneously).
- As the starting condition for Single Shot is the detection of rising edge, the detection can not be made until the first complete scan comes after RUN starts.

[Program example]



```
LD   X00001
OUT  S S11  0.01S  12567
LD   S S11
OUT  R101
```

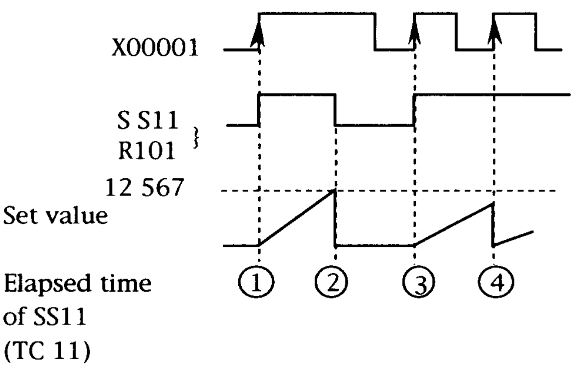
In the above example the word I/O is used as the set value.



```
LD   R7E3
[
WR0011 =12567
]
LD   X00001
OUT  S S11   0.01S  WR0011
LD   S S11
OUT  R101
```

[Explanations]

Time chart



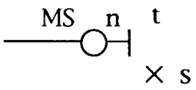
① The elapsed time is reset and counted at the rising edge of X00001, and SS11 is ON.

② When the the elapsed time >=set value, SS11 is OFF. Although X00001 remains ON at this moment, this operation is ignored because the starting condition is an edge trigger in Single Shot.

③ SS11 is ON at the rising edge of the next ON operation of X00001, and the elapsed time is reset and counted.

④ When a rising edge of X00001 is detected before the elapsed time reaches the set value, the Single Shot timer is re- triggered. The elapsed time is then returned to zero and is re-started to increase. SS11 still remains ON.

- Example of the word I/O used as the set value.
  The set value is set on the word I/O at RUN start.
  As another solution specify the power failure registration function on the value of the word I/O.

<table>
<tr><td>Name</td><td colspan="8">Mono - Stable Timer (MONO STABLE TIMER)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"> </td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">1.8</td><td rowspan="5">6.3</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">OUT MS n t s</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">5</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	Timer No.												○	0 ~ 255 Specified (Decimal number)
t	Time base													.01s, .1s, 1s
s	Set value					○	○	○					○	1 ~ 65,535 Specified (Decimal number)

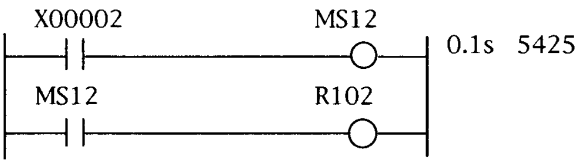
[Function]

- Detects the rising edge of starting condition, starts reset of the elapsed time, and makes the coil ON.
- The coil will be OFF when the elapsed time >= a pre-set value. While MS is ON, the rising edge, as a starting condition, will be ignored.
- The elapsed time of the mono stable timer is stored in TCn. Its maximum value is 65 535 (Decimal).
- Input 0 ~ 255 (Decimal) for timer numbers. Double numbering is not acceptable.
- To change the elapsed time use arithmetic commands, the forced set or the like. When the change is made during RUN, the operation will follow the new value at that moment.
- In the case of the change prior to or during an execution of a program, set the values on WX, WY, WR, WL, WM. The mono stable timer operation will result in reading the newly set value at each scan.
- When the change is made on the time base and/or the set value by a modification of the program during RUN, the operation will follow the new value of the time base and/or the set value at that moment.

[Precautions]

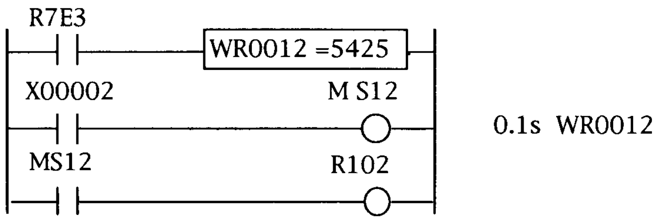
- The time base 0.01s can be used only for the timer numbers 0 ~ 63 (64 points).
- The time bases 0.1s and 1s can be used for all numbers 0 ~ 255 (Decimal).
- The timers can be used on up to 256 points for the total of TD, SS, MS, TMR, WDT. Double numbering is not acceptable. (For instance TD1 and SS1 can not be used simultaneously).
- As the starting condition for Mono Stable timer is the detection of rising edge, the detection can not be made until the first complete scan comes after RUN starts.

[Program example]



```
LD   X00002
OUT  MS12  0.1S  5425
LD   MS12
OUT  R102
```

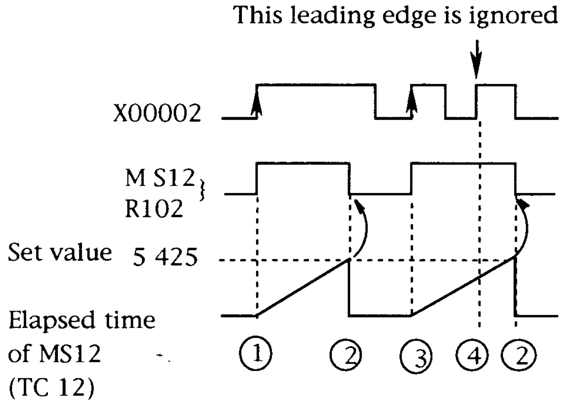
- In the above example the word I/O is used as the set value.



```
LD   R7E3
[
WR0012 =5425
]
LD   X00002
OUT M S12   0.1S  WR0012
LD   MS12
OUT  R102
```

[Explanations]

Time chart



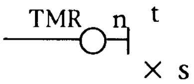
① The elapsed time is reset and counted at the rising edge of X00002, and MS12 is ON.

② When the elapsed time >= the set value, MS12 is OFF. Although X00002 remains ON at this moment, this operation is ignored because the starting condition is an edge trigger in Mono Stable timer.

③ MS12 is ON at the rising edge of the next ON operation of X00002, and the elapsed time is reset and counted.

④ When a rising edge of X00002 is detected before the elapsed time reaches the set value, the Mono Stable timer ignores this rising edge.

- Example of the word I/O used as the set value.
  The set value is set on the word I/O at RUN start.
  As another solution specify the power failure registration function on the value of the word I/O.

<table>
<tr><td>Name</td><td colspan="8">Integral Timer (INTEGRAL TIMER)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">
</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">1.8</td><td rowspan="5">6.3</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">OUT TMR n t s</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">5</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	Timer No.												○	0 ~ 255 Specified (Decimal number)
t	Time base													.01s, .1s, 1s
s	Set value					○	○	○					○	1 ~ 65,535 Specified (Decimal number)

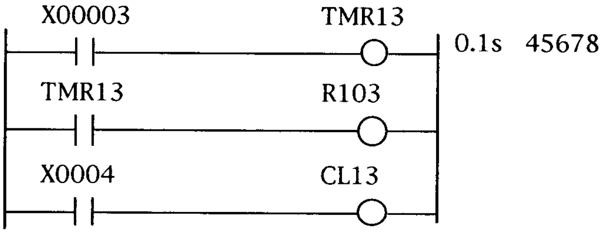
[Function]

- Elapsed time will be counted while the condition for start is ON. The elapsed time will not be cleared even when the starting condition is OFF, but will be counted again when the starting condition will return to ON.
- When the elapsed time >= the set value, the coil will be ON and will not be OFF until the clearing input CLn will become ON.
- The elapsed time of the Integral timer is stored in TCn. Its maximum value is 65 535 (Decimal).
- Input 0 ~ 255 (Decimal) for timer numbers. Double numbering is not acceptable.
- To change the elapsed time use arithmetic commands, the forced set or the like. When the change is made during RUN, the operation will follow the new value at that moment.
- In the case of the change prior to or during an execution of a program, set the values on WX, WY, WR, WL, WM. The on delay timer operation will result in reading the newly set value at each scan.
- When the change is made on the time base and/or the set value by a modification of the program during RUN, the operation will follow the new value of the time base and/or the set value at that moment.

[Precautions]

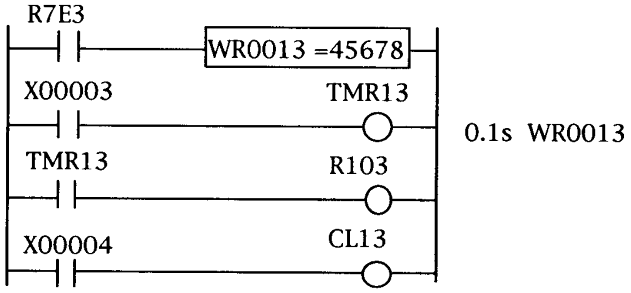
- The time base 0.01s can be used only for the timer numbers 0 ~ 63 (64 points).
- The time bases 0.1s and 1s can be used for all numbers 0 ~ 255 (Decimal).
- The timers can be used on up to 256 points for the total of TD, SS, MS, TMR, WDT. Double numbering is not acceptable. (For instance TD1 and SS1 can not be used simultaneously).
- While The timer clear (CLn) is ON, ON of the starting condition is ignored.

[Program example]



```
LD   X00003
OUT  TMR13  0.1S  45678
LD   TMR13
OUT  R103
LD   X00004
OUT  CL13
```

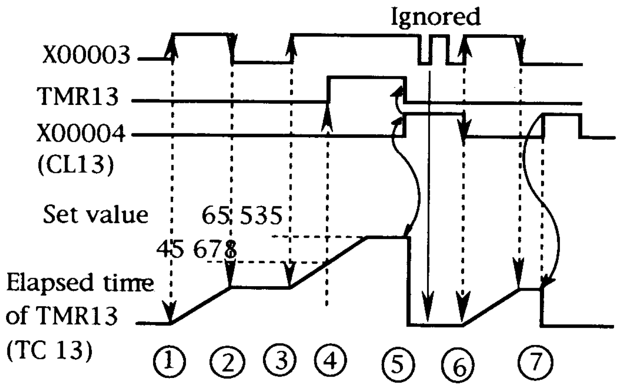
- In the above example the word I/O is used as the set value.



```
LD  R7E3
[
WR0013 =45678
]
LD  X00003
OUT TMR13  0.1S  WR0013
LD  TMR13
OUT R103
LD  X00004
OUT CL13
```

[Explanations]

Time chart



① The elapsed time is reset and counted while X00003 is ON

② When X00003 is OFF, the elapsed time counting is stopped and the elapsed time is held as usual.

③ When X00003 is ON again, the elapsed time is counted again.

④ When the elapsed time >= the set value, the timer coil TMR13 will be ON. This ON will be maintained till the timer clear will be turned to ON.

⑤ When the timer clear (CL13) is ON, both the timer coil and the elapsed time are cleared.

⑥ While the timer clear (CL13) is ON, the starting condition is ignored.

⑦ The elapsed time will be cleared by the timer clear.

- Example of the word I/O used as the set value.
  The set value is set on the word I/O at RUN start.
  As another solution specify the power failure registration function on the value of the word I/O.
- The timer clear will be operated on the last condition before the execution of the timer coil command.

<table>
<tr><td>Name</td><td colspan="9">Watch Dog Timer (WATCH DOG TIMER)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">WDT n t
× S1
× S2</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">48.9
~
68.6</td><td rowspan="5">78.9
~
107.6</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">OUT WDT n t s1 s2</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">7</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	Timer No.												○	0 ~ 255 Specified (Decimal number)
t	Time base													.01s, .1s, 1s
s1	1st Set value					○	○	○					○	1 ~ 65,535 Specified (Decimal number)
s2	2nd Set value					○	○	○					○	1 ~ 65,535 Specified (Decimal number)

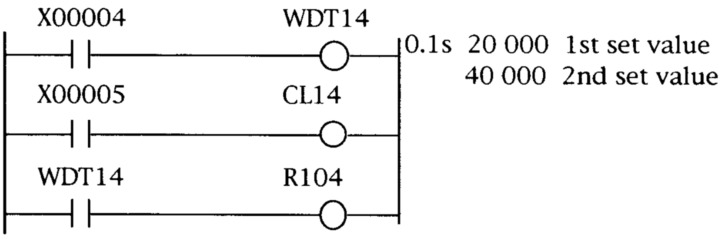
[Function]

- Elapsed time will be reset and counted when the condition for start will be ON. Coil will not be ON when the clear input CLn is accessed during the status (1st set value <= the elapsed time < 2nd set value). The coil will be ON when the clear input CLn is accessed during the status (The elapsed time < 1st set value), or when the status becomes (2nd set value <= the elapsed time). All elapsed times will be cleared when the condition for start is OFF.
- The elapsed time of the Watch Dog timer is stored in TCn. Its maximum value is 65 535 (Decimal).
- Input 0 ~ 255 (Decimal) for timer numbers. Double numbering is not acceptable.
- To change the elapsed time use arithmetic commands, the forced set or the like. When the change is made during RUN, the operation will follow the new value at that moment.
- In the case of the change prior to or during an execution of a program, set the values on WX, WY, WR, WL, WM. The watch dog timer operation will result in reading the newly set value at each scan.
- When the change is made on the time base and/or the set value by a modification of the program during RUN, the operation will follow the new value of the time base and/or the set value at that moment.

[Precautions]

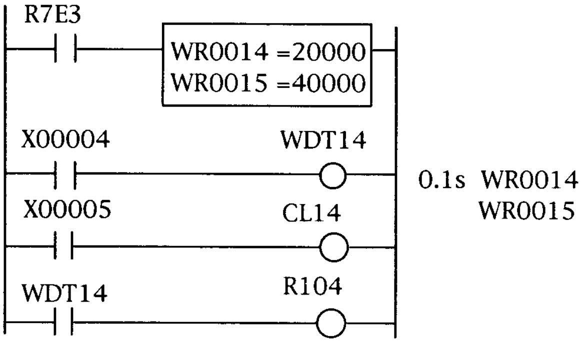
- The time base 0.01s can be used only for the timer numbers 0 ~ 63 (64 points).
- The time bases 0.1s and 1s can be used for all numbers 0 ~ 255 (Decimal).
- The timers can be used on up to 256 points for the total of TD, SS, MS, TMR, WDT. Double numbering is not acceptable. (For instance TD1 and SS1 can not be used simultaneously).
- Be sure to set as {1st set value (S1) < 2nd set value (S2)}. If S1 >= S2, WDTn coil will be ON when the condition for start is ON.

[Example program]



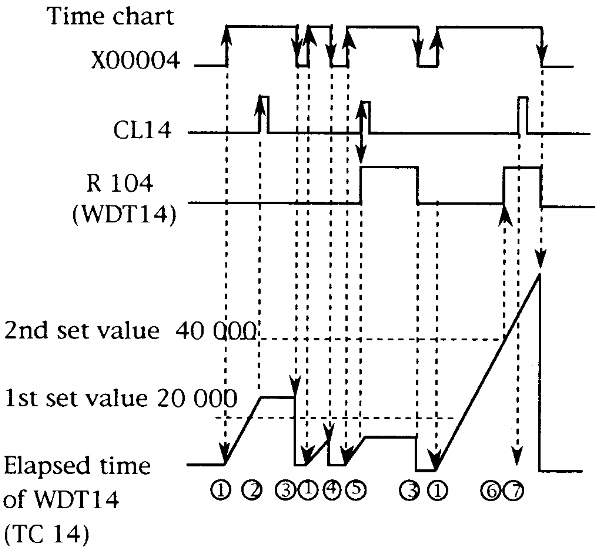
```
LD   X00004
OUT  WDY14 0.1S 20000 40000
LD   X00005
OUT  CL14
LD   WDT14
OUT  R104
```

- In the above example the word I/O is used as the set value.



```
LD  R7E3
[
WR0014 =20000
WR0015 =40000
]
LD  X00004
OUT WDT14  0.1S WR0014 WR0015
LD  X00005
OUT CL14
LD  WDT14
OUT R104
```

[Explanation]



① The elapsed time is reset while X00004 is ON.

② R104 (WDT14) is not ON because the operation is judged as normal from the status that the watch dog clear (CL14) became ON when the elapsed time exceeded 1st set value but before exceeds 2nd set value.

③ When X00004 is OFF, the elapsed time and WDT coil output is cleared.

④ WDT coil is not ON and the elapsed time is cleared because the condition for start was OFF before the elapsed time exceeds 1st set value.

⑤ R104 (WDT14) is ON because the operation is judged as abnormal from the status that the watch dog clear (CL14) became ON before the elapsed time exceeds 1st set value . The elapsed value will be held unchanged.

⑥ R104 (WDT14) is ON because the operation is judged as abnormal from the status that the watch dog clear (CL14) was not ON even when the elapsed time exceeded 2nd set value. The elapsed time continues to be counted from this point.

⑦ "Watch dog clear (CL14) ON" will be ignored in case it is made after ON of WDT coil following the occurrence of excess in elapsed time above 2nd set value.

- The timer clear will be operated on the last condition before the execution of WDT coil command.

- Example of the word I/O used as the set value.
  The set value is set on the word I/O at RUN start.
  As another solution specify the power failure registration function on the value of the word I/O.

<table>
<tr><td>Name</td><td colspan="8">Counter (COUNTER)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"> </td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">2.0</td><td rowspan="5">6.5</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">OUT CU n s</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">5</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	Counter No.												○	0 ~ 511 Specified (Decimal number)
s	Set value					○	○	○					○	1 ~ 65,535 Specified (Decimal number)

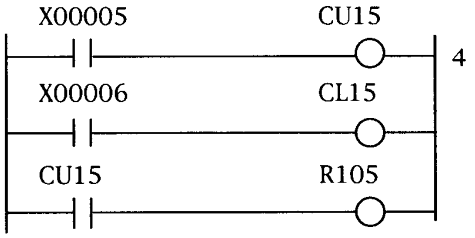
[Function]

- Detects the rising edge of the condition for start and adds one unit of the elapsed value to the operation. Coil will be ON when the elapsed value >= the set value. The coil that was made ON will be turned to OFF and the elapsed value will consequently be returned to zero when the counter clear (CLn) becomes ON.
- The elapsed value of the Counter is stored in TCn. Its maximum value is 65 535 (Decimal).
- Input 0 ~ 511 (Decimal) for counter numbers. Double numbering is not acceptable.
- To change the elapsed value use arithmetic commands, the forced set or the like. When the change is made during RUN, the operation will follow the new value at that moment.
- In the case of the change prior to or during an execution of a program, set the values on WX, WY, WR, WL, WM. The counter operation will result in reading the newly set value at each scan.
- When the change is made on the elapsed value base and/or the set value by a modification of the program during RUN, the operation will follow the new value of the elapsed value and/or the set value at that moment.

[Precautions]

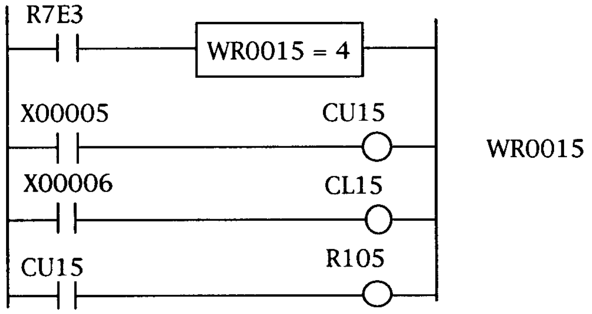
- Counters can be used for up to 512 points (No. 0 ~ 511). The same area is to be shared with timers up to 256 points (No. 0 ~ 255).
- Double numbering is not acceptable for both timers and/or counters. (For instance TD1 and CU1 must not be used simultaneously).
- Rising edge of the condition for start will be ignored when the counter clear (CLn) remains ON.
- As the condition for start of the counter is the edge detection, the condition can not be detected until the 1st complete scan comes after RUN starts.
- The counter will normally be ON and will become the coil to be controlled by CLn, when it is set to zero.

[Example program]



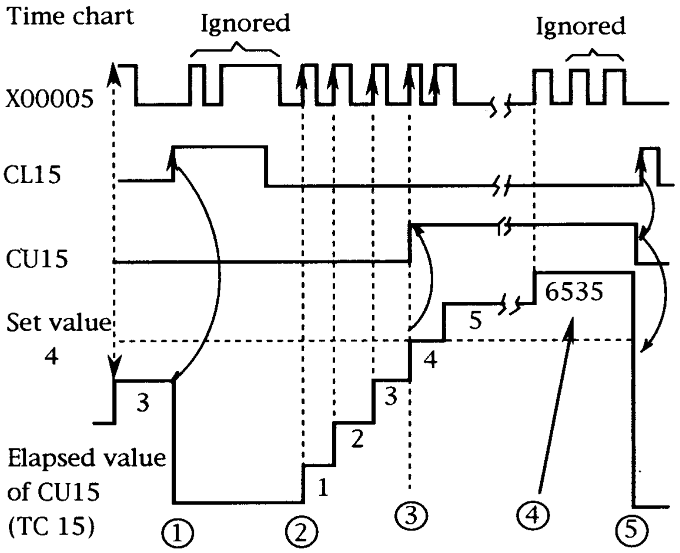
```
LD   X00005
OUT  CU15  4
LD   X00006
OUT  CL15
LD   CU15
OUT  R105
```

- In the above example the word I/O is used as the set value.



```
LD  R7E3
[
WR0015 =4
]
LD  X00005
OUT CU15  WR0015
LD  X00006
OUT  CL15
LD  CU15
OUT  R105
```

[Explanation]



① The elapsed value (counted value) is cleared by the counter clear (CL15). While the counter clear is ON, the elapsed value is not re-counted.

② The elapsed value is counted at the leading edge of X00005.

③ The counter coil (CU15) is ON, when the set value <= the elapsed value.

④ Maximum counted value is 65 535 (Decimal).

⑤ The elapsed value (counted value) and the counter coil are cleared by the counter clear (CL15).

- The clear is operated on the last condition before the counter coil command.

- Example of the word I/O used as the set value.
  The set value is set on the word I/O at RUN start.
  As another solution specify the power failure registration function on the value of the word I/O.

<table>
<tr><td>Name</td><td colspan="8">Ring Counter (RING COUNTER)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"> </td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">2.0</td><td rowspan="5">6.5</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">OUT RCU n s</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">5</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	Counter No.												○	0 ~ 511 Specified (Decimal number)
s	Set value					○	○	○					○	1 ~ 65,535 Specified (Decimal number)

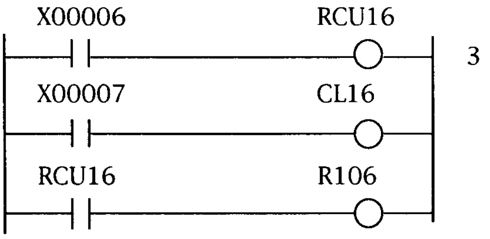
[Function]

- Detects the rising edge of the condition for start and adds one unit of the elapsed value to the operation. Coil will be ON for one scan time. The elapsed value will become zero, when the counter clear (CLn) is ON. The coil will also be OFF.
- The elapsed value of the Ring Counter is stored in TCn. Its value is limited to the set value.
- Input 0 ~ 511 (Decimal) for timer numbers. Double numbering is not acceptable.
- To change the elapsed value use arithmetic commands, the forced set or the like. When the change is made during RUN, the operation will follow the new value at that moment.
- In the case of the change prior to or during an execution of a program, set the values on WX, WY, WR, WL, WM. The ring counter operation will result in reading the newly set value at each scan.
- When the change is made on the elapsed value and/or the set value by a modification of the program during RUN, the operation will follow the new value of the elapsed value and/or the set value at that moment.

[Precautions]

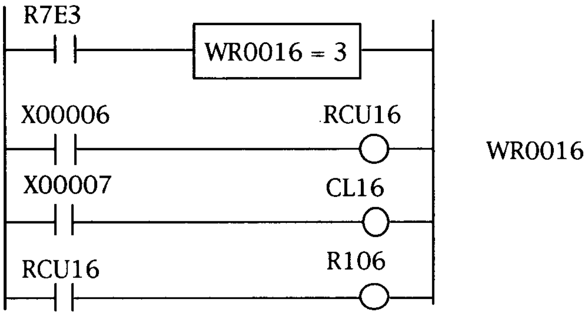
- Counters can be used for up to 512 points (No. 0 ~ 511). The same area is to be shared with timers up to 256 points (No. 0 ~ 255).
- Double numbering is not acceptable for both timers and/or counters. (For instance TD1 and CU1 must not be used simultaneously).
- Rising edge of the condition for start will be ignored when the counter clear (CLn) remains ON.
- As the condition for start of the counter is the edge detection, the condition can not be detected until the 1st complete scan comes after RUN starts.
- The counter will normally be ON and will become the coil to be controlled by CLn, when the ring counter is set to zero.

[Example program]



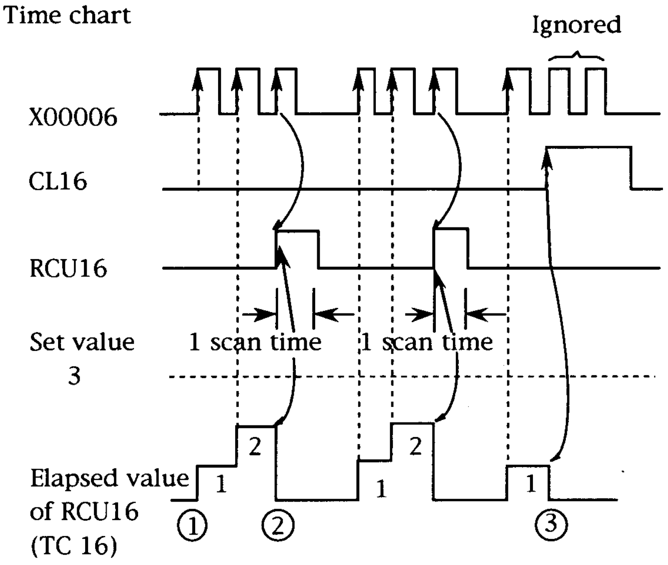
```
LD   X00006
OUT  RCU16   3
LD   X00007
OUT  CL16
LD   RCU16
OUT  R106
```

- In the above example the word I/O is used as the set value.



```
LD   R7E3
[
WR0016 = 3
]
LD   X00006
OUT RCU16  WR0016
LD   X00007
OUT  CL16
LD   RCU16
OUT  R106
```

[Explanation]



① The elapsed value (counted value) is reset by the leading edge of X00006.

② When the set value = the elapsed value, the counter coil (RCU16) is ON for one scan time, and the elapsed value is cleared.

③ When the counter clear (CL16) is ON, the elapsed value is cleared. The elapsed value is not counted while the counter clear is ON.

- The clear is operated on the last condition before the counter coil command.

- Example of the word I/O used as the set value.
  The set value is set on the word I/O at RUN start.
  As another solution specify the power failure registration function on the value of the word I/O.

Name	Up/Down Counter: Up(CTU n), Down(CTD n) (UP/DOWN COUNTER UP, DOWN)

Ladder format	Condition code					Processing time (μ s)		Remarks
CTU n s CTD n	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	Upper values: for CTU Lower values: for CTD
	DER	ERR	SD	V	C	2.0	6.5	
	●	●	●	●	●			
Command format	No. of steps					1.6	5.2	
OUT CTU n s OUT CTD n	Conditions			Step				
	CTU			5				
	CTD			3				

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	Counter No.												○	0 ~ 511 Specified (Decimal number)
s	Set value					○	○	○					○	1 ~ 65,535 Specified (Decimal number)

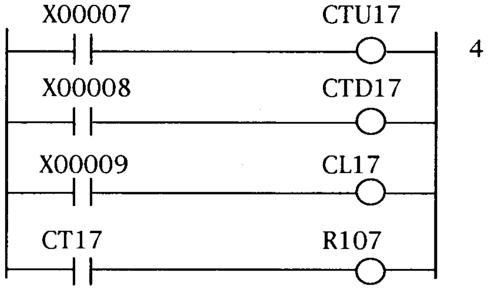
[Function]

- At the rising edge of the condition for start of UP of the Up/Down counter, adds one unit of the elapsed value to the operation. At the rising edge of the condition for start of DOWN, subtracts one unit of the elapsed value from the operation. Coil will be ON when the elapsed value >= the set value. Coil will be OFF when the elapsed value < the set value.
- Coil of the Up/Down counter is CTn.
- The elapsed value of Up/Down counter is stored in TCn. Its value is 0 ~ 65 35 (Decimal).
- Input 0 ~ 511 (Decimal) for timer numbers. Double numbering is not acceptable.
- To change the elapsed value use arithmetic commands, the forced set or the like. When the change is made during RUN, the operation will follow the new value at that moment.
- In the case of the change prior to or during an execution of a program, set the values on WX, WY, WR, WL, WM. The up/down counter operation will result in reading the newly set value at each scan.
- When the change is made on the elapsed value and/or the set value by a modification of the program during RUN, the operation will follow the new value of the elapsed value and/or the set value at that moment.

[Precautions]

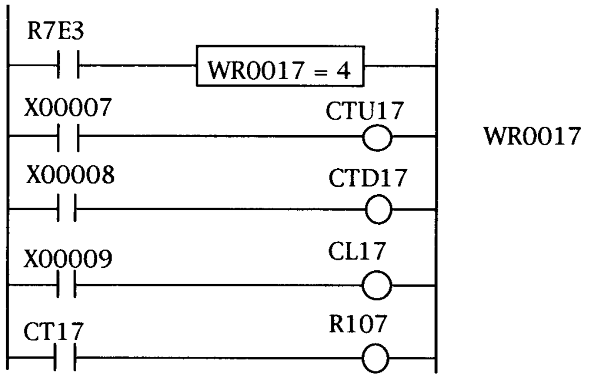
- Counters can be used for up to 512 points (No. 0 ~ 511). The same area is to be shared with timers up to 256 points (No. 0 ~ 255).
- Double numbering is not acceptable for both timers and/or counters. (For instance TD1 and CU1 must not be used simultaneously).
- Use the same number for the up coil and down coil of the up-down counter.
- Rising edge of the condition for start will be ignored when the counter clear (CLn) remains ON.
- As the condition for start of the counter is the edge detection, the condition can not be detected until the 1st complete scan comes after RUN starts.
- The counter will normally be ON and will become the coil to be controlled by CLn, when the ring counter is set to zero.

[Example program]



```
LD   X00007
OUT  CTU17   4
LD   X00008
OUT  CTD17
LD   X00009
OUT  CL17
LD    CT17
OUT  R107
```

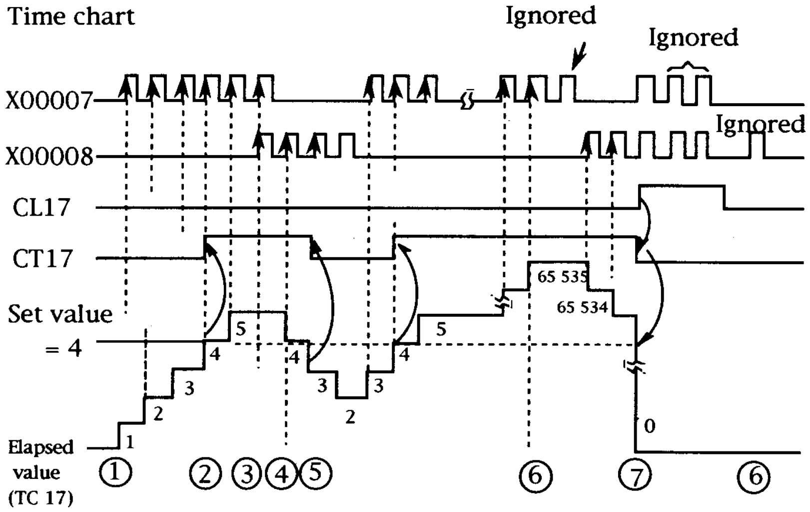
- In the above example the word I/O is used as the set value.



```
LD   R7E3
[
WR0017 = 4
]
LD   X00007
OUT CTU17  WR0017
LD   X00008
OUT  CTD17
LD   X00009
OUT  CL17
LD    CT17
OUT  R107
```

[Explanation]

Time chart



① The elapsed value (counted value) is up-counted at the rising edge of X00007.

② When the set value <= the elapsed value, the counter coil (CT17) is ON.

③ The elapsed value remains unchanged in case the start condition of the UP coil and that of DOWN coil are simultaneously ON.

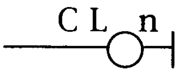
④ The elapsed value is down- counted at the rising edge X00008.

⑤ At the set value > the elapsed value, the counter coil is OFF.

⑥ The elapsed value is 0 ~ 65 535.

⑦ The elapsed value and the counter coil are cleared when the counter clear (CL17) is ON. The elapsed value is not counted while the counter clear is ON.

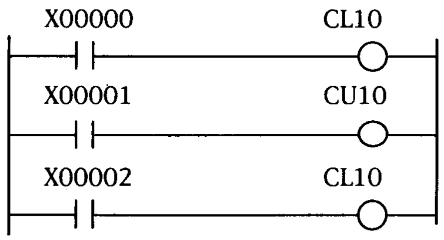
- The clear is operated on the last condition before the counter coil command.
- Example of the word I/O used as the set value.
  The set value is set on the word I/O at RUN start.
  As another solution specify the power failure registration function on the value of the word I/O.

Name	Couter Clear (COUTER CLEAR)							
Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
	DER	ERR	SD	V	C	0.4	0.8	
	●	●	●	●	●			
Command format	No. of steps							
	Conditions			Step				
OUT CL n	—			1				

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	Counter No.												○	0 ~ 511 Specified (Decimal number)

[Function]

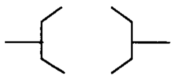
- Clears the elapsed values of TD, SS, MS and the integral timer, and at the same time makes the timer coils OFF.
- In the case of WDT, checks the time for watch function.
- In the case of the counter, clears the elapsed value, and at the same time makes counter coil OFF.
- The counter clear will be operated on the last condition before the execution of the coil command of the counter or the timer that the clear coil specifies on.



1) When X00000 is ON, CL10 that is sequenced prior to CU10 is ON, and CU10 is cleared.
2) Even when X00002 is ON but X00001 is OFF, CU10 will not be cleared because CL10 will be brought to OFF by the circuit that is sequenced prior to the execution of CU10.

[Precautions]

- Numbers should be the same as those of the timer and counter.

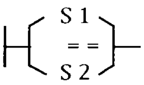
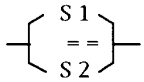
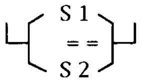
<table>
<tr><td>Name</td><td colspan="8">Start and End of Relational Box (RELATIONAL BOX)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"></td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">0.4</td><td rowspan="5">1.2</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">(   )</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">0</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		

[Function]

Expresses the start and the end of the relational box. For the operation inside the relational box, refer to the descriptions for corresponding items for the relational box.

[Precautions]

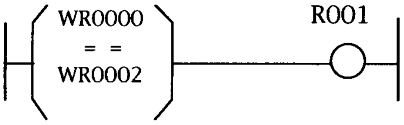
<table>
<tr><td colspan="2">Name</td><td colspan="8">= Relational Box (= RELATIONAL BOX)</td></tr>
<tr><td>Ladder format</td><td>Command format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"></td><td rowspan="3">LD
(S1==S2)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper values: for Word
Lower values: for Double word</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">1.0
~
17.3</td><td rowspan="3">4.6
~
20.8</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td rowspan="2"></td><td rowspan="2">AND
(S1==S2)</td><td colspan="5">No. of steps</td></tr>
<tr><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">30.2
~
54.3</td><td rowspan="3">50.2
~
86.2</td></tr>
<tr><td rowspan="2"></td><td rowspan="2">OR
(S1==S2)</td><td colspan="3">Word</td><td colspan="2">*1) 5 ~ 6</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">*2) 5 ~ 8</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
S1	Relational No.1					○	○	○	○	○	○	○	○	
S2	Relational No.2					○	○	○	○	○	○	○	○	

[Function]

- Absolute numbers of S1 and S2 are compared.
  Relational box is ON for S1 = S2.
  Relational box is OFF for S1 < > S2.
- Expression of S1 and S2 in words: 0 ~ 65 535 (Decimal), H0000 ~ HFFFF (Hexadecimal).
  Expression of S1 and S2 in double words: 0 ~ 4 294 967 295 (Decimal), H00000000 ~ HFFFFFFFF (Hexadecimal).

[Example program]



```
LD     (WR0000  = =  WR0002)
OUT    R001
```

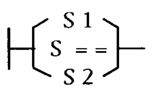
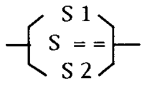
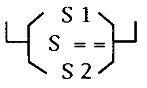
[Explanation]

R001 is ON for WR0000 = WR0002

[Precautions]

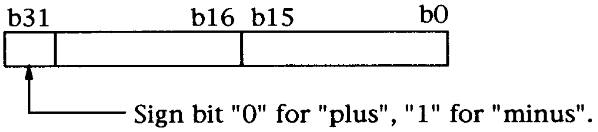
*1	
LD (S1□S2)	5 Steps
AND (S1□ S2)	5 Steps
OR (S1□ S2)	6 Steps

*2		LD, AND (S1□ S2)	OR (S1□ S2)
I/O	I/O	5 Steps	6 Steps
I/O	Constant	6 Steps	7 Steps
Constant	I/O	6 Steps	7 Steps
Constant	Constant	7 Steps	8 Steps

Name	Signed = Relational Box (SIGNED = RELATIONAL BOX)								
Ladder format	Command format	Condition code					Processing time (μ s)		Remarks
		R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
		DER	ERR	SD	V	C			
	LD (S1 S==S2)	●	●	●	●	●	30.2 ~ 54.3	50.2 ~ 86.2	
	AND (S1 S==S2)	No. of steps							
		Conditions			Step				
	OR (S1 S==S2)	Double word			*1) 5 ~ 8				

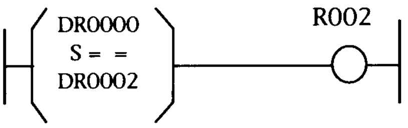
Usable I/O		Bit				Word				Double word				
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM	Constant	Others
S1	Relational No.1									○	○	○	○	
S2	Relational No.2									○	○	○	○	

[Function]



- S1 and S2 are compared as signed numbers.
- S1 and S2 are expressed by "double word".
- The highest order bit (MSB) is the sign bit.
  "0" for "plus", and "1" for "minus".
  Relational box is ON for S1 = S2.
  Relational box is OFF for S1 < > S2.
- Expression of S1 and S2 :
  -2 147 483 648 ~ +2 147 483 647 (Decimal),
  H80000000 ~ H7FFFFFFF (Hexadecimal).

[Example program]



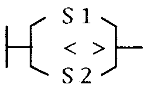
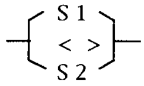
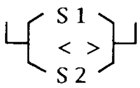
```
LD     (DR0000   S = = DR0002)
OUT    R002
```

[Explanation]

R002 is ON for DR0000 = DR0002. (Compared as signed numbers)

[Precautions]

*1		LD, AND (S1□ S2)	OR (S1□ S2)
I/O	I/O	5 Steps	6 Steps
I/O	Constant	6 Steps	7 Steps
Constant	I/O	6 Steps	7 Steps
Constant	Constant	7 Steps	8 Steps

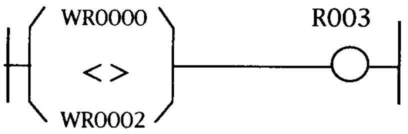
<table>
<tr><td colspan="2">Name</td><td colspan="8">< > Relational Box (< > RELATIONAL BOX)</td></tr>
<tr><td>Ladder format</td><td>Command format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"></td><td rowspan="3">LD
(S1 < > S2)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper values: for Word
Lower values: for Double word</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">1.0
~
17.3</td><td rowspan="3">4.6
~
20.8</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td rowspan="2"></td><td rowspan="2">AND
(S1 < > S2)</td><td colspan="5">No. of steps</td></tr>
<tr><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">32.6
~
49.3</td><td rowspan="3">53.8
~
78.7</td></tr>
<tr><td rowspan="2"></td><td rowspan="2">OR
(S1 < > S2)</td><td colspan="3">Word</td><td colspan="2">*1) 5 ~ 6</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">*2) 5 ~ 8</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>S1</td><td>Relational No.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></tr>
<tr><td>S2</td><td>Relational No.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></tr>
<tr><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]

- Absolute numbers of S1 and S2 are compared.
  Relational box is OFF for S1 = S2.
  Relational box is ON for S1 < > S2.
- Expression of S1 and S2 in words: 0 ~ 65 535 (Decimal), H0000 ~ HFFFF (Hexadecimal).
  Expression of S1 and S2 in double words: 0 ~ 4 294 967 295 (Decimal), H00000000 ~ HFFFFFFFF (Hexadecimal).

[Example program]



```
LD     (WR0000 <> WR0002)
OUT    R003
```

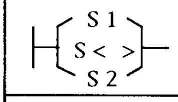
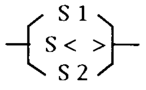
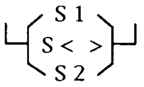
[Explanation]

R003 is ON for WR0000 < > WR0002

[Precautions]

*1	
LD (S1□S2)	5 Steps
AND (S1□ S2)	5 Steps
OR (S1□ S2)	6 Steps

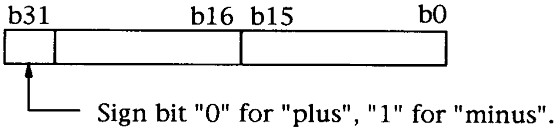
*2	LD, AND (S1□ S2)	OR (S1□ S2)
I/O I/O	5 Steps	6 Steps
I/O Constant	6 Steps	7 Steps
Constant I/O	6 Steps	7 Steps
Constant Constant	7 Steps	8 Steps

<table>
<tr><td colspan="2">Name</td><td colspan="8">Signed < > Relational Box (SIGNED < > RELATIONAL BOX)</td></tr>
<tr><td>Ladder format</td><td>Command format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"></td><td rowspan="3">LD
(S1 S< >S2)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">32.6
~
49.3</td><td rowspan="5">53.8
~
78.7</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td rowspan="2"></td><td rowspan="2">AND
(S1 S< >S2)</td><td colspan="5">No. of steps</td></tr>
<tr><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td></td><td>OR
(S1 S< >S2)</td><td colspan="3">Double word</td><td colspan="2">*1) 5 ~ 8</td></tr>
</table>

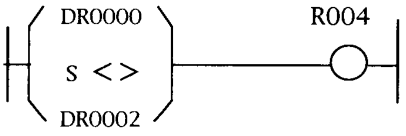
<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>S1</td><td>Relational No.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></tr>
<tr><td>S2</td><td>Relational No.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></tr>
<tr><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]

- S1 and S2 are compared as signed numbers.
- S1 and S2 are expressed by "double word".
  The highest order bit (MSB) is the sign bit.
  "0" for "plus", and "1" for "minus".
  Relational box is OFF for S1 = S2.
  Relational box is ON for S1 < > S2.
- Expression of S1 and S2 :
  -2 147 483 648 ~ +2 147 483 647 (Decimal),
  H80000000 ~ H7FFFFFFF (Hexadecimal).



[Example program]



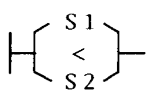
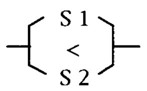
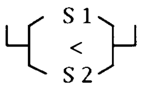
```
LD     (DR0000  S  <>  DR0002)
OUT    R004
```

[Explanation]

R004 is ON for DR0000 < > DR0002. (Compared as signed numbers)

[Precautions]

*1	LD, AND (S1□ S2)	OR (S1□ S2)
I/O I/O	5 Steps	6 Steps
I/O Constant	6 Steps	7 Steps
Constant I/O	6 Steps	7 Steps
Constant Constant	7 Steps	8 Steps

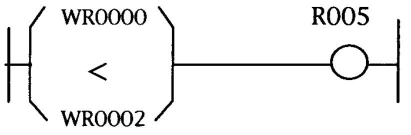
<table>
<tr><td colspan="2">Name</td><td colspan="8">< Relational Box (< RELATIONAL BOX)</td></tr>
<tr><td>Ladder format</td><td>Command format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"></td><td rowspan="3">LD
(S1 < S2)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper values: for Word
Lower values: for Double word</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">1.0
~
17.3</td><td rowspan="3">4.6
~
20.8</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td rowspan="2"></td><td rowspan="2">AND
(S1 < S2)</td><td colspan="5">No. of steps</td></tr>
<tr><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">37.1
~
48.2</td><td rowspan="3">60.5
~
77.1</td></tr>
<tr><td rowspan="2"></td><td rowspan="2">OR
(S1 < S2)</td><td colspan="3">Word</td><td colspan="2">*1) 5 ~ 6</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">*2) 5 ~ 8</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>S1</td><td>Relational No.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></tr>
<tr><td>S2</td><td>Relational No.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></tr>
<tr><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]

- Absolute numbers of S1 and S2 are compared.
  Relational box is ON for S1 < S2.
  Relational box is OFF for S1 > = S2.
- Expression of S1 and S2 in words: 0 ~ 65 535 (Decimal), H0000 ~ HFFFF (Hexadecimal).
  Expression of S1 and S2 in double words: 0 ~ 4 294 967 295 (Decimal), H00000000 ~ HFFFFFFFF (Hexadecimal).

[Example program]



LD (WR0000 < WR0002)
OUT R005

[Explanation]

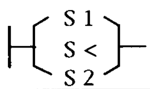
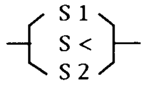
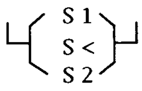
R005 is ON for WR0000 < WR0002

[Precautions]

*1	
LD (S1□S2)	5 Steps
AND (S1□ S2)	5 Steps
OR (S1□ S2)	6 Steps

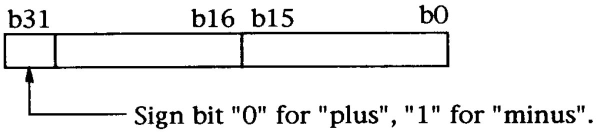
*2	LD, AND (S1□ S2)	OR (S1□ S2)
I/O I/O	5 Steps	6 Steps
I/O Constant	6 Steps	7 Steps
Constant I/O	6 Steps	7 Steps
Constant Constant	7 Steps	8 Steps

Name	Signed < Relational Box (SIGNED < RELATIONAL BOX)

Ladder format	Command format	Condition code					Processing time (μ s)		Remarks
		R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
	LD (S1 S < S2)	DER	ERR	SD	V	C	29.6 ~ 56.2	49.4 ~ 89.2	
		●	●	●	●	●			
	AND (S1 S < S2)	No. of steps							
		Conditions			Step				
	OR (S1 S < S2)	Double word			*1) 5 ~ 8				

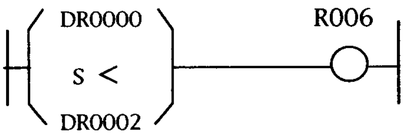
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
S1	Relational No.1									○	○	○	○	
S2	Relational No.2									○	○	○	○	

[Function]



- S1 and S2 are compared as signed numbers.
- S1 and S2 are expressed by "double word".
  The highest order bit (MSB) is the sign bit.
  "0" for "plus", and "1" for "minus".
  Relational box is ON for S1 < S2.
  Relational box is OFF for S1 > = S2.
- Expression of S1 and S2 :
  -2 147 483 648 ~ +2 147 483 647 (Decimal),
  H80000000 ~ H7FFFFFFF (Hexadecimal).

[Example program]



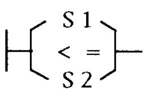
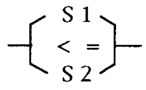
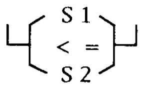
```
LD    (DR0000  S  < DR0002)
OUT   R006
```

[Explanation]

R006 is ON for DR0000 < DR0002. (Compared as signed numbers)

[Precautions]

*1	LD, AND (S1□ S2)	OR (S1□ S2)
I/O I/O	5 Steps	6 Steps
I/O Constant	6 Steps	7 Steps
Constant I/O	6 Steps	7 Steps
Constant Constant	7 Steps	8 Steps

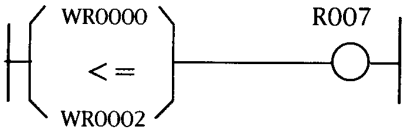
<table>
<tr><td colspan="2">Name</td><td colspan="8">< = Relational Box (< = RELATIONAL BOX)</td></tr>
<tr><td>Ladder format</td><td>Command format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"></td><td rowspan="3">LD
(S1 < = S2)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper values: for Word
Lower values: for Double word</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">1.0
~
17.3</td><td rowspan="3">4.6
~
20.8</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td rowspan="2"></td><td rowspan="2">AND
(S1 < = S2)</td><td colspan="5">No. of steps</td></tr>
<tr><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">38.4
~
49.2</td><td rowspan="3">62.4
~
78.6</td></tr>
<tr><td rowspan="2"></td><td rowspan="2">OR
(S1 < = S2)</td><td colspan="3">Word</td><td colspan="2">*1) 5 ~ 6</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">*2) 5 ~ 8</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R,
L,
M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR,
WL,
WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR,
DL,
DM</td></tr>
<tr><td>S1</td><td>Relational No.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></tr>
<tr><td>S2</td><td>Relational No.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></tr>
<tr><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]

- Absolute numbers of S1 and S2 are compared.
  Relational box is ON for S1 < = S2.
  Relational box is OFF for S1 > S2.
- Expression of S1 and S2 in words: 0 ~ 65 535 (Decimal), H0000 ~ HFFFF (Hexadecimal).
  Expression of S1 and S2 in double words: 0 ~ 4 294 967 295 (Decimal), H00000000 ~ HFFFFFFFF (Hexadecimal).

[Example program]



```
LD     (WR0000  < =  WR0002)
OUT    R007
```

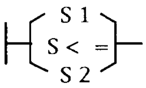
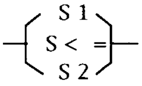
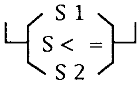
[Explanation]

R007 is ON for WR0000 < = WR0002

[Precautions]

*1	
LD (S1□S2)	5 Steps
AND (S1□ S2)	5 Steps
OR (S1□ S2)	6 Steps

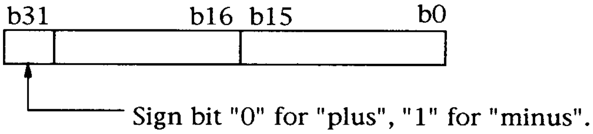
*2	LD, AND (S1□ S2)	OR (S1□ S2)
I/O I/O	5 Steps	6 Steps
I/O Constant	6 Steps	7 Steps
Constant I/O	6 Steps	7 Steps
Constant Constant	7 Steps	8 Steps

<table>
<tr><td>Name</td><td colspan="9">Signed < = Relational Box (SIGNED < = RELATIONAL BOX)</td></tr>
<tr><td>Ladder format</td><td>Command format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3"></td><td rowspan="3">LD
(S1 S< =S2)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">30.6
~
57.6</td><td rowspan="5">50.9
~
91.1</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td rowspan="2"></td><td rowspan="2">AND
(S1 S< =S2)</td><td colspan="5">No. of steps</td></tr>
<tr><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td></td><td>OR
(S1 S< =S2)</td><td colspan="3">Double word</td><td colspan="2">*1) 5 ~ 8</td></tr>
</table>

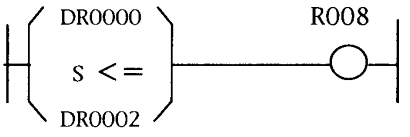
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
S1	Relational No.1									○	○	○	○	
S2	Relational No.2									○	○	○	○	

[Function]

- S1 and S2 are compared as signed numbers.
- S1 and S2 are expressed by "double word".
  The highest order bit (MSB) is the sign bit.
  "0" for "plus", and "1" for "minus".
  Relational box is ON for S1 < = S2.
  Relational box is OFF for S1 > S2.
- Expression of S1 and S2 :
  -2 147 483 648 ~ +2 147 483 647 (Decimal),
  H80000000 ~ H7FFFFFFF (Hexadecimal).



[Example program]



```
LD     (DR0000 S  <= DR0002)
OUT    R008
```

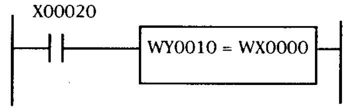
[Explanation]

R008 is ON for DR0000 < = DR0002. (Compared as signed numbers)

[Precautions]

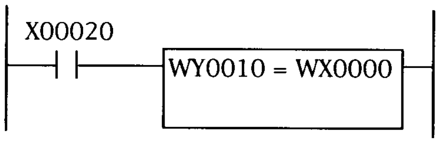
*1		LD, AND (S1□ S2)	OR (S1□ S2)
I/O	I/O	5 Steps	6 Steps
I/O	Constant	6 Steps	7 Steps
Constant	I/O	6 Steps	7 Steps
Constant	Constant	7 Steps	8 Steps

Name	Start and End of Processing Box (PROCESSING BOX)

Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
	DER	ERR	SD	V	C	0.6	1.8	
	●	●	●	●	●			
Command format	No. of steps							
	Conditions				Step			
[ ]	—				3			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		

[Function]



LD X00020
[
WY0010 = WX0000
]

Expresses the start and the end of the processing box.
In the above example, the operation expressed in the processing box will be executed when the input X00020 is ON.

[Precautions]

3.3 Details of Commands

<table>
<tr><td>Name</td><td colspan="8">ASSIGNMENT STATEMENT</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">d = s</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">Refer to the table below</td><td rowspan="5">Refer to the table below</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">d = s</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">Refer the table below</td><td colspan="2"></td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination		○	○		○	○	○	○		○	○		
s	Source	○	○	○		○	○	○	○	○	○	○	○*	
( )	Index value					○	○	○						

[Function]

- s is assigned to d.
- Array variables can be used for d and s.
  External I/O (X, WX, DX, Y, WX, DX) can not be used for I/O of Array variables.

Item / Condition		No. of steps	Processing time(μ s)											
			H-252						H-250					
			Bit		Word		Double word		Bit		Word		Double word	
d	s	( ) for DW	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
I/O	I/O	3 (4)	0.8	1.1	0.9	9.1	1.6	18.1	3.8	4.1	3.8	11.9	5.0	21.2
I/O	Array	4	78.9	126.2	49.8	130.6	61.5	139.4	123.1	193.8	79.5	200.3	97.0	213.5
Array	I/O	4 (5)	67.1	110.6	46.8	127.5	56.9	134.8	105.5	170.5	75.0	195.8	90.1	206.7
Array	Array	5	109.9	201.4	71.5	233.0	87.4	242.3	169.4	306.3	112.0	353.5	135.8	367.4

* Constant in case "d" is word.
0 ~ 65 535 or -32 768 ~ +32 767 (Decimal)
H0000 ~ HFFFF or H8000 ~ H7FFF (Hexadecimal)
In case "d" is double word.
0 ~ 4 294 967 295 or -2 147 483 648 ~ +2 147 483 647 (Decimal)
H00000000 ~ HFFFFFFFF or H80000000 ~ H7FFFFFFF (Hexadecimal)

[Precautions]

- Combinations of d and s

d	s
Bit	Bit
Word	Word
Doubleword	Doubleword

- In case array variables are used, DER = "1" when the assignment exceeds maximum permissible I/O number.

[Example program]

①

② Use array variables to destination.

③ Use array variables to source.

④ Use array variables to both destination and source.

I/O assignment

	0	1	2	3	4	5	6	7	8
AVR	CPU	X 32	Y 32	X 32	Y 32	4/4W	X 8 W	Not used	Not used	Not used

[Explanations]

① Value of WX0000 is assigned to WR0000 at the rising edge of X00000.

② Value of WX0000 is assigned to WR No. expressed by WR0000 + WM000, at the rising edge of X00001

1) In the case of WM000 = H0010, the above operation becomes WR0010 = WX0000.

③ At the rising edge of input X00002, word number that is expressed by WR0000 + WM001 in I/O assignment is assigned to WR0000.

1) In case WM001 = H0010, it means WR0000 = WR0010.

④ At the rising edge of input X00003, the value of I/O expressed by WR0000 + WM001 is assigned to I/O expressed by WR0000 + WM000.

Example) In case WM000 = H0010 and WM001 = H0015, the result is WR0010 = WR0015.

<table>
<tr><td>Name</td><td colspan="8">BINARY ADDITION</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">d = s1 + s2</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">Upper values: for word.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">42.4
~
42.8</td><td rowspan="3">68.5
~
69.1</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>↕</td><td>↕</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">d = s1 + s2</td><td colspan="4">Conditions</td><td>Step</td><td rowspan="2">52.8
~
53.2</td><td rowspan="2">83.9
~
84.6</td><td rowspan="2">Lower values: for double word.</td></tr>
<tr><td colspan="4">—</td><td>3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination						○	○	○		○	○		
s1	Augend					○	○	○	○	○	○	○	○	
s2	Addend					○	○	○	○	○	○	○	○	

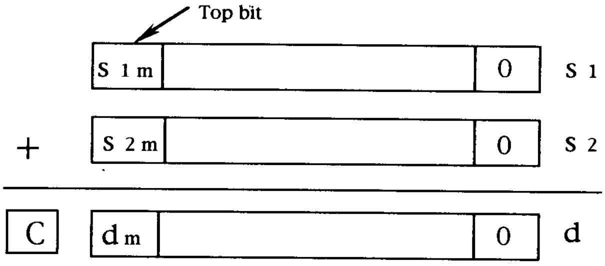
[Function]

- Operates binary addition of s1 and s2 and assigns the sum to "d" in binary code data.
  Example) WR0002 = WR0000 + WR0001
- C flag (Carry: R7F0) will be reset to zero in case the operational result is within the value shown below. Outside this range "1" will be set.
  In the case of word: H0000 ~ HFFFF (Hexadecimal) or 0 ~ 65 535 (Decimal).
  In the case of double word: H00000000 ~ HFFFFFFFF (Hexadecimal) or 0 ~ 4 294 967 295 (Decimal).

$$C = S_{1m} \bullet S_{2m} + S_{1m} \bullet \overline{d_m} + S_{2m} \bullet \overline{d_m}$$

- V flag (Overflow : R7F1) will be set to "1" when the operational result is a meaningless signed binary data, and will be reset to "0" when the result has a meaning. (Refer to the table below).

s1	s2	d	V
Plus	Plus	Plus	0
Plus	Plus	Minus	1
Plus	Minus	Plus/ Minus	0
Minus	Plus	Plus/ Minus	0
Minus	Minus	Plus	1
Minus	Minus	Minus	0



$$V = S_{1m} \bullet S_{2m} \bullet \overline{d_m} + \overline{S_{1m}} \bullet \overline{S_{2m}} \bullet d_m$$

[Precautions]

- Combinations of d, s1 and s2

d	s1	s2
Word	Word	Word
Doubleword	Doubleword	Doubleword

<table>
<tr><td>Name</td><td colspan="9">BCD (BINARY CODE DECIMAL) ADDITION</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">d = s1 B + s2</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper values: for word.
Lower values: for doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">53.3</td><td rowspan="3">84.8</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>↕</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">d = s1 B + s2</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">84.5</td><td rowspan="3">131.4</td></tr>
<tr><td colspan="3">Word</td><td colspan="2">4</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">6</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>Destination</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>S1</td><td>Augend</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>Addend</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]

- Operates BCD addition of s1 and s2 and assigns the sum to "d" in BCD data.
  Example) WL0002 = WL0000 B + WL0001
- C flag (Carry R7F0) will be set to "1", when the result has a carry, and to "0" when the result has no carry.
- DER (Data error: R7F4) will be set to "1" and the operation will not be executed when the value of S1 or S2 is invalid as BCD data. In this instance C will be hold with the last result and will not be outputted to "d". When the said value is valid DER will be reset to "0", and the result will be outputted to "d".
- S1, S2 for word: 0000 ~ 9999 (BCD)
- S1, S2 for double word: 00000000 ~ 99999999 (BCD)

[Precautions]

- Combinations of d, s1 and s2

d	s1	s2
Word	Word	Word
Doubleword	Doubleword	Doubleword

<table>
<tr><td>Name</td><td colspan="8">BINARY SUBTRACTION</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">$d = s_1 - s_2$</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper values: for word.
Lower values: for double word.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">40.4</td><td rowspan="3">65.5</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>↕</td><td>↕</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">$d = s_1 - s_2$</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">49.7</td><td rowspan="3">79.4</td></tr>
<tr><td colspan="3">Word</td><td colspan="2">4</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">6</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>Destination</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>s_1</td><td>Minuend</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>s_2</td><td>Subtrahend</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>

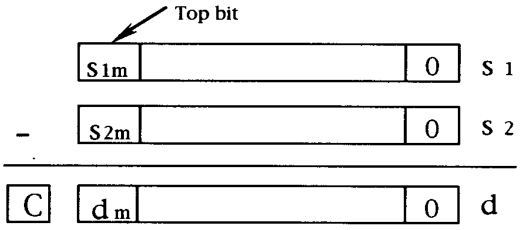
[Function]

- Subtract s2 from s1 as binary code data, and assign the difference to "d" as binary code data.
  Example) WY0010 = WX0000 - WX0001
- C flag (Carry : R7F0) will be set to "1" when the result has a borrow, and will be reset to "0", when the result has no borrow.
  For s1 < s2, "1" will be set to C
  For s1 >= s2, "0" will be reset to C

$$C = \overline{s_{1m}} \bullet s_{2m} + \overline{s_{1m}} \bullet d_m + s_{2m} \bullet d_m$$

- V flag (Overflow : R7F1) will be set to "1" when the operational result is a meaningless signed binary data, and will be reset to "0" when the result has a meaning. (Refer to the table below).

s1	s2	d	V
Plus	Plus	Plus/ Minus	0
Minus	Minus	Plus/ Minus	0
Plus	Minus	Plus	0
Plus	Minus	Minus	1
Minus	Plus	Plus	1
Minus	Plus	Minus	0



$$V = \overline{s_{1m}} \bullet s_{2m} \bullet d_m + s_{1m} \bullet \overline{s_{2m}} \bullet \overline{d_m}$$

[Precautions]

- Combinations of d, s_1 and s_2

d	s_1	s_2
Word	Word	Word
Doubleword	Doubleword	Doubleword

<table>
<tr><td>Name</td><td colspan="8">BCD (BINARY CODE DECIMAL) SUBTRACTION</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">d = s1 B - s2</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="8">Upper values: for word.
Lower values: for doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">53.7</td><td rowspan="3">85.4</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>↕</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">d = s1 B - s2</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">81.3</td><td rowspan="3">126.7</td></tr>
<tr><td colspan="3">Word</td><td colspan="2">4</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">6</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination						○	○	○		○	○		
S1	Minuend					○	○	○	○	○	○	○	○	
S2	Subtrahend					○	○	○	○	○	○	○	○	

[Function]

- Subtract s2 from s1 as BCD data, and assign the difference to "d" as BCD data.
  Example) WR0003 = WR0004 B - WR0005
- C flag (Carry : R7F0) will be set to "1" when the result has a borrow, and will be reset to "0", when the result has no borrow.
- DER (Data error: R7F4) will be set to "1" and the operation will not be executed when the value of S1 or S2 is invalid as BCD data. In this instance C will be hold with the last result and will not be outputted to "d". When the said value is valid DER will be reset to "0", and the result will be outputted to "d".

[Precautions]

- Combinations of d, s1 and s2

d	S1	S2
Word	Word	Word
Doubleword	Doubleword	Doubleword

Name	BINARY MULTIPLICATION

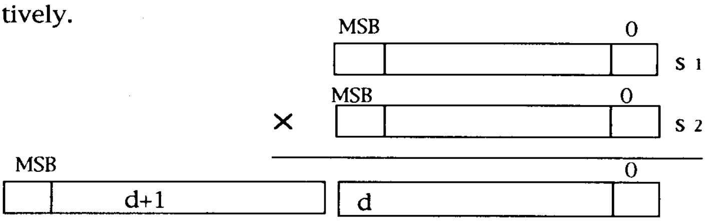
Ladder format	Condition code					Processing time (μs)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	Upper values: for word.
d = S1 × S2	DER	ERR	SD	V	C	70.2 ~ 99.0	110 ~ 153.1	
	↕	●	●	●	●			

Command format	No. of steps		Processing time (μs)		Remarks
	Conditions	Step	H-252	H-250	
d = S1 × S2	Word	4	162.2 ~ 1086.5	247.6 ~ 1629.8	Lower values: for doubleword.
	Double word	6			

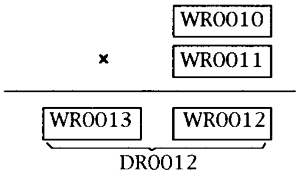
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination						○	○	○		○	○		
S1	Multiplicand					○	○	○	○	○	○	○	○	
S2	Multiplier					○	○	○	○	○	○	○	○	

[Function]

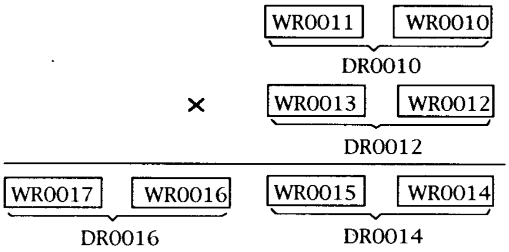
- Operates binary multiplication of s1 and s2 and assigns the product to "d + 1" (upper digits) and "d" (lower digits) in binary data.
- DER (Data error : R7F4) will be set to "1" and only lower word will be assigned when "d +1" exceeds usable I/O area (WR43FF (WR3FF for H-250), WL3FF, WL13FF, WM13FF). When the result does not exceed the said I/O, DER will be reset to "0" and lower word and upper word will be assigned to "d" and "d + 1" respectively.



Example)
WR0012 = WR0010 × WR0011



Example)
DR0014 = DR0010 × DR0012



[Precautions]

- Combinations of d, S1 and S2

d	S1	S2
Word	Word	Word
Doubleword	Doubleword	Doubleword

As the result of operation is assigned to "d" and "d + 1" without exception, adequate care should be taken to avoid using the word or double word in "d + 1" as I/O of other purpose.

In H-252, the internal output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when d+1 exceeds WR3FF

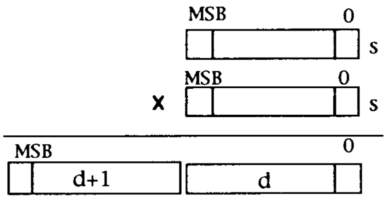
Name	BCD MULTIPLICATION

Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
d = S1 B × S2	DER	ERR	SD	V	C	209.4 ~ 344.7	318.2 ~ 520.5	Upper values: for word.
	↕	●	●	●	●			
Command format	No. of steps							
d = S1 B × S2	Conditions				Step	434.7 ~ 1889.0	655.1 ~ 2829.8	Lower values: for doubleword.
	Word				4			
	Double word				6			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination						○	○	○		○	○		
S1	Multiplicand					○	○	○	○	○	○	○	○	
S2	Multiplier					○	○	○	○	○	○	○	○	

[Function]

- Operates BCD multiplication of s1 and s2 and assigns the product to "d + 1" (upper digits) and "d" (lower digits) in BCD data.
- DER (Data error : R7E4) will be set to "1" and the operation will not be executed when the value of S1 or S2 is invalid BCD data. DER will also be set to "1" but only lower word will be assigned to "d" when "d +1" exceeds usable I/O area (WR43FF (WR3FF for H-250), WL3FF, WL13FF, WM13FF). When S1 and S2 are valid as BCD data and "d + 1" is within usable I/O, DER will be reset to "0" and the result will be assigned to "d + 1" and "d".

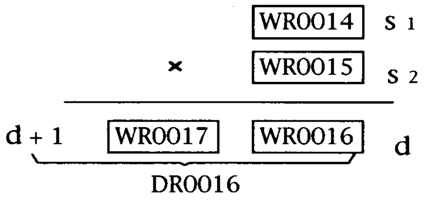


[Precautions]

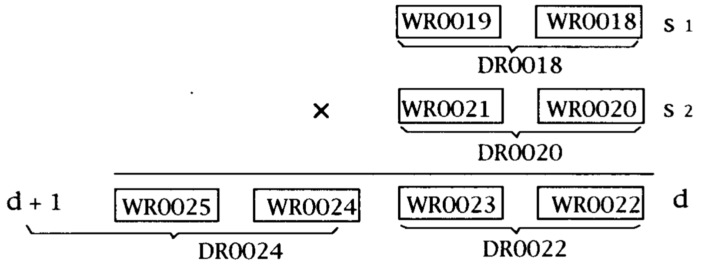
- Combinations of d, S1 and S2

d	S1	S2
Word	Word	Word
Doubleword	Doubleword	Doubleword

Example)
WR0016 = WR0014 B x WR0015



Example)
DR22 = DR18B x DR20



As the result of operation is assigned to "d" and "d + 1" without exception, adequate care should be taken to avoid using the word or double word in "d + 1" as I/O of other purpose.

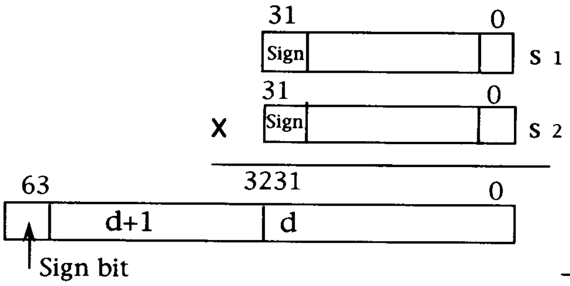
In H-252, the internal output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when d+1 exceeds WR3FF

<table>
<tr><td>Name</td><td colspan="8">SIGNED BINARY MULTIPLICATION</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">d = S1 S × S2</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">172.4
~
1128.8</td><td rowspan="5">262.9
~
1693.1</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">d = S1 S × S2</td><td colspan="4">Conditions</td><td>Step</td></tr>
<tr><td colspan="4">Double word</td><td>6</td></tr>
</table>

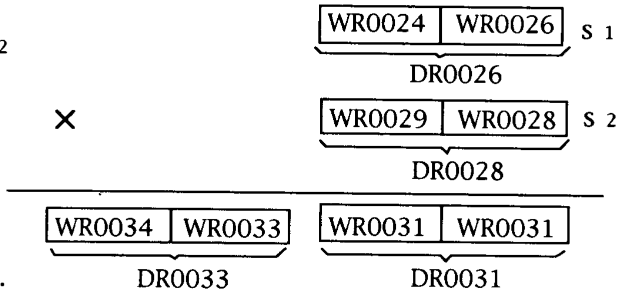
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination										○	○		
S1	Multiplicand									○	○	○	○	
S2	Multiplier									○	○	○	○	

[Function]

- Operates binary multiplication of s1 and s2 treated as signed data and assigns the product to "d + 1" (upper digits) and "d" (lower digits) in binary data.
- DER (Data error : R7E4) will be set to "1" and only lower word will be assigned to "d" when "d +1" exceeds usable I/O area (WR43FF (WR3FF for H-250), WL3FF, WL13FF, WM13FF). When the result does not exceed the said I/O, DER will be reset to "0" and lower word and upper word will be assigned to "d" and "d + 1" respectively.



Example)
DR0031 = DR0026 s× DR0028



Sign of the result is allocated on the highest bit.
s1, s2 : -2 147 483 648 ~ + 2 147 483 647 (Decimal)
H80000000 ~ H7FFFFFFF (Hexadecimal)

[Precautions]

- The result of operation is assigned to "d" and "d + 1" without exception, therefore adequate care should be taken to avoid using the word or double word in "d + 1" as I/O of other purposes.

In H-252, the internal output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when d+1 exceeds WR3FF

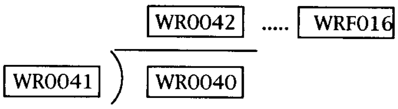
<table>
<tr><td>Name</td><td colspan="9">BINARY DIVISION</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">$d = s_1 \ / \ s_2$</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">Upper values: for word.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">104.1</td><td rowspan="3">160.7</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">$d = s_1 \ / \ s_2$</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">1494.1
~
1686.6</td><td rowspan="3">2239.3
~
2527.2</td><td rowspan="3">Lower values: for doubleword.</td></tr>
<tr><td colspan="3">Word</td><td colspan="2">4</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">6</td></tr>
</table>

<table>
<tr><td rowspan="2" colspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>Destination</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>s_1</td><td>Dividend</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>s_2</td><td>Divisor</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>

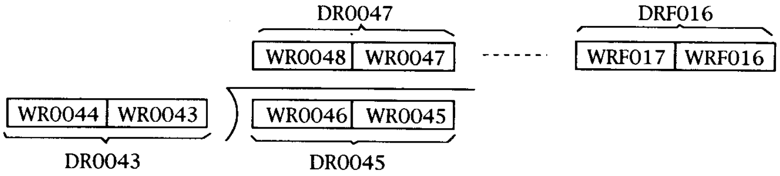
[Function]

- Operates binary division of s1 and s2 and assigns the quotient to "d" in binary code data. (Integer operation).
  Remainder of the division of s1 by s2 (s1 mod s2) is assigned to special internal output WRF016 (DRF016 for double word).
- DER (Data error : R7F4) will be set to "1" and the operation will not be executed when S2 = 0. When s2 < > 0 DER will be reset to "0" and the operation will be executed.

Example) WR0042 = WR0040 / WR0041



Example) DR0047 = DR0045 / DR0043



[Precautions]

- Combinations of d, s_1 and s_2

d	s_1	s_2
Word	Word	Word
Doubleword	Doubleword	Doubleword

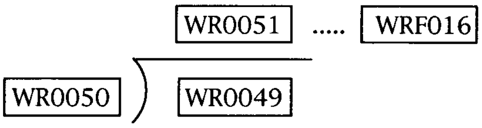
<table>
<tr><td>Name</td><td colspan="8">BCD DIVISION</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">d = s1 B / s2</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper values: for word.
Lower values: for doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">222.5
~
286.1</td><td rowspan="3">337.7
~
432.9</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">d = s1 B / s2</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">1417.7
~
2271.3</td><td rowspan="3">2125.0
~
3401.6</td></tr>
<tr><td colspan="3">Word</td><td colspan="2">4</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">6</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>Destination</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>s1</td><td>Dividend</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>Divisor</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]

- Operates BCD division of s1 and s2 and assigns the quotient to "d" in BCD code data. (Integer operation).
  Remainder of the division of s1 by s2 (s1 mod s2) is assigned to special internal output WRF016 (DRF016 for double word).
- DER (Data error : R7F4) will be set to "1" and the operation will not be executed when s1 or s2 is judged invalid as data or when s2 = 0. When they are valid and s2 < > 0, DER will be reset to "0" and the operation will be executed.

Example) WR0051 = WR0049 B / WR0050



s1, s2 : for Word : 0000 ~ 9999 (BCD)
for Double word : 00000000 ~ 99999999 (BCD)

[Precautions]

- Combinations of d, s1 and s2

d	s1	s2
Word	Word	Word
Doubleword	Doubleword	Doubleword

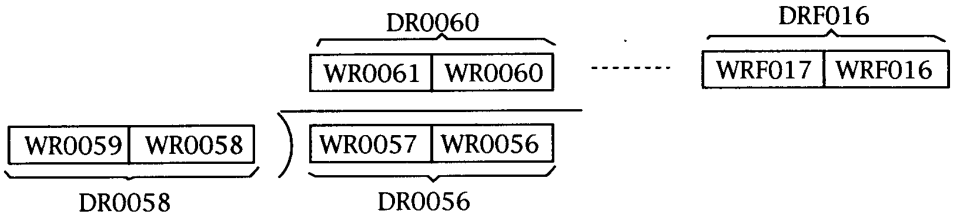
<table>
<tr><td>Name</td><td colspan="9">SIGNED BINARY DIVISION</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">d = s₁ s / s₂</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="6">1142.0
~
1731.5</td><td rowspan="6">1712.8
~
2594.3</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>↕</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">d = s₁ s / s₂</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3" rowspan="2">Double word</td><td colspan="2" rowspan="2">6</td></tr>
<tr></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination										○	○		
s1	Dividend									○	○	○	○	
s2	Divisor									○	○	○	○	

[Function]

- Operates binary division of s1 and s2 as signed binary data and assigns the quotient to "d" in binary code data. (Integer operation).
  Remainder of the division of s1 by s2 (s1 mod s2) is assigned to special internal output WRF016 as signed binary data. (Sign will be located on the highest bit).
- DER (Data error : R7F4) will be set to "1" and the operation will not be executed when s2 = 0. When s2 < > 0, DER will be reset to "0" and the operation will be executed.
- V (Overflow : R7F1) will be set to "1" when the positive quotient exceeds H7FFFFFFF (Hexadecimal). In all other cases V will be reset.
  s1, s2 : -2 147 483 648 ~ +2 147 483 647 (Decimal)
  H80000000 ~ H7FFFFFFF (Hexadecimal)

Example) DR0060 = DR0056 s / DR0058



[Precautions]

<table>
<tr><td>Name</td><td colspan="8">LOGICAL SUM (OR)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μs)</td><td>Remarks</td></tr>
<tr><td rowspan="3">d = s1 OR s2</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper:
for Bit.
Middle:
for Word..
Lower:
for
Doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="2">32.3
~33.7</td><td rowspan="2">53.1
~55.4</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td><td rowspan="2">25.9</td><td rowspan="2">43.8</td></tr>
<tr><td rowspan="3">d = s1 OR s2</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">Word</td><td colspan="2">4</td><td rowspan="2">36.7</td><td rowspan="2">60.0</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">6</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination		○	○			○	○	○		○	○		
s1	Comparand	○	○	○		○	○	○	○	○	○	○	○	
s2	Comparer	○	○	○		○	○	○	○	○	○	○	○	

[Function]

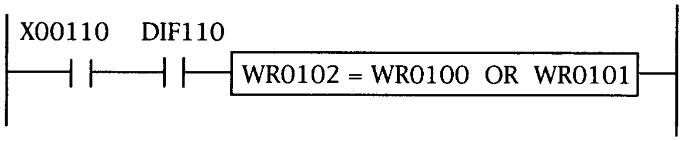
Makes logical sum (OR) of S1 and S2 assigns the result to "d".

Truth table

s 1	s 2	d
0	0	0
0	1	1
1	0	1
1	1	1

$d \leftarrow s_1 + s_2$

[Example program]



```
LD   X00110
AND  DIF110
[
WR0102 = WR0100 OR WR0101
]
```

[Explanations]

- At the rising edge of X00110, the logical sum (OR) of WR0100 and WR0101 is made and its result is set to WR0102.

WR0100 = H1234
WR0101 = H5678
WR0102 = H567C

⇨

WR0100 = 0001001000110100
WR0101 = 0101011001111000
WR0102 = 0101011001111100

[Precautions]

- Combinations of d, s1 and s2

d	s1	s2
Bit	Bit	Bit
Word	Word	Word
Doubleword	Doubleword	Doubleword

<table>
<tr><td>Name</td><td colspan="8">LOGICAL PRODUCT (AND)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">d = s1 AND s2</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper:
for Bit.
Middle:
for Word..
Lower:
for Doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="2">32.2 ~33.7</td><td rowspan="2">53.1 ~55.4</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td><td rowspan="2">25.9</td><td rowspan="2">43.8</td></tr>
<tr><td rowspan="3">d = s1 AND s2</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">Word</td><td colspan="2">4</td><td rowspan="2">36.7</td><td rowspan="2">60.0</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">6</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination		○	○			○	○	○		○	○		
s1	Comparand	○	○	○		○	○	○	○	○	○	○	○	
s2	Comparer	○	○	○		○	○	○	○	○	○	○	○	

[Function]

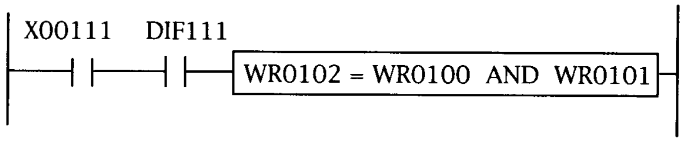
Makes logical product (AND) of s1 and s2 and assigns the result to "d".

Truth table

s 1	s 2	d
0	0	0
0	1	0
1	0	0
1	1	1

$d \leftarrow s_1 \bullet s_2$

[Example program]



```
LD   X00111
AND  DIF111
[
WR0102 = WR0100  AND  WR0101
]
```

[Explanations]

At the rising edge of X00111, the logical product (AND) of WR0100 and WR0101 is made and its result is set to WR0102.

WR0100 = H1234 ⇨ WR0100 = 0001001000110100

WR0101 = H5678 ⇨ WR0101 = 0101011001111000

WR0102 = H1230 ⇨ WR0102 = 0001001000110000

[Precautions]

- Combinations of d, s1 and s2

d	s1	s2
Bit	Bit	Bit
Word	Word	Word
Doubleword	Doubleword	Doubleword

<table>
<tr><td>Name</td><td colspan="8">LOGICAL EXCLUSIVE SUM (EXCLUSIVE - OR)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">$d = s_1$ XOR s_2</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper:
for Bit.
Middle:
for Word..
Lower:
for
Doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="2">31.0
~32.5</td><td rowspan="2">51.3
~ 53.6</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td><td rowspan="2">25.9</td><td rowspan="2">43.8</td></tr>
<tr><td rowspan="3">$d = s_1$ XOR s_2</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">Word</td><td colspan="2">4</td><td rowspan="2">36.7</td><td rowspan="2">60.0</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">6</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination		○	○			○	○	○		○	○		
s_1	Comparand	○	○	○		○	○	○	○	○	○	○	○	
s_2	Comparer	○	○	○		○	○	○	○	○	○	○	○	

[Function]

Makes logical exclusive sum (XOR) of s1 and s2 and assigns the result to "d".

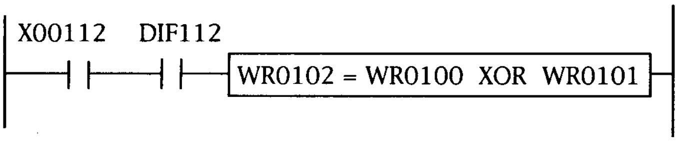
$$d \leftarrow s_1 \oplus s_2 = \overline{s_1} \bullet s_2 + s_1 \bullet \overline{s_2}$$

Truth table

$d \leftarrow s_1 \oplus s_2$

s_1	s_2	d
0	0	0
0	1	1
1	0	1
1	1	0

[Example program]



```
LD    X00112
AND   DIF112
[
WR0102 = WR0100  XOR  WR0101
]
```

[Explanations]

At the rising edge of X00112, the logical exclusive sum (XOR) of WR0100 and WR0101 is made and its result is set to WR0102.

WR0100 = H1234		WR0100 = 0001001000110100
WR0101 = H5678	⇨	WR0101 = 0101011001111000
WR0102 = H444C		WR0102 = 0100010001001100

[Precautions]

- Combinations of d, s_1 and s_2

d	s_1	s_2
Bit	Bit	Bit
Word	Word	Word
Doubleword	Doubleword	Doubleword

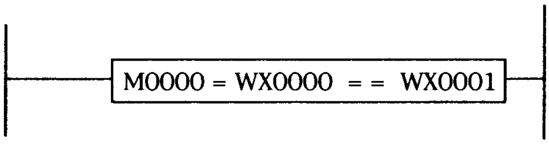
Name	= RELATIONAL EXPRESSION							
Ladder format	Condition code					Processing time (μ s)		Remarks
$d = s_1 == s_2$	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	Upper values: for word. Lower values: for doubleword.
	DER	ERR	SD	V	C	1.2 ~ 26.1	4.6 ~ 28.9	
	●	●	●	●	●			
Command format	No. of steps							
$d = s_1 == s_2$	Conditions			Step		33.5 ~ 40.8	55.1 ~ 66.0	
	S is Word			4				
	S is Doubleword			6				

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination		○	○										
s_1	Comparand					○	○	○	○	○	○	○	○	
s_2	Comparer					○	○	○	○	○	○	○	○	

[Function]

- When s1 = s2, "1" will be set to "d".
  In all other cases "d" will be reset to "0".

[Example program]



[
M0000 = WX0000 == WX0001
]

[Explanations]

- When WX0000 = WX0001, "1" will be set to M0000.

[Precautions]

- Combinations of d, s1 and s2

d	s1	s2
Bit	Word	Word
Bit	Doubleword	Doubleword

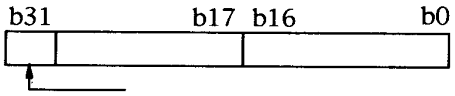
<table>
<tr><td>Name</td><td colspan="9">SIGNED = RELATIONAL EXPRESSION</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">d = s1 s== s2</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">33.5
~
40.8</td><td rowspan="5">55.1
~
66.0</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">d = s1 s== s2</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">S is Double word</td><td colspan="2">6</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>Destination</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>S1</td><td>Comparand</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>Comparer</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]

- When s1 = s2, "1" will be set to "d".
  In all other cases "d" will be reset to "0".
- S1 and S2 are binary code data. They are plus when the highest bit is "0", and are minus when the highest bit is "1"

- S1, S2 : -2 147 483 648 ~ +2 147 483 647 (Decimal)
  H80000000 ~ H7FFFFFFF (Hexadecimal)



Sign bit "0" = Plus, "1" = Minus

Example) M0000 = DR0000 s == DR0002

[Precautions]

Name	< > RELATIONAL EXPRESSIN

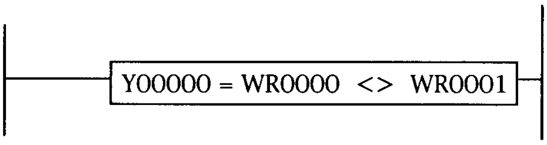
Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	Upper values: for word. Lower values: for doubleword.
d = s_1 <> s_2	DER	ERR	SD	V	C	1.2 ~ 26.1	4.6 ~ 28.9	
	●	●	●	●	●			
Command format	No. of steps							
	Conditions			Step		34.1 ~ 40.8	56.1 ~ 66.0	
d = s_1 <> s_2	S is Word			4				
	S is Doubleword			6				

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination		○	○										
s_1	Comparand					○	○	○	○	○	○	○	○	
s_2	Comparer					○	○	○	○	○	○	○	○	

[Function]

- When s1 <> s2, "1" will be set to "d".
  In all other cases "d" will be reset to "0".

[Example program]



[
Y00000 = WR0000 <> WR0001
]

[Explanations]

- When WR0000 <> WR0001, "1" is set to Y0000

[Precautions]

- Combinations of d, s_1 and s_2

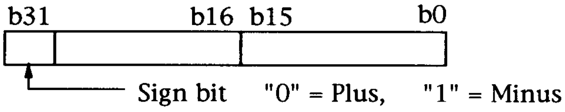
d	s_1	s_2
Bit	Word	Word
Bit	Doubleword	Doubleword

<table>
<tr><td>Name</td><td colspan="7">SIGNED < > RELATIONAL EXPRESSIN</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">d = s1 s <> s2</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="6">34.1
~
40.8</td><td rowspan="6">56.1
~
66.0</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">d = s1 s <> s2</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">S is Double word</td><td colspan="2">6</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination		○	○										
s1	Comparand									○	○	○	○	
s2	Comparer									○	○	○	○	

[Function]

- When s1 <> s2, "1" will be set to "d".
  In all other cases "d" will be reset to "0".
- s1 and s2 are binary code data. They are plus when the highest bit is "0", and are minus when the highest bit is "1"
  s1, s2 : -2 147 483 648 ~ +2 147 483 647 (Decimal)
  H80000000 ~ H7FFFFFFF (Hexadecimal)



Example) Y0000 = DR0000 s <> DR0002

[Precautions]

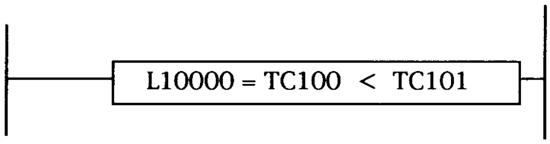
Name	< RELATIONAL EXPRESSIN							
Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
$d = s_1 < s_2$	DER	ERR	SD	V	C	1.2 ~ 26.1	4.6 ~ 28.9	Upper values: for word. Lower values: for doubleword.
	●	●	●	●	●			
Command format	No. of steps					38.2 ~ 40.1	62.1 ~ 65.0	
	Conditions				Step			
$d = s_1 < s_2$	S is Word				4			
	S is Doubleword				6			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination		○	○										
s_1	Comparand					○	○	○	○	○	○	○	○	
s_2	Comparer					○	○	○	○	○	○	○	○	

[Function]

When s1 < s2, "1" will be set to "d".
In all other cases "d" will be reset to "0".

[Example program]



```
[
L10000 = TC100 < TC101
]
```

[Explanations]

When TC100 < TC101, "1" is set to L10000
(TCn is an elapsed value of a timer No. "n" or a counter No. "n").

[Precautions]

- Combinations of d, s_1 and s_2

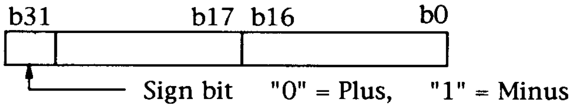
d	s_1	s_2
Bit	Word	Word
Bit	Doubleword	Doubleword

<table>
<tr><td>Name</td><td colspan="8">SIGNED < RELATIONAL EXPRESSIN</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">d = s_1 s < s_2</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">34.3
~
46.2</td><td rowspan="5">56.4
~
74.2</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">d = s_1 s < s_2</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">S is Double word</td><td colspan="2">6</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination		○	○										
s_1	Comparand									○	○	○	○	
s_2	Comparer									○	○	○	○	

[Function]

- When s1 < s2, "1" will be set to "d".
  In all other cases "d" will be reset to "0".
- s1 and s2 are binary code data. They are plus when the highest bit is "0", and are minus when the highest bit is "1"
  s1, s2 : -2 147 483 648 ~ +2 147 483 647 (Decimal)
  H80000000 ~ H7FFFFFFF (Hexadecimal)



Example) R100 = DM000 s < DM002

[Precautions]

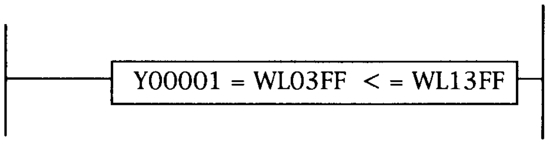
<table>
<tr><td>Name</td><td colspan="8">< = RELATIONAL EXPRESSIN</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">$d = s_1$ <= s_2</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper values: for word.
Lower values: for doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">1.2
~
26.1</td><td rowspan="3">4.6
~
28.9</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">$d = s_1$ <= s_2</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">37.9
~
39.5</td><td rowspan="3">61.8
~
64.1</td></tr>
<tr><td colspan="3">S is Word</td><td colspan="2">4</td></tr>
<tr><td colspan="3">S is Doubleword</td><td colspan="2">6</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination		○	○										
s_1	Comparand					○	○	○	○	○	○	○	○	
s_2	Comparer					○	○	○	○	○	○	○	○	

[Function]

- When s1 <= s2, "1" will be set to "d".
  In all other cases "d" will be reset to "0".

[Example program]



[
Y00001 = WL03FF < = WL13FF
]

[Explanations]

- When WL03FF <= WL13FF, "1" is set to Y0001

[Precautions]

- Combinations of d, s_1 and s_2

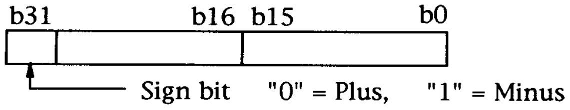
d	s_1	s_2
Bit	Word	Word
Bit	Doubleword	Doubleword

<table>
<tr><td>Name</td><td colspan="8">SIGNED < = RELATIONAL EXPRESSIN</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">$d = s_1 \quad s <= s_2$</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="6">42.8
~
43.7</td><td rowspan="6">69.1
~
70.4</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">$d = s_1 \quad s <= s_2$</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3" rowspan="2">S is Double word</td><td colspan="2" rowspan="2">6</td></tr>
<tr></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Destination		○	○										
s_1	Comparand									○	○	○	○	
s_2	Comparer									○	○	○	○	

[Function]

- When s1 <= s2, "1" will be set to "d".
  In all other cases "d" will be reset to "0".
- s1 and s2 are binary code data. They are plus when the highest bit is "0", and are minus when the highest bit is "1"
  s1, s2 : -2 147 483 648 ~ +2 147 483 647 (Decimal)
  H80000000 ~ H7FFFFFFF (Hexadecimal)



Example) Y00001 = DL03FE s < = DL13FE

[Precautions]

3.3 Details of Commands

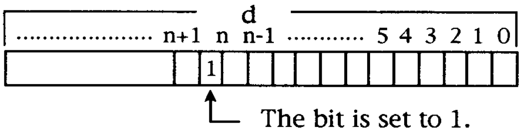
(1) Basic Commands
(2) Arithmetic Commands
(3) Application Commands
(4) Control Commands
(5) Transfer Commands for Sophisticated Function Module
(6) FUN Commands

<table>
<tr><td>Name</td><td colspan="9">BIT SET</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">BSET(d,n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6">Upper values: for word.
Lower values: for doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">31.5
~
83.8</td><td rowspan="3">52.2
~
130.4</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">BSET (d,n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">32.7
~
86.1</td><td rowspan="2">54.0
~
133.8</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	I/O with bit reset						○	○	○		○	○		
n	Bit position to be reset					○	○	○	○				○	The constant is specified in decimal.

[Function]

- The "n"th bit of the I/O (word or double word) specified by d is set to 1.
- The contents of the other bits are not changed.



When d indicates a word:

- The bit position is specified by the contents (0 to 15) of the low-order 4 bits (b3 to b0) of n(WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 15 can be specified as n (constant) (Decimal)

When d indicates a double word:

- The bit position is specified by the contents (0 to 31) of the low-order 5 bits (b4 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 31 can be specified as n (constant). (Decimal)

[Precautions]

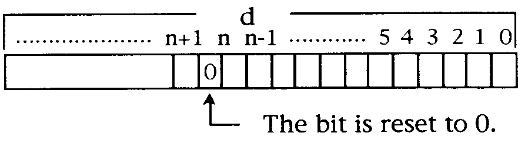
<table>
<tr><td>Name</td><td colspan="9">BIT RESET</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">BRES(d,n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6">Upper values: for word.

Lower values: for doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">34.3
~
86.7</td><td rowspan="3">56.4
~
130.4</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">BRES (d,n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">35.6
~
89.3</td><td rowspan="2">58.4
~
138.6</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>I/O with bit reset</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>n</td><td>Bit position to be reset</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>The constant is specified in decimal.</td></tr>
<tr><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]

- The "n"th bit of the I/O (word or double word) specified by d is set to 0.
- The contents of the other bits are not changed.



The bit is reset to 0.

When d indicates a word:

- The bit position is specified by the contents (0 to 15) of the low-order 4 bits (b3 to b0) of n(WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 15 can be specified as n (constant) (Decimal)

When d indicates a double word:

- The bit position is specified by the contents (0 to 31) of the low-order 5 bits (b4 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 31 can be specified as n (constant). (Decimal)

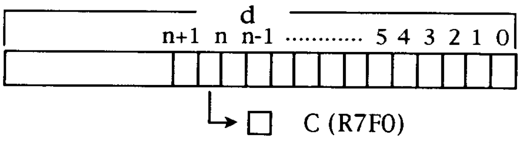
[Precautions]

<table>
<tr><td>Name</td><td colspan="9">BIT TEST</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">BTS(d,n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6">Upper values: for word.
Lower values: for doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">32.6 ~ 86.9</td><td rowspan="3">53.8 ~ 135.0</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>↕</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">BTS (d,n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">35.4 ~ 88.9</td><td rowspan="2">53.8 ~ 135.0</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>I/O to be tested</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>n</td><td>Bit position to be tested</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>The constant is specified in decimal.</td></tr>
<tr><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]

- The contents of the "n"th bit of the I/O (word or double word) specified by d are checked. When the contents are 1, C (R7F0) is set to 1. When the contents are 0, C (R7F0) is reset to 0.
- The content of "d" is changed.



When d indicates a word:

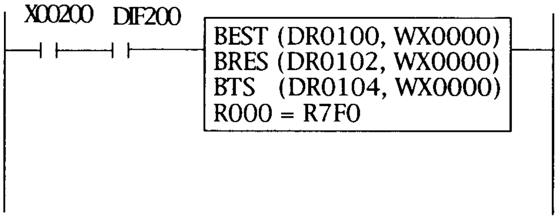
- The bit position is specified by the contents (0 to 15) of the low-order 4 bits (b3 to b0) of n(WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 15 can be specified as n (constant) (Decimal)

When d indicates a double word:

- The bit position is specified by the contents (0 to 31) of the low-order 5 bits (b4 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 31 can be specified as n (constant). (Decimal)

[Precautions]

[Program example]



```
LD X00200
AND DIF200
[
BEST (DR0100,
WX0000)
BRES (DR0102,
WX0000)
BTS  (DR0104,
WX0000)
R000 = R7F0
]
```

[Program explanation]

When WX0000 = H1234 at the rising edge of X00200 (WX0000 = $000100100011\underbrace{0100}_{\text{20 (decimal)}}$)

Assuming that DR0100 = H00000000, DR0102 = HFFFFFFFF, and DR0104 = H5555AAAA are set, at the rising edge of X00200:
the 20th bit of DR0100 is set to 1 by BSET,

```
          b31 ——— b20 ——————————— b0
DR0100=00000000000000000000000000000000
                  ↑
     This bit is set to 1.
```

the 20th bit of DR0102 is reset to 0 by BRES.

```
          b31 ——— b20 ——————————— b0
DR0102=11111111111111111111111111111111
                  ↑
     This bit is set to 0.
```

the 20th bit of DR0104 is checked by BTS.

```
          b31 ——— b20 ——————————— b0
DR0104=01010101010101010110101010101010101010
                  ↑
     This bit is checked,
```

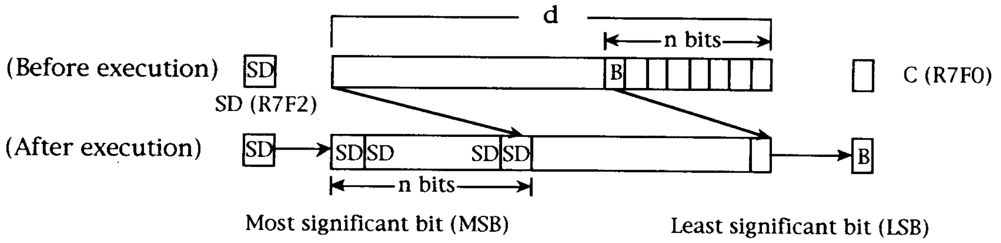
Since the 20th bit is 1, C (R7F0) is set to 1.

Name	SHIFT RIGHT							
Ladder format	Condition code					Processing time (μ s)		Remarks
SHR(d,n)	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	Upper values: for word. Lower values: for doubleword.
	DER	ERR	SD	V	C	41.4 ~ 93.1	67.0 ~ 144.3	
	●	●	●	●	↕			
Command format	No. of steps							
SHR (d,n)	Conditions				Step	53.3 ~ 209.9	84.3 ~ 319.0	
	—				3			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	I/O to be shifted						○	○	○		○	○		
n	No. of bits to be shifted					○	○	○	○				○	The constant is specified in decimal.

[Function]

- The contents of d are shifted right (low-order direction) n bit positions.
- The SD (R7F2) contents are set in n bits from the most significant bit.
- The contents of the "n"th bit from the least significant bit are set in C (R7F0).



When d indicates a word:

- The shift amount is specified by the contents (0 to 15) of the low-order 4 bits (b3 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 15 can be specified as n (constant). (Decimal)

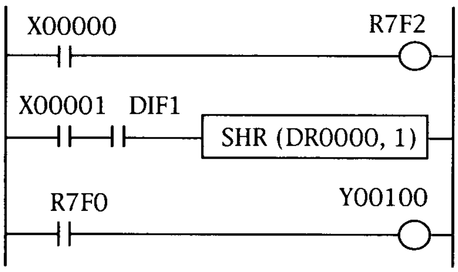
When d indicates a double word:

- The shift amount is specified by the contents (0 to 31) of the low-order 5 bits (b4 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 31 can specified as n (constant). (Decimal)

[Precautions]

- When n = 0, the contents of d are not shifted. C holds the previous status.

[Program example]



X00000 ······Input of a defective To SD

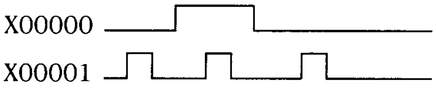
X00001 ······Conveyor movement

Y00100 Output of a defective carry

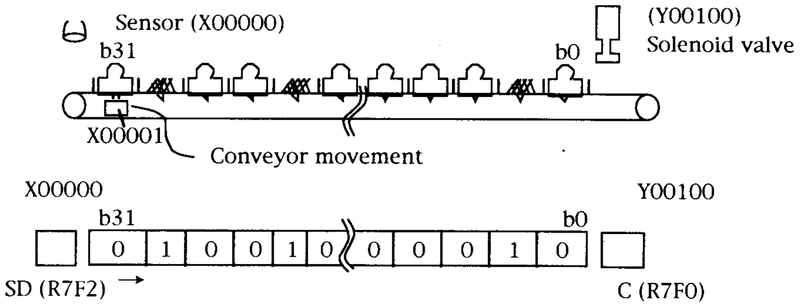
```
LD X00000
OUT R7F2
LD X00001
AND DIF1
[
SHR (DR0000, 1)
]
LD R7F0
OUT Y00100
```

[Program explanation]

- A conveyor with 32 stands moves to the right,
- Whenever one stand moves to the right, 1 pulse is supplied to X1.
- A sensor is mounted at the left end of the conveyor. When a defective is put on the conveyor, X00000 is turned ON.
  The X00000 (sensor input) and X00001 (conveyor movement) signals are as follows:



- When the conveyor moves to the right, data is shifted bit by bit. When the data is outputted to the carrier (at the right end of the conveyor), the (Y00100) solenoid valve is turned ON and the defective is ejected.



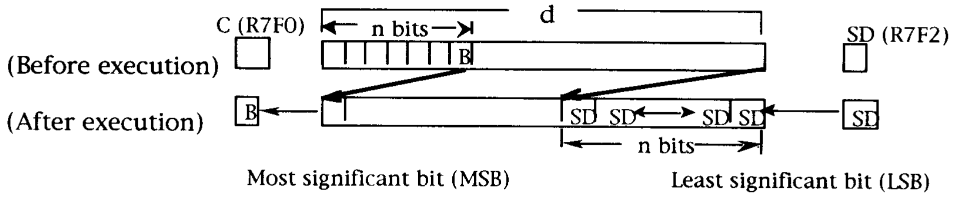
Data is shifted bit by bit.

<table>
<tr><td>Name</td><td colspan="8">SHIFT LEFT</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">SHL(d,n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">Upper values: for word.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">41.2
~
93.1</td><td rowspan="3">66.7
~
144.3</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>↕</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">SHL (d,n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">54.3
~
211.2</td><td rowspan="2">86.2
~
320.9</td><td rowspan="2">Lower values: for doubleword.</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>I/O to be shifted</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>n</td><td>No. of bits to be shifted</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>The constant is specified in decimal.</td></tr>
<tr><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]

- The contents of d are shifted left (high-order direction) n bit positions.
- The SD (R7F2) contents are set in n bits from the least significant bit.
- The contents of the "n"th bit from the most significant bit are set in C (R7F0).



When d indicates a word:

- The shift amount is specified by the contents (0 to 15) of the low-order 4 bits (b3 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 15 can be specified as n (constant). (Decimal)

When d indicates a double word:

- The shift amount is specified by the contents (0 to 31) of the low-order 5 bits (b4 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 31 can specified as n (constant). (Decimal)

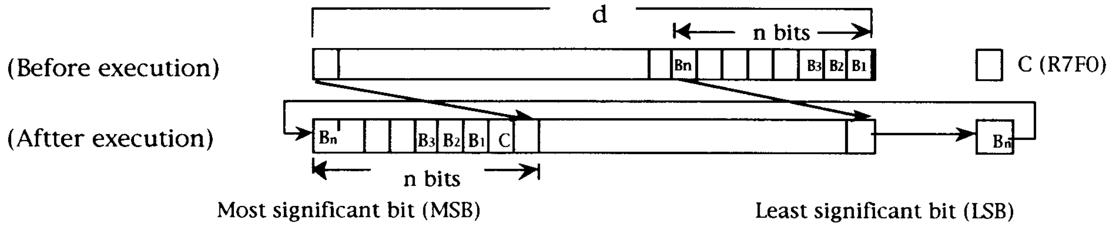
[Precautions]
- When n (shift amount) is 0, the contents of d are not shifted. C holds the previous status.

Name	ROTATE RIGHT							
Ladder format	Condition code					Processing time (μ s)		Remarks
ROR(d,n)	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	Upper values: for word.
	DER	ERR	SD	V	C	41.2 ~ 76.3	66.7 ~ 119.2	
	●	●	●	●	↕			
Command format	No. of steps							
ROR (d,n)	Conditions			Step		54.0 ~ 174.9	85.7 ~ 266.7	Lower values: for doubleword.
	—			3				

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	I/O to be rotated						○	○	○		○	○		
n	No. of bits to be rotated					○	○	○	○				○	The constant is specified in decimal.

[Function]

- The contents of d are rotated right (low-order direction) n bit positions.
- The contents of C (R7F0) are inputted into the most significant bit and the contents of the least significant bit inputted into C (R7F0). This processing is repeated n times.
- The contents of C (R7F0) are set in the "n"th bit from the most significant bit.
- The contents of the "n"th bit from the least significant bit are set in C (R7F0).



When d indicates a word:

- The shift amount is specified by the contents (0 to 15) of the low-order 4 bits (b3 to b0) of n (WX,WY,WR,WL,WM,TC). (The higt-order bits are ignored and assumed as 0.)
- One of 0 to 15 can be specified as n (constant). (Decimal)

When d indicates a double word:

- The shift amount is specified by the contents (0 to 31) of the low-order 5 bits (b4 to b0) of n (WX,WY,WR,WL,WM,TC). (The higt-order bits are ignored and assumed as 0.)
- One of 0 to 31 can specified as n (constant). (Decimal)

[Precautions]

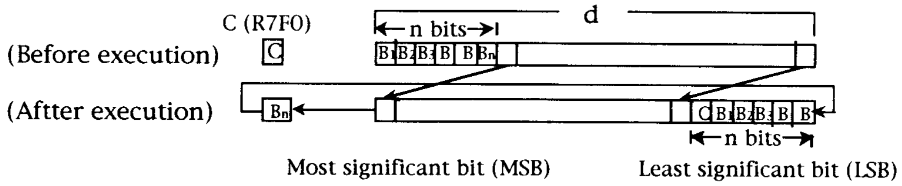
- When n (rotation amount) is 0, the contents of d are not rotated. C holds the previous status.

<table>
<tr><td>Name</td><td colspan="8">ROTATE LEFT</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">ROL(d,n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">Upper values:
for word.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">41.2
~
76.3</td><td rowspan="3">66.7
~
119.2</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>↕</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">ROL (d,n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">54.0
~
174.9</td><td rowspan="2">85.7
~
266.7</td><td rowspan="2">Lower values:
for doubleword.</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	I/O to be rotated						○	○	○		○	○		
n	No. of bits to be rotated					○	○	○	○				○	The constant is specified in decimal.

[Function]

- The contents of d are rotated left (high-order direction) n bit positions.
- The contents of C (R7F0) are set in the "n"th bit from the least significant bit.
- The contents of the "n"th bit from the most significant bit are set in C (R7F0).



When d indicates a word:

- The shift amount is specified by the contents (0 to 15) of the low-order 4 bits (b3 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 15 can be specified as n (constant). (Decimal)

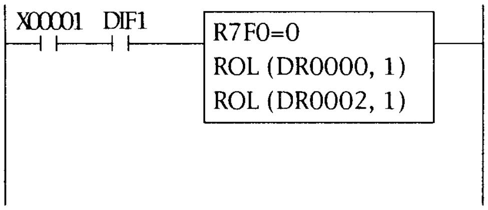
When d indicates a double word:

- The shift amount is specified by the contents (0 to 31) of the low-order 5 bits (b4 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 31 can specified as n (constant). (Decimal)

[Precautions]

- When n (rotation amount) is 0, the contents of d are not rotated. C holds the previous status.

[Program example]

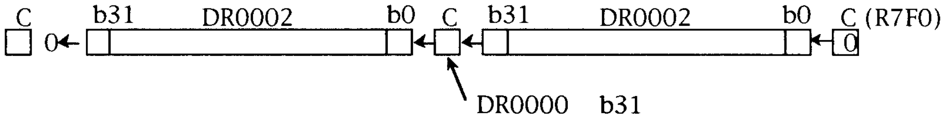


```
LD X00001
AND DIF1
[
R7F0 = 0
ROL (DR0000, 1)
ROL (DR0002, 1)
]
```

[Program explanation]

- 64-bit is shifted bit by bit at the rising edge of X00001. 0 is inputted in the shifted empty area.

Entire movement

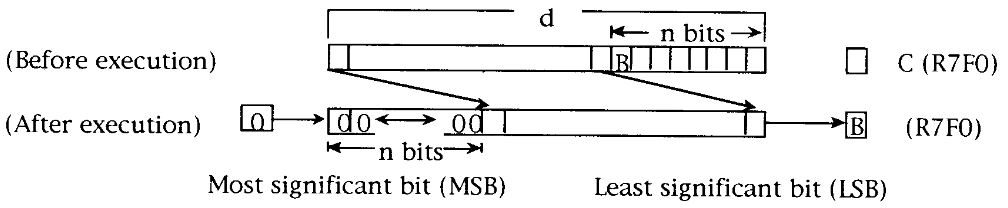


Name	LOGICAL SHIFT RIGHT							
Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
LSR(d,n)	DER	ERR	SD	V	C	35.0 ~ 70.1	57.4 ~ 109.9	Upper values: for word.
	●	●	●	●	↕			
Command format	No. of steps							
	Conditions				Step			
LSR (d,n)	—				3	47.7 ~ 168.7	76.4 ~ 257.4	Lower values: for doubleword.

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	I/O to be shifted						○	○	○		○	○		
n	No. of bits to be shifted					○	○	○	○				○	The constant is specified in decimal.

[Function]

- The contents of d are shifted right (low-order direction) n bit positions.
- 0 is set in n bits from the most significant bit.
- The contents of the "n"th bit from the least significant bit are set in C (R7F0).



When d indicates a word:

- The shift amount is specified by the contents (0 to 15) of the low-order 4 bits (b3 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 15 can be specified as n (constant). (Decimal)

When d indicates a double word:

- The shift amount is specified by the contents (0 to 31) of the low-order 5 bits (b4 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 31 can specified as n (constant). (Decimal)

[Precautions]

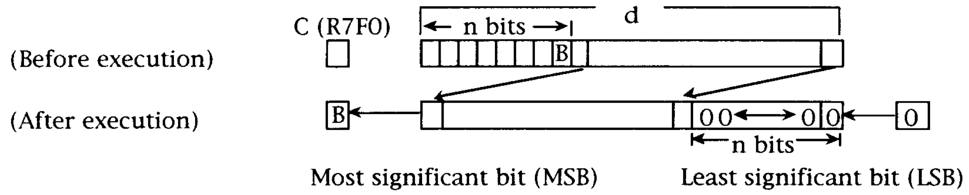
- When n (shift amount) is 0, the contents of d are not shifted. C holds the previous status.

<table>
<tr><td>Name</td><td colspan="9">LOGICAL SHIFT LEFT</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">LSL(d,n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">Upper values:
for word.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">35.0
~
70.1</td><td rowspan="3">57.4
~
109.9</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>↕</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">LSL (d,n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">47.7
~
168.7</td><td rowspan="2">76.4
~
257.4</td><td rowspan="2">Lower values:
for doubleword.</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>I/O to be shifted</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>n</td><td>No. of bits to be shifted</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>The constant is specified in decimal.</td></tr>
<tr><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]

- The contents of d are shifted left (high-order direction) n bit positions.
- 0 is set in n bits from the least significant bit.
- The contents of the "n"th bit from the most significant bit are set in C (R7F0).



When d indicates a word:

- The shift amount is specified by the contents (0 to 15) of the low-order 4 bits (b3 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 15 can be specified as n (constant). (Decimal)

When d indicates a double word:

- The shift amount is specified by the contents (0 to 31) of the low-order 5 bits (b4 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 31 can specified as n (constant). (Decimal)

[Precautions]

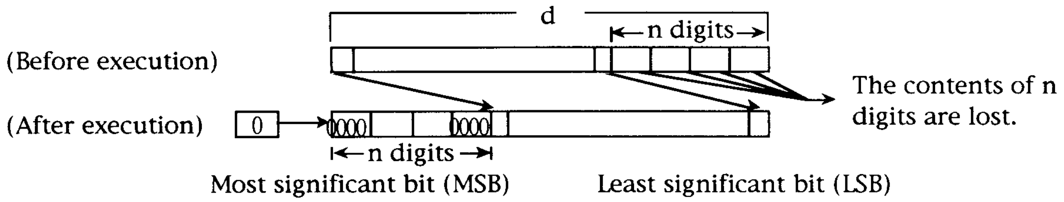
- When n (shift amount) is 0, the contents of d are not shifted. C holds the previous status.

<table>
<tr><td>Name</td><td colspan="9">BCD SHIFT RIGHT</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">BSR(d,n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">Upper values:
for word.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">34.3
~
49.2</td><td rowspan="3">56.4
~
78.6</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">BSR (d,n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">51.7
~
132.8</td><td rowspan="2">82.3
~
203.6</td><td rowspan="2">Lower values:
for doubleword.</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>I/O to be shifted</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>n</td><td>No. of bits to be shifted</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>The constant is specified in decimal.</td></tr>
<tr><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]

- The contents of d are shifted right (low-order direction) n digit positions. (One digit is 4 bits long.)
- 0 is set in n digits from the high-order position bit.
- The contents of n digits from the low order position are lost.



When d indicates a word:

- The shift amount is specified by the contents (0 to 3) of the low-order 2 bits (b1 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 3 can be specified as n (constant). (Decimal)

When d indicates a double word:

- The shift amount is specified by the contents (0 to 7) of the low-order 3 bits (b2 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 7 can specified as n (constant). (Decimal)

[Precautions]

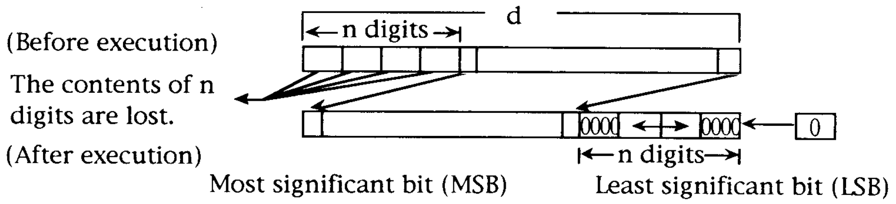
- When n (shift amount) is 0, the contents of d are not shifted.

<table>
<tr><td>Name</td><td colspan="8">BCD SHIFT LEFT</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">BSL(d,n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">Upper values: for word.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">34.3
~
49.2</td><td rowspan="3">56.4
~
78.6</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">BSL (d,n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">51.7
~
132.8</td><td rowspan="2">82.3
~
203.6</td><td rowspan="2">Lower values: for doubleword.</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	I/O to be shifted						○	○	○		○	○		
n	No. of bits to be shifted					○	○	○	○				○	The constant is specified in decimal.

[Function]

- The contents of d are shifted left (high-order direction) n digit positions. (One digit is 4 bits long.)
- 0 is set in n digits from the low-order position.
- The contents of n digits from the high-order position are lost.



When d indicates a word:

- The shift amount is specified by the contents (0 to 3) of the low-order 2 bits (b1 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 3 can be specified as n (constant). (Decimal)

When d indicates a double word:

- The shift amount is specified by the contents (0 to 7) of the low-order 3 bits (b2 to b0) of n (WX,WY,WR,WL,WM,TC). (The high-order bits are ignored and assumed as 0.)
- One of 0 to 7 can specified as n (constant). (Decimal)

[Precautions]

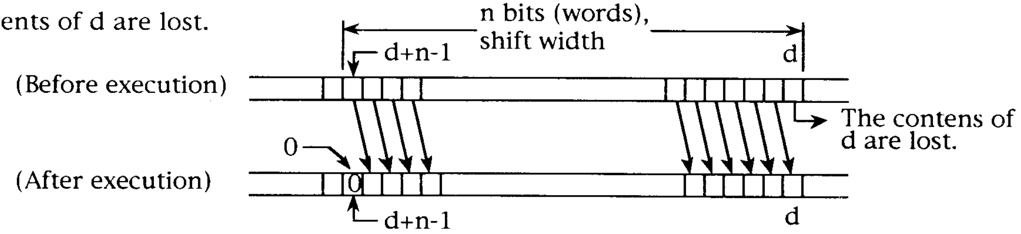
- When n (shift amount) is 0, the contents of d are not shifted.

<table>
<tr><td>Name</td><td colspan="8">Batch Shift Right (SHIFT RIGHT BLOCK)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">WSHR(d,n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6">Upper values: for bit.
Lower values: for word.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="2">65.7+ 9.4n</td><td rowspan="2">103.4+14.0n</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td><td rowspan="3">62.8+10.7n</td><td rowspan="3">98.9+16.0n</td></tr>
<tr><td rowspan="2">WSHR (d,n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Top I/O to be shifted			○				○						
n	No. of bits (words) to be shifted					○	○	○	○				○	The constant is specified in decimal.

[Function]

- n bits (words) from d to d+n-1 are shifted right (in the I/O number decreasing direction) one bit (word) position.
- 0 (H0000) is set in the bit (word) of d+n-1.
- The contents of d are lost.



When n is one of WX,WY,WR,WL,WM,and TC:
The contents (0 to 255) of the low-order 8 bits (b7 to b0) of n are the number of bits (words) to be shifted.

When n is a constant:
One of 0 to 255 can be specified as the number of bits (words) to be shifted.(Decimal)

[Precautions]

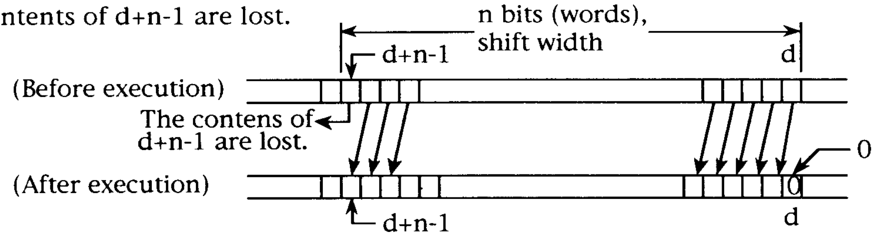
- Keep d+n-1 within the I/O limits (R7FF, LO3FFF, L13FFF, M3FFF, WR43FF(WR3FF for H-250), WL3FF, WL13FF ,WM3FF).
  When d+n-1 is beyond the limits, DER is "1" and the contents between d and the maximum limit are shifted.
- When n is 0, no batch shift is performed. DER (R7F4) is 0.
- In H-252, the internal output area is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, when the processing is assigned from WR3FF to WR400, DER=1 will be caused and the processing will be executed from d to WR3FF, and be disabled from WR400 thereafter.

Name	Batch Shift Left (SHIFT LEFT BLOCK)							
Ladder format	Condition code					Processing time (μ s)		Remarks
WSHL(d,n)	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	Upper values: for bit. Lower values: for word.
	DER	ERR	SD	V	C	61.6+ 9.4n	97.2+14.0n	
	↕	●	●	●	●			
Command format	No. of steps							
WSHL (d,n)	Conditions			Step		70.6+10.7n	110.7+16.0n	
	—			3				

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Top I/O to be shifted			○				○						
n	No. of bits (words) to be shifted					○	○	○	○				○	The constant is specified in decimal.

[Function]

- n bits (words) from d to d+n-1 are shifted left (in the I/O number increasing direction) one bit (word) position.
- 0 (H0000) is set in the bit (word) of d.
- The contents of d+n-1 are lost.



When n is one of WX,WY,WR,WL,WM,or TC:

The contents (0 to 255) of the low-order 8 bits (b7 to b0) of n are the number of bits (words) to be shifted.

When n is a constant:

One of 0 to 255 can be specified as the number of bits (words) to be shifted.(Decimal)

[Precautions]

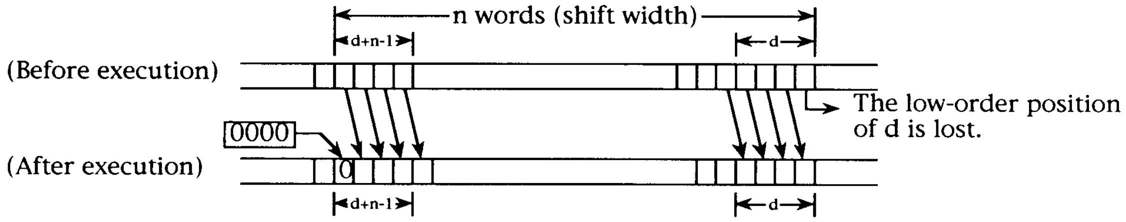
- Keep d+n-1 within the I/O limits (R7FF, LO3FFF, L13FFF, M3FFF, WR43FF(WR3FF for H-250), WL3FF, WL13FF, WM3FF).
  When d+n-1 is beyond the limits,DER is "1" and the contents between d and the maximum limit are shifted.
- When n is 0, no batch shift is performes. DER (R7F4) is 0.
- In H-252, the internal output area is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, when the processing is assigned from WR3FF to WR400, DER=1 will be caused and the processing will be executed from d to WR3FF, and be disabled from WR400 thereafter.

Name	Batch BCD Shift Right (SHIFT RIGHT BLOCK)							
Ladder format	Condition code					Processing time (μ s)		Remarks
WBSR (d,n)	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
	DER	ERR	SD	V	C	58.3+21.7n	92.3+32.4n	
	↕	●	●	●	●			
Command format	No. of steps							
WBSR (d,n)	Conditions			Step				
	—			3				

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Top I/O to be shifted							○						
n	No. of bits (words) to be shifted					○	○	○	○				○	The constant is specified in decimal.

[Function]

- n words from d to d+n-1 are shifted right (in the I/O number decreasing direction) one digit position as BCD data 4n digits long. (One digit is 4 bits long.)
- 0 is set in the high-order position d+n-1.
- The low -order position of d is lost.



When n is one of WX,WY,WR,WL,and WM:

The contents (0 to 255) of the low-order 8 bits (b7 to b0) of n are the number of words to be shifted.

When n is a constant:

One of 0 to 255 can be specified as the number of words to be shifted.(Decimal)

[Precautions]

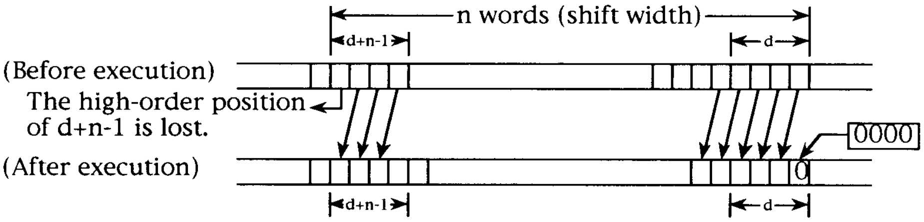
- Keep d+n-1 within the I/O limits (WR43FF(WR3FF for H-250), WL3FF, WL13FF, WM3FF). When d+n-1 is beyond the limits, DER is "1" and the contents between d and the maximum limit are shifted.
- When n is 0, no batch shift is performed.DER (R7F4) is 0.
- In H-252, the internal output area is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, when the processing is assigned from WR3FF to WR400, DER=1 will be caused and the processing will be executed from d to WR3FF, and be disabled from WR400 thereafter.

<table>
<tr><td>Name</td><td colspan="8">Batch BCD Shift Left (BCD SHIFT LEFT BLOCK)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">WBSL (d,n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">59.6+21.7n</td><td rowspan="5">94.3+32.4n</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">WBSL (d,n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R,
L,
M</td><td>TD, SS, WDT,
MS, TMR, CU,
RCU, CT</td><td>WX</td><td>WY</td><td>WR,
WL,
WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR,
DL,
DM</td></tr>
<tr><td>d</td><td>Top I/O to be shifted</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>n</td><td>No. of bits (words) to be shifted</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>The constant is specified in decimal.</td></tr>
<tr><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]

- n words from d to d+n-1 are shifted left (in the I/O number increasing direction) one digit position as BCD data 4n digits long. (One digit is 4 bits long.)
- 0 is set in the low-order position d+n-1.
- The high -order position of d is lost.



When n is one of WX,WY,WR,WL,WM, or TC:

The contents (0 to 255) of the low-order 8 bits (b7 to b0) of n are the number of words to be shifted.

When n is a constant:

One of 0 to 255 can be specified as the number of words to bee shifted.(Decimal)

[Precautions]

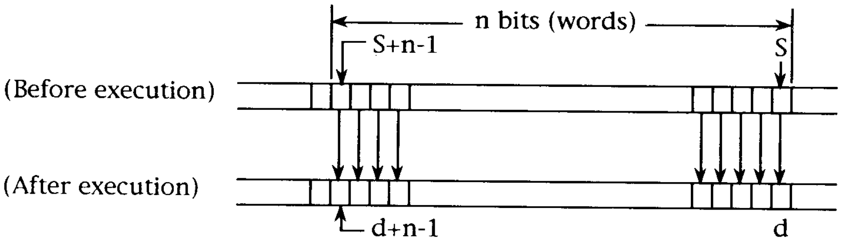
- Keep d+n-1 within the I/O limits (WR43FF(WR3FF for H-250),WL3FF,WL13FF,WM3FF). When d+n-1 is beyond the limits,DER is "1" and the contents between d and the maximum limit are shifted.
- When n is 0,no batch shift is performed.DER (R7F4) is 0.
- In H-252, the internal output area is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, when the processing is assigned from WR3FF to WR400, DER=1 will be caused and the processing will be executed from d to WR3FF, and be disabled from WR400 thereafter.

<table>
<tr><td>Name</td><td colspan="8">Block Transfer (MOVE)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">MOV (d,S,n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper values for Bit
Lower values for Word</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">126.5+8.4n</td><td rowspan="3">194.2+12.6n</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">MOV (d,S,n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">89.1+9.7n</td><td rowspan="3">138.2+14.5n</td></tr>
<tr><td colspan="3" rowspan="2">—</td><td colspan="2" rowspan="2">4</td></tr>
<tr></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>Top I/O to transfer destination</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>s</td><td>Top I/O to transfer source</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>n</td><td>Number of bits (words) to be transferred</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>The constant is specified in decimal.</td></tr>
</table>

[Function]

- n bits (words) from S to S+n-1 are transferred to d+n-1.
- The values from S to S+n-1 are held. When the range of the transfer source is overlapped with the of transfer destination,the transferred value is selected.



When n is one of WX,WY,WR,WL,WM, or TC:

The contents (0 to 255) of the low -order 8 bits (b7 to b0) of n are the number of bits (words) to be transferred.

When n is a constant:

One of 0 to 255 can be specified as the number of bits (words) to be transferred.(Decimal)

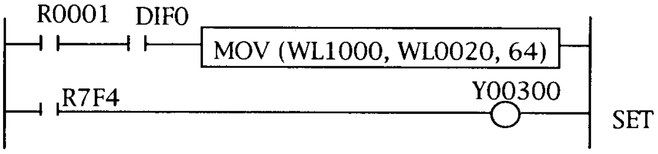
[Precautions]

- Keep d+n-1 and S+n-1 within the I/O limits (R7FF, L03FFF, L13FFF, M3FFF, WR43FF(WR3FF for H-250), WL3FF, WL13FF, WM3FF). When d+n-1 and S+n-1 are beyond the limits,DER becomes "1" and the bits (words), in numbers that is smaller of the ((the last bit)-d+1) or the ((the last bit)-S+1), are transferred.
- When n is 0,no batch transfer is performed. DER (R7F4) is 0.
- In H-252, the internal output area is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, when the processing is assigned from WR3FF to WR400, DER=1 will be caused and the processing will be executed from d to WR3FF, and be disabled from WR400 thereafter.

[Program example]

Data in the first link area (WL0020 to WL005F) is transferred to the second link area (WL1000 to WL103F).

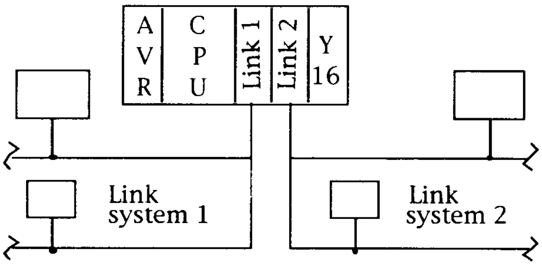
```
LD R0001
AND DIF0
[
MOV (WL1000, WL0020, 64)
]
LD R7F4
SET Y00300
```

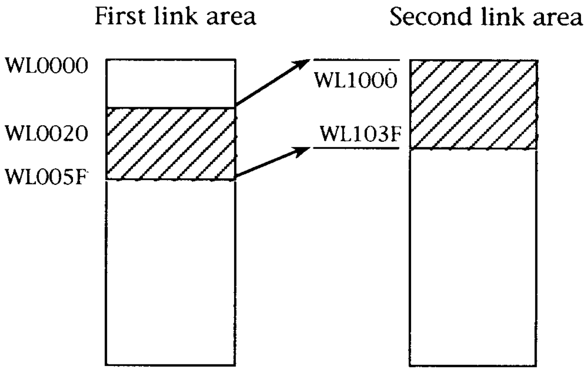


[Program explanation]

64-word data is transferred from link system 1 of the first link to link system 2 of the second link.

The transfer areas are WL0020 to WL005F and WL1000 to WL103F.



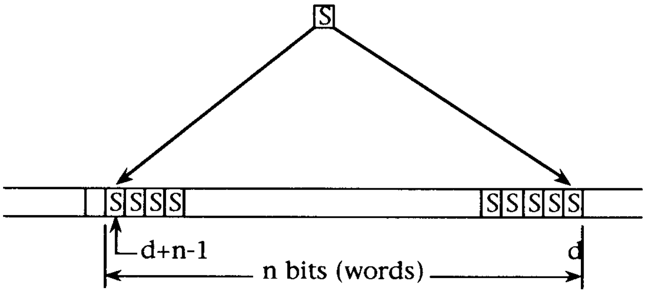


<table>
<tr><td>Name</td><td colspan="8">COPY</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">COPY(d,S,n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">Upper values for Bit</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">77.8+4.6n</td><td rowspan="3">121.5+6.8n</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">COPY(d,S,n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">65.9+5.2n</td><td rowspan="2">103.5+7.8n</td><td rowspan="2">Lower values for Word</td></tr>
<tr><td colspan="3">—</td><td colspan="2">4</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Top I/O of copy destination			○				○						
s	I/O of copy source	○	○	○		○	○	○	○				○	
n	Number of bits (words) to be copied					○	○	○	○				○	The constant is specified in decimal.

[Function]

- The value of S (bit,word) is copied from d to d+n-1.
- The value of S held.
- Bits are copied in bits and words are copied in words.



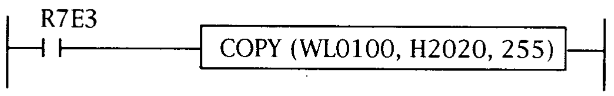
When n is one of WX, WY, WL,WM, or TC:
The contents (0 to 255) of the low-order 8 bits (b7 to b0) of n are the number of bits (words) to be copied.

When n is a constant:
One of 0 to 255 can be specified as the number of bits (words) to be copied. (Decimal)

[Precautions]

- Keep d+n-1 within the I/O limits (R7FF, LO3FFF, LI3FFF, M3FFF, WR43FF(WR3FF for H-250), WL3FF, WL13FF, WM3FF). When d+n-1 is beyond the limits, DER is "1" and the contents between d and the maximum limit are copied.
- When n is 0, no batch copy is performed. DER (R7F4) is 0,
- In H-252, the internal output area is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, when the processing is assigned from WR3FF to WR400, DER=1 will be caused and the processing will be executed from d to WR3FF, and be disabled from WR400 thereafter.

[Program example]



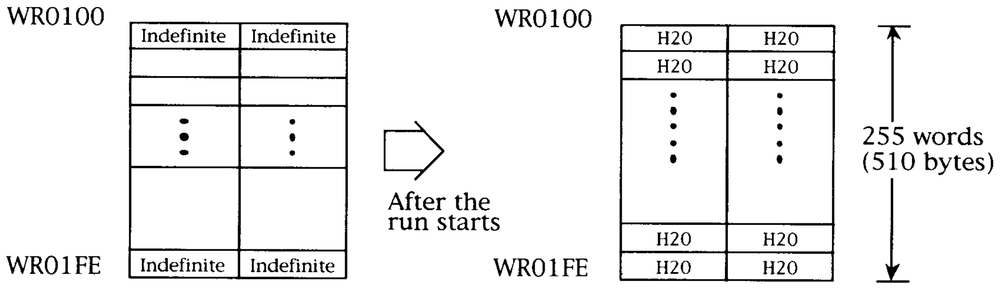
```
LD R7E3
[
COPY (WR0100, H2020, 255)
]
```

The default value (H2020) is set in the range from WR0100 to WR01FE.

[Program explanation]

The communication data area from WR0100 to WR01FE is filled with the space code (H20) as a default value at he first scan after the run starts.

R7E3: 1 scan ON after the run starts

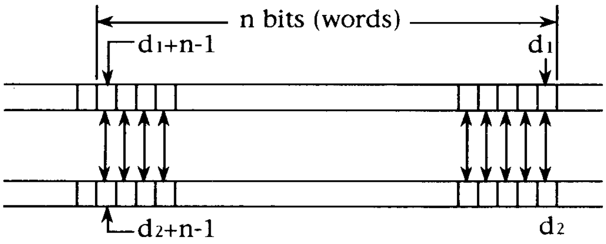


<table>
<tr><td>Name</td><td colspan="9">Block Exchange (EXCHANGE)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">XCG (d_1,d_2,n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper values for Bit
Lower values for Word</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">144.5+20.5n</td><td rowspan="3">221.2+30.6n</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">XCG (d_1,d_2,n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">140.9+19.4n</td><td rowspan="3">215.8+29.0n</td></tr>
<tr><td colspan="3" rowspan="2">——</td><td colspan="2" rowspan="2">4</td></tr>
<tr></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d_1	Top I/O to exchange destination			○				○						
d_2	Top I/O to exchange source			○				○						
n	Number of bits (words) to be exchange					○	○	○	○				○	The constant is specified in decimal.

[Function]

- The contents of n bits (words) from d1 to d_1+n-1 are exchanged the contents of n bits (words) from d_2 to d_2+n-1.
- Bits are exchanged with bits and words are exchanged with words.



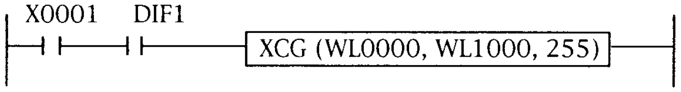
When n is one of WX,WY,WR,WL,WM,or TC:

The contents (0 to 255) of the low-order B bits (b7 to b0) of n are the number of bits (words) to be exchanged.

When n is a constant:

One of 0 to 255 can be specified as the number of bits (words) to be exchanged.(Decimal)10

Example:



WL0000 to WL00FE are exchanged with WL1000 to WL10FE.

[Precautions]

- Keep d+n-1 within the I/O limits (R7FF, L03FFF, L13FFF, M3FFF, WR43FF(WR3FF for H-250), WL3FF, WL13FF, WM3FF). When d+n-1 and S+n-1 are beyond the limits, DER is "1" and the contents up to the maximum limits of the number of bits (words) specified as d1 and d2 whichever smaller are exchanged.
- When n (block width) is 0, no batch exchange is performed. DER (R7F4) is 0.
- In H-252, the internal output area is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, when the processing is assigned from WR3FF to WR400, DER=1 will be caused and the processing will be executed from d to WR3FF, and be disabled from WR400 thereafter.

<table>
<tr><td>Name</td><td colspan="9">Inversion (NOT)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">NOT (d)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6">Upper values: for Bit.
Middle values for Word
Lower values: for Doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="2">29.1
~
30.6</td><td rowspan="2">48.6
~
50.9</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td><td rowspan="2">25.5</td><td rowspan="2">43.2</td></tr>
<tr><td rowspan="2">NOT (d)</td><td colspan="4">Conditions</td><td>Step</td></tr>
<tr><td colspan="4">——</td><td>2</td><td>35.9</td><td>58.7</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>I/O to be inverted</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></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></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>

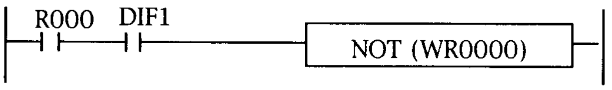
[Function]

- The bits of the contents of d are inverted.

$d \leftarrow \overline{d}$

Before execution	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
After execution	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1

[Program example]



```
LD  R000
AND  DIF1

NOT WR0000
```

[Program explanation]

The contents of WR0000 are inverted at the rising edge f R000.

Example: When the instruction is executed when WR0000 is H1234, WR0000 = HEDCB.When the instruction is executed once again, WR0000 = H1234.

[Precautions]

- The start condition of this instruction should be Edge Trigger.

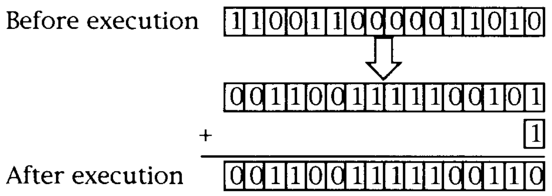
<table>
<tr><td>Name</td><td colspan="8">Twos Complement (NEGATE)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (µ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">NEG (d)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6">Upper values: for word.
Lower values: for doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">26.3</td><td rowspan="3">44.3</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">NEG (d)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">36.6</td><td rowspan="2">59.8</td></tr>
<tr><td colspan="3">—</td><td colspan="2">2</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>I/O to be complemented</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></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></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>

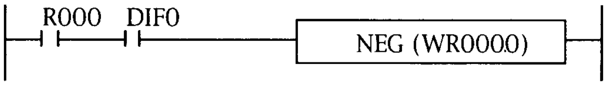
[Function]

- The twos complement of d is calculated. (The bits of the contents of d are inverted and added with 1. C(R7F0) is not changed.

$$d \leftarrow \bar{d}+1$$



[Program example]



```
LD R000
AND DIF0
NEG WR0000
```

[Program explanation]

- The twos complement of the contents of WR0000 is obtained at the rising edge of R0000.

Example: When the instruction is executed when WR0000 is H1234, WR0000 =HEDCC. When the instruction is executed once again, WR0000 = H1234.

[Precautions]

- The start condition of this instruction should be Edge Trigger.

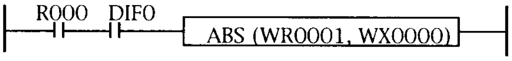
<table>
<tr><td>Name</td><td colspan="8">Absolute Value (ABSOLUTE)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">ABS (d,s)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper values: for word.
Lower values: for doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">29.6
~
32.3</td><td rowspan="3">49.4
~
53.3</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>↕</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">ABS (d,s)</td><td colspan="4">Conditions</td><td>Step</td><td rowspan="3">36.6
~
41.5</td><td rowspan="3">59.8
~
67.2</td></tr>
<tr><td colspan="4">Word</td><td>3</td></tr>
<tr><td colspan="4">Double word</td><td>4</td></tr>
</table>

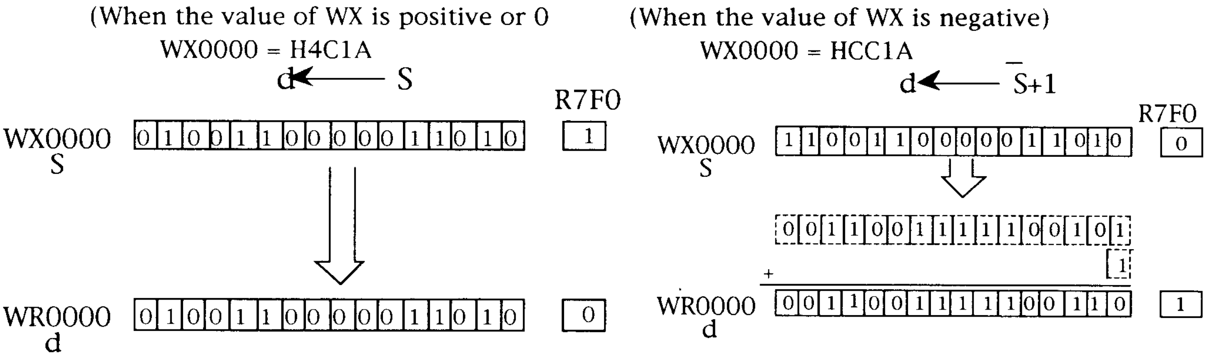
<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>I/O after absolute value taken</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>s</td><td>I/O before absolute value taken</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></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]

- When S is positive or O: The contents of S are set in d. C (R7F0) is 0.
- When S is negative :The twos complement of the contents of S is set in d. C (R7F0) is 1.
- For d or S, words are used in words ,and double words are used in double words.

Example:





(When the value of WX is positive or 0
WX0000 = H4C1A

WX0000 S: 0 1 0 0 1 1 0 0 0 0 0 1 1 0 1 0 R7F0: 1

WR0000 d: 0 1 0 0 1 1 0 0 0 0 0 1 1 0 1 0 R7F0: 0

(When the value of WX is negative)
WX0000 = HCC1A

WX0000 S: 1 1 0 0 1 1 0 0 0 0 0 1 1 0 1 0 R7F0: 0

0 0 1 1 0 0 1 1 1 1 1 0 0 1 0 1
+ 1

WR0000 d: 0 0 1 1 0 0 1 1 1 1 1 0 0 1 1 0 R7F0: 1

When S is a word : 0 ~ 32 767 (decimal) corresponds to H0000 ~ H7FFF (hexadecimal).
-32 768~-1(decimal) corresponds to H8000~HFFFF (hexadecimal).

When S is a double word: 0 ~ 2 147 483 647 (decimal) corresponds to H00000000 ~ H7FFFFFFF (hexadecimal).
-2 147 483 648~-1 (decimal) corresponds to H80000000 ~ HFFFFFFFF (hexadecimal).

[Precautions]

- The start condition of this instruction should be Edge Trigger.

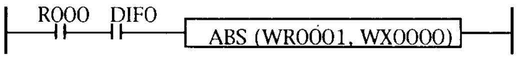
<table>
<tr><td>Name</td><td colspan="8">Sign Addition (SIGN GET)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">SGET (d,s)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">Upper values: for word.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">27.6
~
30.2</td><td rowspan="3">46.3
~
50.2</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">SGET (d,s)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">32.8
~
37.8</td><td rowspan="3">46.3
~
5.02</td><td rowspan="3">Lower values: for doubleword.</td></tr>
<tr><td colspan="3">Word</td><td colspan="2">3</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">4</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>I/O after sign addition</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>s</td><td>I/O before sign addition</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></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>

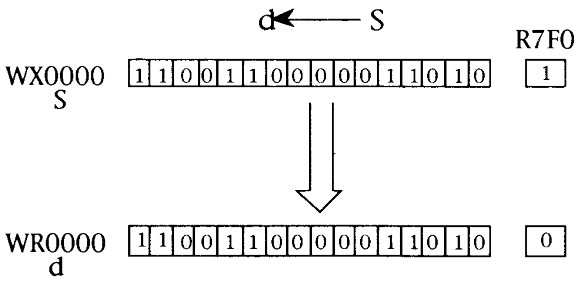
[Function]

- When C (R7F0) is 0 : The contents of S are set in d.
- When C (R7F0) is 1 : The twos complement of the contents of S is set in d.
- The contents of C (R7F0) are not changed.
- For d or S, words are used in words ,and double words are used in double words.

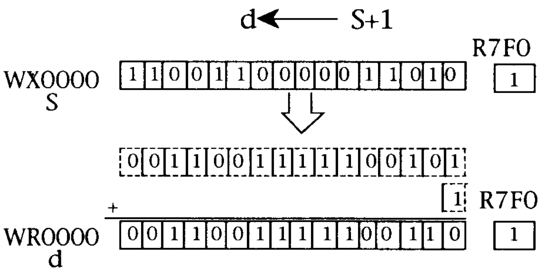
Example:



When C (R7F0) is 0



When C (R7F0) is 1



[Precautions]

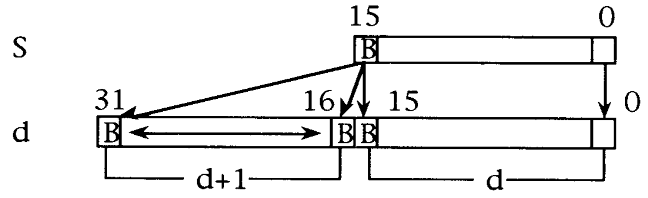
- The start condition of this instruction should be Edge Trigger.

<table>
<tr><td>Name</td><td colspan="8">Sign Extension (EXTEND)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">EXT (d,s)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">27.5
~
27.9</td><td rowspan="5">46.1
~
46.8</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">EXT (d,s)</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

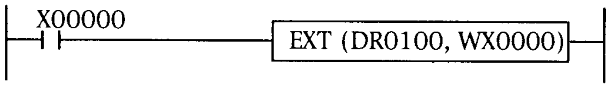
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	I/O after sign extension										○	○		
s	I/O before sign extension					○	○	○	○				○	

[Function]

- The sign bit (most significant bit) of S is extended to a high-order word of d.
- The low-order word of d is the contents of S.



[Program example]



```
LD X00000
[
EXT (DR0100, WX0000)
]
```

[Program explanation]

- When X00000 is turned ON, the contents of WX0000 are extended to DR0100.

When X00000 is positive or 0	When WX00000 is negative
Example: WX00000 = H7FFF (+32 767) → DR0100 = H00007FFF (+32 767)	Example : WX00000 = H8000 (-32 768) → DR0100 = HFFFF8000(-32 768)

[Precautions]

<table>
<tr><td>Name</td><td colspan="9">Binary → BCD Conversion (BCD)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">BCD(d,s)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">Uppervalues for Word</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">60.3
~
64.5</td><td rowspan="3">95.2
~
101.5</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">BCD (d,s)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">109.8
~
142.0</td><td rowspan="3">169.2
~
217.4</td><td rowspan="3">Lower values for Double word</td></tr>
<tr><td colspan="3">Word</td><td colspan="2">3</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">4</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>I/O(BCD) after conversion</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>s</td><td>I/O(BIN) before conversion</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></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]

- The contents of S are converted from a binary number to a BCD number and the result is outputted to d.
- When the conversion result of S is larger than the number of digits of the BCD data of d, DER (R7F4) is 1 and no conversion is executed.

When S is a word : Set it as follows : H0000 ≦ S ≦ H270F (0 to 9999)
When S is a double word: Set it as follows : H00000000 ≦ S ≦ H05F5EOFF (0 to 99999999)

		1				B				4				F				
(Before execution)	S	0	0	0	1	1	0	1	1	0	1	0	0	1	1	1	1	(Binary) H1B4F = 6991

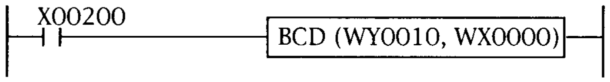
⇩

		6				9				9				1				
(After execution)	d	0	1	1	0	1	0	0	1	1	0	0	1	0	0	0	1	(BCD)

Combination of d and S

d	S
Word	Word
Double word	Double word

Example:



[Precautions]

- When a data error occurs, the contents of d are left unchanged.

<table>
<tr><td>Name</td><td colspan="8">BCD → Binary Conversion (BINARY)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">BIN (d, S)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7">Upper values: for word.
Lower values: for doubleword.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">65.7 ~ 78.6</td><td rowspan="3">103.3 ~ 122.6</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">BIN (d, S)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="3">97.0 ~ 157.0</td><td rowspan="3">150.1 ~ 239.8</td></tr>
<tr><td colspan="3">Word</td><td colspan="2">3</td></tr>
<tr><td colspan="3">Double word</td><td colspan="2">4</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>I/O(BIN) after conversion</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>s</td><td>I/O (BCD) before conversion</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></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]

- The contents of S are converted from a BCD number to a binary number and the result is outputted to d.
- When the contents of S are not BCD data (when A to F are found in the data), DER (R7F4) is 1 and no conversion is executed. (d is left unchanged.)

		6				9				9				1				
(Before execution)	S	0	1	1	0	1	0	0	1	1	0	0	1	0	0	0	1	(BCD)

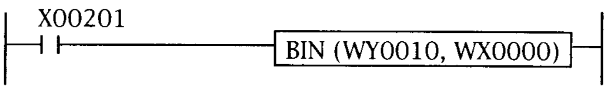
⇩

		1				B				4				F				
(After execution)	d	0	0	0	1	1	0	1	1	0	1	0	0	1	1	1	1	(Binary)

Combination of d and S

d	S
Word	Word
Double word	Double word

Example:



[Precautions]

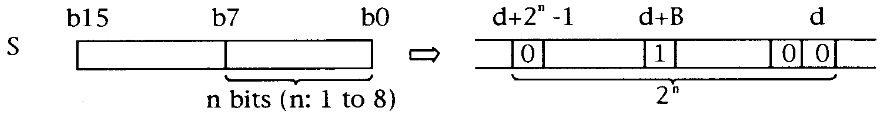
- When a data error occurs, the contents of d are left unchanged.

Name	DECODE							
Ladder format	Condition code					Processing time (μ s)		Remarks
DECO (d, S, n)	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
	DER	ERR	SD	V	C	97.5 +5.6 × 2^n	151.0 +8.3 × 2^n	
	↕	●	●	●	●			
Command format	No. of steps							
	Conditions			Step				
DECO(d, S, n)	—			4				

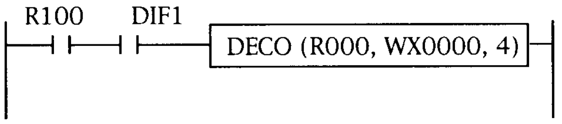
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Top I/O to decoding destination			○										
S	Word I/O to be decoded					○	○	○	○				○	
n	No. of bits to be decoded												○	1 to 8 (decimal)

[Function]

- The low-order n bits of S are decoded to 2^n and 1 is outputted to the decoded bits of the bit string from d to d+2^n-1. (n: 1 to 8)
- When n is 0, no data is decoded. The contents from to d d+2^n-1 are left unchanged.



[Program example]



```
LD R100
AND DIF1
[
DECO (R000, WX0000, 4)
]
```

[Program explanation]

At the rising edge of R100, a bit , that is appointed with the value of low-order 4 bits of WX0000 from top address R000, is set to 1.

Example: When WX0000 = HFFFF, 1 is set in the 15th bit, that is, R00F from R000 in this program.

[Precautions]

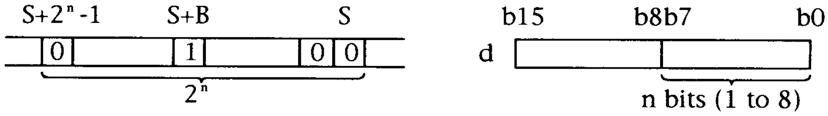
- Keep d+2^n-1 within the I/O limits(R7FF, L03FFF, L13FFF, M3FFF). When d+2^n-1 is beyond the limits, DER becomes "1", and d to the last bit are decoded.
- The value of n should be one of 1 to 8.

<table>
<tr><td>Name</td><td colspan="9">ENCODE</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">ENCO(d, S, n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">87.6
+6.8 × 2^n</td><td rowspan="5">136.1
+10.1×2^n</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>↕</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">ENCO(d, S, n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">4</td></tr>
</table>

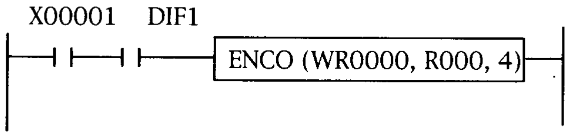
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Top I/O of encoding destination						○	○						
S	Top I/O of bit string to be encoded			○										
n	No. of bits to be encoded												○	1 to 8 (decimal)

[Function]

- The value of bit position, that is set to 1 and within 2^n bits from S to S+2^n--1, is encoded and set in d.
- When n is 0, no data is encoded. The contents of d are left unchanged.
- When a plurality of ones are found between S and S+2^n-1, the high-order bit is encoded. When the all bits are 0, C = 1 and d = 0.



[Program example]



LD X00001
AND DIF1
[
ENCO (WR0000, R000, 4)
]

[Program explanation]

- The most significant bit where 1 is set is detected from the bit string between R000 and R00F (2^4-1=15 bits) at the rising edge of X00001 and a binary number 4 bits in length is set in the word I/O of d.

Example: When 1 is set in the 7th and 6th bits of R000 to R00F, H0007 is set in WR0000.

[Precautions]

- Keep $S+2^n-1$ within the I/O limits(R7FF, LO3FFF, L13FFF, M3FFF).When $S+2^n-1$ is beyond the limits, DER becomes "1" and s to the last bit are encoded.

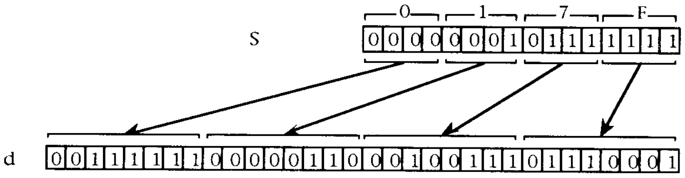
- The value of n should be one of 1 to 8.

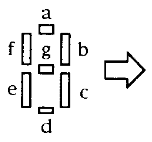
<table>
<tr><td>Name</td><td colspan="8">7-Segment Decode (SEGMENT)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">SEG (d, S)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">73.4</td><td rowspan="5">114.8</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">SEG (d, S)</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>I/O of decoding destination</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>s</td><td>Decoding contents</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></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]

- The contents of S are converted to a 4-digit 7-segment display code on the assumption that one digit is 4 bits in length and the result is outputted to d.





Conversion table

<table>
<tr><td rowspan="2">Input date, 4 bits</td><td colspan="8">Output data</td><td rowspan="2">Dis-play</td></tr>
<tr><td></td><td>g</td><td>f</td><td>e</td><td>d</td><td>c</td><td>b</td><td>a</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td></tr>
<tr><td>2</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td><td>2</td></tr>
<tr><td>3</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>3</td></tr>
<tr><td>4</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>4</td></tr>
<tr><td>5</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>5</td></tr>
<tr><td>6</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>6</td></tr>
<tr><td>7</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>7</td></tr>
<tr><td>8</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>8</td></tr>
<tr><td>9</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>9</td></tr>
<tr><td>A</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td><td>A</td></tr>
<tr><td>B</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>B</td></tr>
<tr><td>C</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>C</td></tr>
<tr><td>D</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>D</td></tr>
<tr><td>E</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>E</td></tr>
<tr><td>F</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>F</td></tr>
</table>

[Precautions]

Name	SQUARE ROOT							
Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
SQR (d, S)	DER	ERR	SD	V	C	1270.9 ~ 1572.0	1905.6 ~ 2355.8	
	↕	●	●	●	●			
Command format	No. of steps							
	Conditions			Step				
SQR (d, S)	—			4				

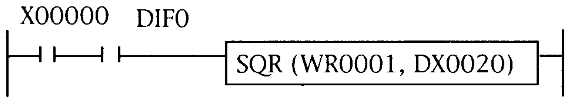
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Square root (BCD)						○	○						
s	I/O (BCD) with square root to be calculated									○	○	○	○	

[Function]

- The square root of the contents of S is calculated and the result is outputted to d.
- Set BCD data in S.
  When the BCD data of S is faulty, DER (R7F4) is 1 and no operation is performed. (There is data other than H0 to H9.)
- The figures below the decimal point are omitted.

$$\boxed{\quad d \quad} \longleftarrow \sqrt{\boxed{\quad S \quad}}$$

[Program example]



```
LD X00000
AND DIFO
[
SQR (WR0001, DX0020)
]
```

[Program explanation]

- The square root of the value of DX0020 is calculated at the rising edge of X00000 and the result is substituted in WR0001.

  Example: The following is obtained by operation when DX0020 = H00002159 (BCD).

  WR0001 = H0046 (BCD) ($\sqrt{2159}$ = 46.465...)

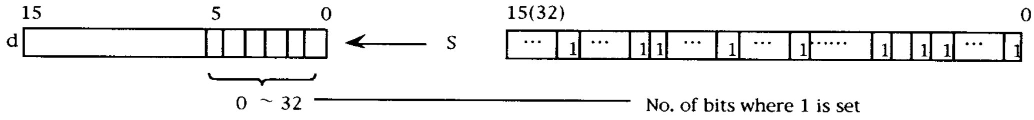
[Precautions]

Name	BIT COUNT							
Ladder format	Condition code					Processing time (μ s)		Remarks
BCU (d, S)	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	Upper values: for word. Lower values: for double word.
	DER	ERR	SD	V	C	77.1 ~ 80.6	120.3 ~ 125.5	
	●	●	●	●	●			
Command format	No. of steps							
BCU (d, S)	Conditions			Step		133.2 ~ 140.2	204.2 ~ 214.7	
	Word			3				
	Double word			4				

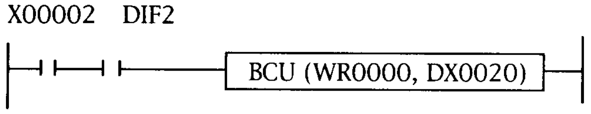
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	No. of bits of 1						○	○						
s	I/O for counting bits of 1					○	○	○	○	○	○	○	○	

[Function]

- The number of bits of the contents (16 bits for word or 32 bits for double word) of S where 1 is set outputted to d.



[Program example]



LD X00002
AND DIF2
[
BCU (WR0000, DX0020)
]

[Program explanation]

- The number of bits where 1 is set is counted from the data inputted in DX0020 at the rising edge of X00002 and the result is set in WR0000.

Example: The number of bits where 1 is set is 16 (decimal) when

	A	7	1	4	F	1	5	3
DX0020 =	1 0 1 0	0 1 1 1	0 0 0 1	0 1 0 0	1 1 1 1	0 0 0 1	0 1 0 1	0 0 1 1

Therefore, the result is WR0000 = H0010.

[Precautions]

Name	Exchange (SWAP)

Ladder format	Condition code					Processing time (μ s)		Remarks
SWAP (d)	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
	DER	ERR	SD	V	C	24.7	42.1	
	●	●	●	●	●			

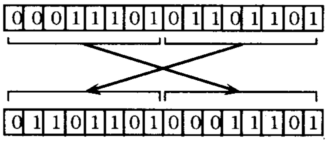
Command format	No. of steps	
SWAP (d)	Conditions	Step
	—	2

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	I/O to be exchanged						○	○						

[Function]

. The high-order 8 bits of the contents of d are exchanged with the low-order 8 bits.

(Before execution) d



(After execution) d

[Precautions]

. The start condition of this instruction should be Edge Trigger.

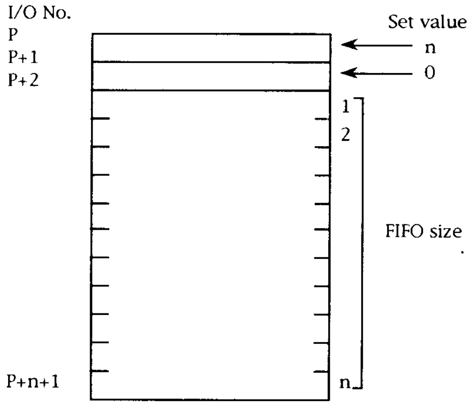
<table>
<tr><td>Name</td><td colspan="8">Fifo Initial (FIFO INITIALIZE)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FIFIT (P, n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">116.1</td><td rowspan="5">178.6</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FIFIT (P, n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
P	FIFO top I/O							○						
n	FIFO size												○	0 to 256

[Function]

FIFO is an abbreviation for first-in first-out. Data is stored in the buffer and fetched in the order of entry. This instruction initializes FIFO.

- The FIFO top I/O number P and the FIFO size n are set.
  When 0 ≦ n ≦ 256, n is set in P.
  When 257 ≦ n, 256 is set in P.
- The initial value 0 is set in P+1 as a use count of FIFO.
- n+2 words from P to P+n+1 are used for FIFO.



[Precautions]

- When P+n+1 is beyond the I/O limits (WR43FF(WR3FF for H-250), WLO3FF, WL13FF, WM3FF), DER (R7F4) is 1 and the maximum value (last) of the range - (P+1) is set in P.
- In H-252, the internal output area is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, when the processing is assigned from WR3FF to WR400, DER=1 will be caused and the processing will be executed from p to WR3FF, and be disabled from WR400 thereafter.

Name	FIFO WRITE							
Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
FIFWR (P, S)	DER	ERR	SD	V	C	134.9	206.8	
	↕	●	●	●	●			
Command format	No. of steps							
	Conditions			Step				
FIFWR (P, S)	—			3				

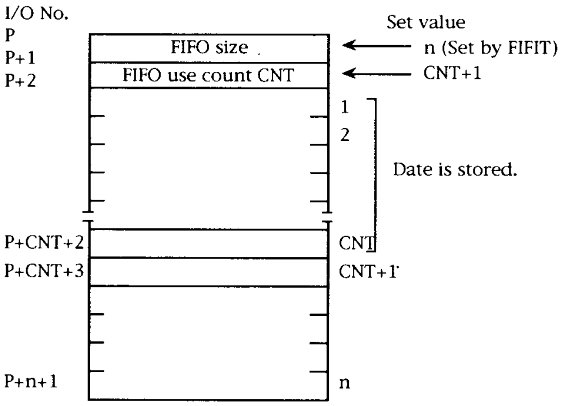
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
P	FIFO top I/O							○						
n	Contents to be written into FIFO					○	○	○	○				○	

[Function]

Data is written into the FIFO buffer of the top I/O number P.

When use count CNT < size n, the contents of S are written into P+CNT+2. 1 is added to the use count CNT.

When use count CNT ≧ size n, DER (R7F4) is 1 and no data is written.



[Precautions]

- When P+n+1 is beyond the I/O limits (WR43FF(WR3FF for H-250), WLO3FF, WL13FF, WM3FF), DER (R7F4) is 1 and no data is written.
- In H-252, the internal output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

Name	FIFO READ							
Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
FIFRD(P,d)	DER	ERR	SD	V	C	104.4 +11.1n	161.2 +16.6n	n denotes FIFO size.
	↕	●	●	●	●			
Command format	No. of steps							
FIFRD(P,d)	Conditions				Step			
	—				3			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
P	FIFO TOP I/O							○						
d	I/O for storing read date						○	○	○					

[Function]

Data in the FIFO buffer of the top I/O number P is read.

When 1 ≦ use count CNT ≦ size n :

The contents of P+2 are read and written into d.

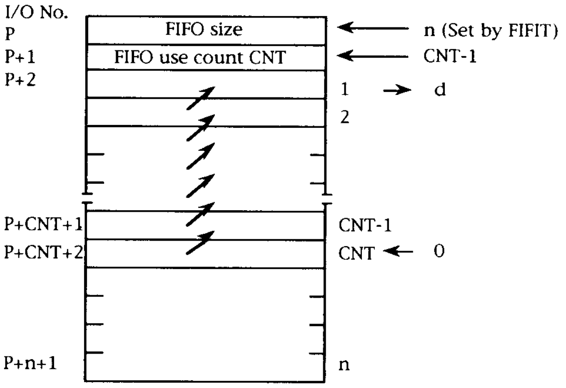
The contents of P+3 to P+CNT+2 are transferred to the previous I/O respectively.

O is written into P+CNT+2.

The contents of CNT are subtracted by1. When CNT = 0, no data is read. The previous data is held.

When use count CNT > size n :

DER (R7F4) is 1 and no data is read.



[Precautions]

- When P+n+1 is beyond the I/O limits (WR43FF(WR3FF for H-250), WLO3FF, WL13FF, WM3FF), DER (R7F4) is 1 and no data is read.
- In H-252, the internal output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="9">Connection (UNIT)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td colspan="2">Remarks</td></tr>
<tr><td rowspan="3">UNIT (d, S, n)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7" colspan="2"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="6">82.7
~
111.0</td><td rowspan="6">128.8
~
171.0</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">UNIT (d, S, n)</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3" rowspan="2">—</td><td colspan="2" rowspan="2">4</td></tr>
<tr></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>d</td><td>Write destination I/O of UNIT result</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>S</td><td>Top I/O to be connected</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>n</td><td>No. of words to be connected</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>n = 0 to 4</td></tr>
</table>

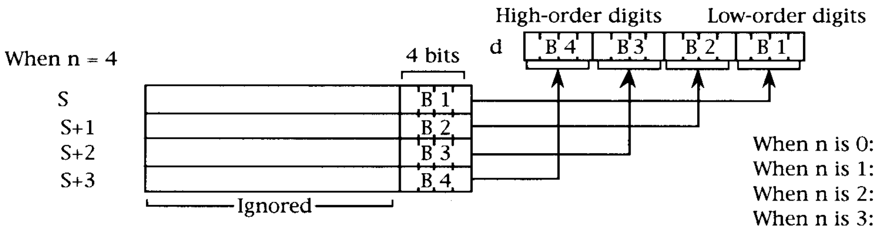
[Function]

The values of low-order 4 bits of n (1 to 4) words from S are set in units of 4 bits from the low-order position of d.

When n is one of 1 to 3, the bits which are not set in d are set to 0.

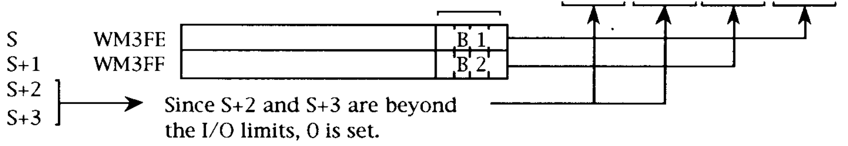
The data from S to S+n-1 is held even if the Unit instruction is executed.

When S+n-1 is beyond the I/O limits (WR43FF(WR3FF for H-250), WLO3FF, WL13FF, WM3FF), DER (R7F4) is 1 and only the low-order 4 bits from S to the I/O range are set in d.



When n is 0: B1 to B4 of d are 0.
When n is 1: B2 to B4 of d are 0.
When n is 2: B3 and B4 of d are 0.
When n is 3: B4 of d is 0.

When S+n-1 is beyond the I/O limits (S = WM3FE, n = 4):



[Precautions]

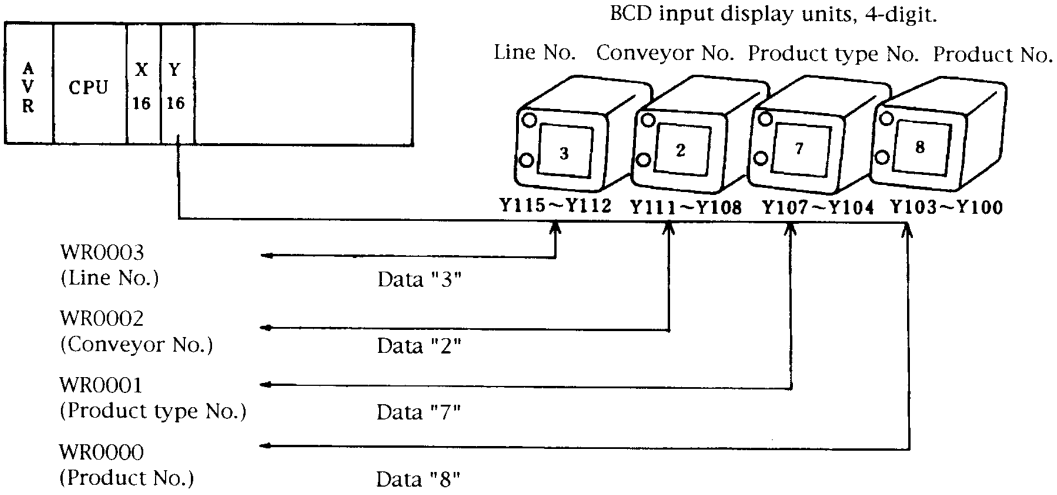
- When n =0 , the instruction is not executed.
- In H-252, the internal output area is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, when WR3FF and WR400~43FF are assigned simultaneously in the processing, DER=1 will be caused and the processing will be executed from S to WR3FF, and be disabled from WR400 thereafter.

[Program example]

LD X00001
AND DIFO
[
UNIT (WY0010, WR0000,4)
]

[Program explanation]

A 4-digit BCD input display unit is connected to WY0010 and independend data of WRO to WR3 is displayed in each digit. (Data of only low-order 4 bits of WRO to WR3 are valid.)



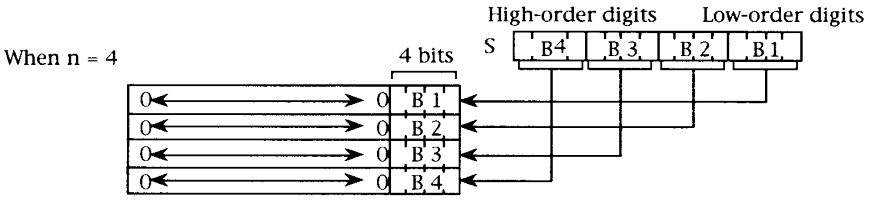
Name	Separation (DISTRIBUTE)

Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
DIST (d, S, n)	DER	ERR	SD	V	C	80.2 ~ 107.4	125.0 ~ 165.6	
	↕	●	●	●	●			
Command format	No. of steps							
	Conditions			Step				
DIST (d, S, n)	—			4				

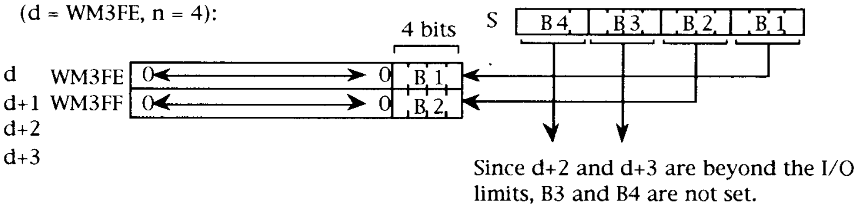
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Top I/O of separation destination							○						
S	I/O to be separaed					○	○	○	○				○	
n	No. of words to be separaed												○	n = 0 to 4

[Function]

S is separated to values of each 4 bits and set in the low-order 4 bits of n words from d.
The high-order 12 bits from d to d+n-1 are 0.
The data of S is held even when the DIST instruction is executed.
When d+n-1 is beyond the I/O limits (WR43FF(WR3FF for H-250), WL03FF, WL13FF, WM3FF), DER (R7F4) is 1 and the separated data of S is set in the low-order 4 bits from d to the I/O range.



When d+n-1 is beyond the I/O limits
(d = WM3FE, n = 4):



[Precautions]

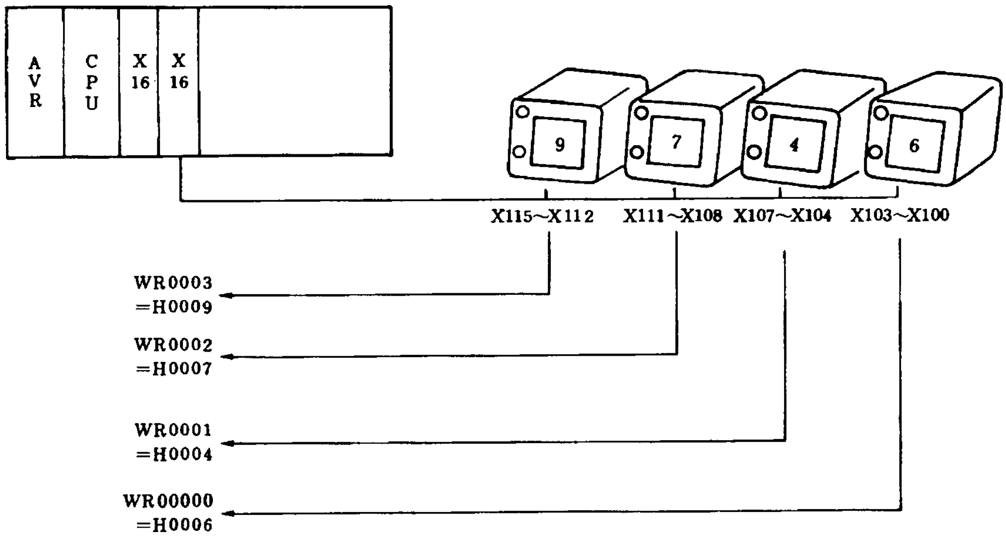
- When n = 0, the instruction is not executed.
- In H-252, the internal output area is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, when WR3FF and WR400~43FF are assigned simultaneously in the processing, DER=1 will be caused and the processing will be executed from d to WR3FF, and be disabled from WR400 thereafter.

[Program example]

```
LD X00001
AND DIFO
[
DIST (WR0000,WX0010,4)
]
```

[Program explanation]

The input of the 4-bit and 4-digit Dight switch is connected to WX0010 and each digit data is stored in WR0000 to WR0003 as independent data.



Name	I/O ADDRESS CONVERSION						
Ladder format	Condition code					Processing time (μ s)	Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	
ADRIO (d, S)	DER	ERR	SD	V	C	20.7	
	●	●	●	●	●		
Command format	No. of steps						
	Conditions			Step			
ADRIO (d, S)	—			3			

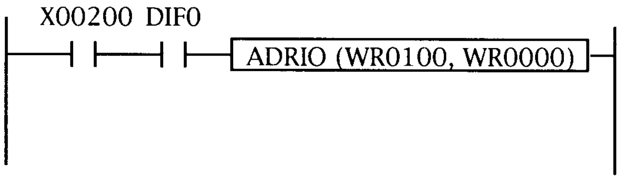
Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Conversion address						○	○						
S	I/O to be converted	○	○	○		○	○	○						

[Function]

Real address from S to d

- The real addres of the I/O specified by S is determined and set in d.

[Program example]



[Program explanation]

- The real address (H4400) of WR0 is set in WR0100 at the rising edge of X00200.
  After the instruction is executed: WR0000 WR0100 H4400

[Precautions]

- This command is effective only for H-252.

3.3 Details of Commands

(1) Basic Commands
(2) Arithmetic Commands
(3) Application Commands
(4) Control Commands
(5) Transfer Commands for Sophisticated Function Module
(6) FUN Commands

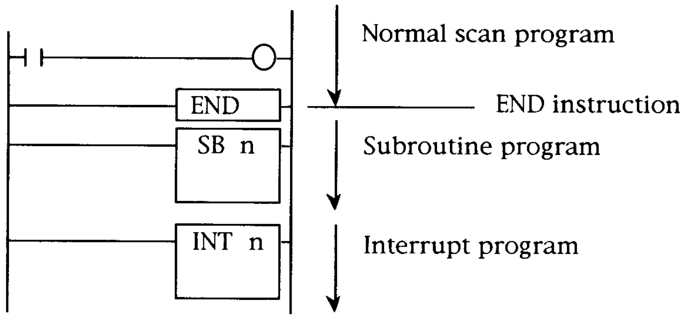
Name	Ending Normal Scan (END)							
Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
END	DER	ERR	SD	V	C	32	180 ~ 188	In the direct mode
	●	●	●	●	●			
Command format	No. of steps							
END	Conditions			Step		1300	1332 ~ 1340	In the refresh mode
	——			1				

Usable I/O	Bit				Word				Double word			Constant	Others
	X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		

[Function]

- This instruction indicates the end of a normal scan program. (When this instruction is executed, control is returned to the top of the program and normal scan is executed.)
- If no subroutine or interrupt scan program is available, this instruction is not required.
- If a subroutine or interrupt scan program is available, code this instruction at the end of a normal scan program.
- Code this instruction only once in a program. Do not enter a start condition.

[Program example]



[Precautions]

- The END instruction is checked before operation starts. If an error is found , one of the following error codes is set in internal special output WRF001. In this case, 34 is set in the internal special output WRF000 as the CPU error code.

CPU error code	Internal special output	Error code	Nature of error	Action
34	WRF001	H0010	No definition for END instruction	Code an END instruction.
		H0022	Two or more END instruction are coded.(area error)	Delete the unnecessary END instruction.
		H0032	A start condition is set in the END instruction.	Delete the start condition.

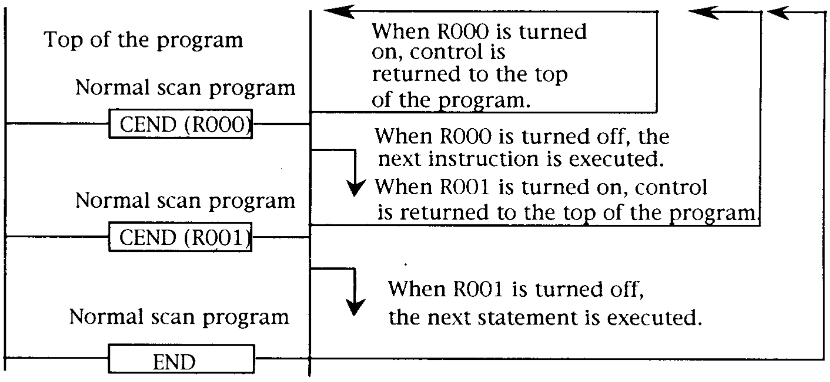
<table>
<tr><td>Name</td><td colspan="8">Conditional End of Scan (CONDITIONAL END)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">CEND (S)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">When the condition is not fulfilled</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">16.6</td><td rowspan="3">29.8</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">CEND (S)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">(Processing speed of END command)+2.8</td><td rowspan="2">(Processing speed of END command)+5.5</td><td rowspan="2">When the condition is fulfilled</td></tr>
<tr><td colspan="3">—</td><td colspan="2">2</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
S	Scan end condition	○	○	○										

[Function]

- When this instruction is executed with scan end condition S satisfied, control is returned to the top of the normal scan program and the program is executed.
- When this instruction is executed with scan end condition S not satisfied, the next instruction is executed.
- This instruction can be executed only in a normal scan program. This instruction can be coded as many times as desired.
- A start condition can be set in this instruction. When both S and the start condition are satisfied, this instruction is executed.

[Program example]



[Precautions]

- The CEND instruction is checked before operation starts. If an error is found , the following error code is set in internal special output WRF001. In this case, 34 is set in the internal special output WRF000 as the CPU error code.

CPU error code	Internal special output	Error code	Nature of error	Action
34	WRF001	H0023	A CEND instruction follows an END instruction	Place a CEND instruction before an END instruction

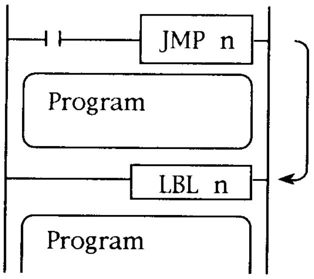
<table>
<tr><td>Name</td><td colspan="8">Unconditional Jump (JUMP)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">JMP n</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">75.6
~
181.6</td><td rowspan="5">103.5
~
261.9</td></tr>
<tr><td>●</td><td>①</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">JMP n</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">2</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>n</td><td>Code No.</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>0 to 255 (decimal)</td></tr>
<tr><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></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]

- When the start condition in JMPn is satisfied, the program is jumped from this instruction to LBLn having the same code number as JMPn. JMPn must be paired with LBLn.
- When the start condition is not satisfied, the next instruction is executed.
- When this instruction is placed together with other instructions in an arithmetic box, this instruction must be the last instruction in the box.
- The effect of this instruction is valid only in the same scan program. (This instruction cannot cause a jump from a normal scan program to a subroutine or interrupt scan program, or vice versa.)
- JMPn instructions can be nested. However, take case so that cycle time over errors are not caused.

[Program example]



- When the start condition is satisfied, a jump to LBLn occurs.
- If the program to which control is jumped has a timer, the elapsed time is updated but no instruction is executed. Therefore, output is disabled even if the ON condition is satisfied.

[Precautions]

- This instruction is checked during execution. If an error is found, one of the following error codes is set in internal special output R7F3 and WRF015. In this case, no Jump occurs and the next program is executed.

<table>
<tr><td colspan="2">Internal special output</td><td>Error code</td><td>Nature of error</td><td>Action</td></tr>
<tr><td rowspan="2">R7F3 = "1"</td><td rowspan="2">WRF015</td><td>H0015</td><td>No LBLn is found</td><td>Define LBLn as a jump destination</td></tr>
<tr><td>H0040</td><td>An attempt was made to jump to another program.</td><td>Correct the program so that a jump to another program area does not occur.</td></tr>
</table>

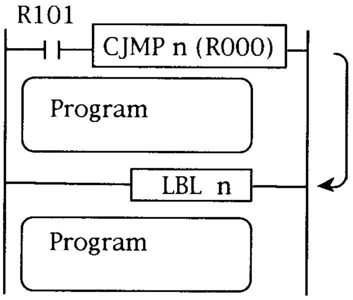
<table>
<tr><td>Name</td><td colspan="8">Conditional Jump (CONDITIONAL JUMP)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">CJMP n (s)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="4">When the condition is not fulfilled</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">15.6</td><td rowspan="3">28.4</td></tr>
<tr><td>●</td><td>①</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">CJMP n (s)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">78.9
~
184.9</td><td rowspan="2">108.4
~
266.8</td><td rowspan="2">When the condition is fulfilled</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>n</td><td>Code No.</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>0 to 255 (decimal)</td></tr>
<tr><td>S</td><td>Jump condition</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></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]

- When jump condition S in CJMPn (S) is satisfied, the program is jumped from this instruction to LBLn having the same code number as CJMPn (S). CJMPn (S) must be paired with LBLn.
- When the start condition or jump condition is not satisfied, the next instruction is exeecuted.
- When placing this instruction together with other instructions in an arithmetic box,take care. If the condition is satisfied, a Jump occurs without the remaining operations in the box being performed.
- The effect of this instruction is valid only in the same scan program. (This instruction cannot cause a jump from a normal scan program to a subroutine or interrupt scan program, or vice versa.)
- CJMPn (S) instructions can be nested. However, take case so that cycle time over error is not caused.

[Program example]



- When both the I/O R000 bit for the start condition and that for the jump condition are satisfied, a jump to LBLn occurs.
- If the program to which control is jumped has a timer, the elapsed time is updated but no instruction is executed. Therefore, the timer coil is not turned on even if the ON condition is satisfied.

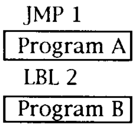
[Precautions]

- This instruction is checked during execution. If an error is found, one of the following error codes is set in internal special output R7F3 and WRF015. In this case, no Jump occurs and the next program is executed.

Internal special output		Error code	Nature of error	Action
R7F3 = "1"	WRF015	H0015	No LBLn is found	Define LBLn as a jump destination
		H0040	An attempt was made to jump to another program.	Correct the program so that a jump to another program area does not occur.

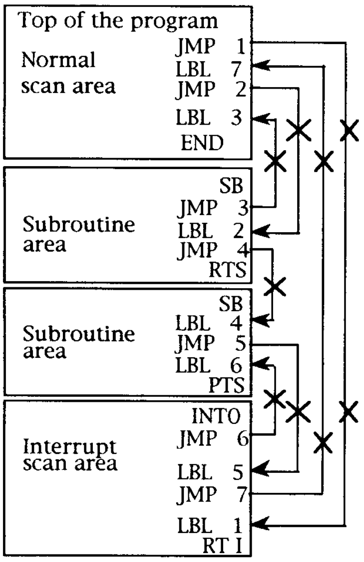
[Syntax of JMP and CJMP]

① LBLn having the same code number (n) as the JMP instruction is required.



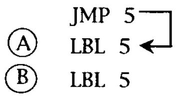
- If JMP1 is executed without LBL1, an error indicating that no LBL is defined is caused.
  When this happens, the next program A is executed without JMP1 being executed.

② A jump beyond the area including the instruction cannot be caused



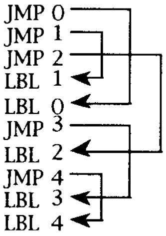
- LBL1 is not in the normal scan area.If the JMP1 instruction is executed, an error indicating an out-of area jump occurs.
  When this happens, the next instruction is executed without the JMP1 instruction being executed.
- JMP2 to JMP7 function in the same way as JMP1.

③ LBLn having the same code number (n) as the JMP instruction must not be duplicated.

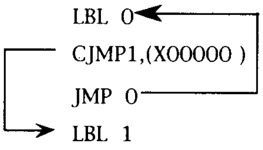


- In processing before operation, label instructions Ⓐ and Ⓑ have the same code number whose value is 5. This causes a duplicated definition error.

④ JMP instructions can be nested.

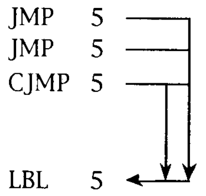


⑤ The JMP instruction can cause a jump to an instruction before it.

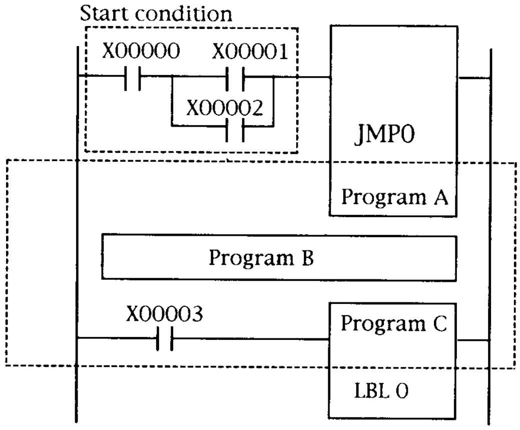


- JMP0 causes a jump to JBL0 placed before it.
- When input X00000 is turned on, control exits from the loop between LBL0 and JMP0 by jumping from CJMP1 (X00000) to LBL1.
- If CJMP1, (X00000) or another instruction that exits control from the loop is not coded,the program loops indefinitely between LBL0 and JMP0.

⑥ JMP instructions having the same code number can be duplicated.



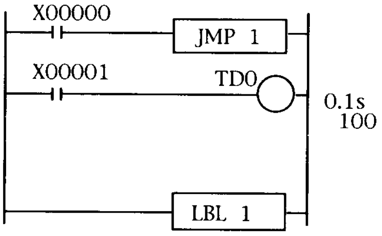
⑦ A start condition can be programmed for a JMP instruction.



- If a jump is made from JMP0 to LBL0, programs A, B, and C are not executed.

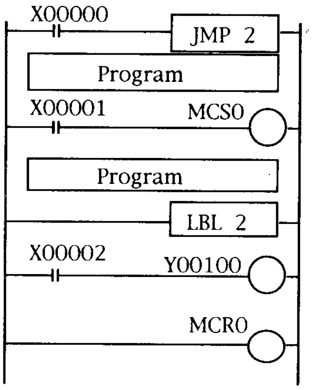
⑧ The CJMP instruction also conforms to the syntax described in ① to ⑦ above.

Note 1: When a JMP instruction causes a jump to LBL, the status of each I/O between JMP and LBL is retained. However, the elapsed time in the timer is updated.



- If X00001 is turned on then X00000 is turned on, the elapsed time in TD0 is updated even if a jump is made from JMP to LBL1.
  If X00000 is left on, TD0 is not turned on even if it exceeds 100.

Note 2: When a JMP instruction is combined with an MCS or MCR instruction, operation is performed as follows. Take care when programming this combination.



- When JMP2 does not cause a jump, Y00100 is turned on with X00001 and X00002 turned on.
- When JMP2 causes a jump and X00000 is turned on, Y00100 conforms to the on-off status of X00002 regardless of whether X00001 is turned on or off.

Note 3: Do not create circuit that causes a jump outside between MCS and MCR.

<table>
<tr><td>Name</td><td colspan="9">Label (LABEL)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">LBL n</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">0.2</td><td rowspan="5">0.6</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">LBL n</td><td colspan="4">Conditions</td><td>Step</td></tr>
<tr><td colspan="4">—</td><td>1</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	Code No.												○	0 to 255 (decimal)

[Function]

- This instruction is a destination to which a jump is made when JMPn or CJMPn(S) is executed. (This number n in LBLn must be the same as that in JMPn or CJMPn(S).)
- The number n in LBLn must not be duplicated in the same program.
- This instruction itself performs nothing.
- Any start condition in LBLn is ignored.

[Precautions]

- This instruction is checked before operation starts. If an error is found, the following error code is set in internal special output WRF001. In this case, 34 is set in the internal special output WRF000 as the CPU error code.

CPU error code	Internal special output	Error code	Nature of error	Action
34	WRF001	H0001	LBL is defined twice	Delete the unnecessary LBLn.

<table>
<tr><td>Name</td><td colspan="8">FOR</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FOR n (S)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">46.5
~
111.7</td><td rowspan="5">60.0
~
157.5</td></tr>
<tr><td>●</td><td>①</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FOR n (S)</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R,
L,
M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR,
WL,
WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR,
DL,
DM</td></tr>
<tr><td>n</td><td>Code No.</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>0 to 49 (decimal)</td></tr>
<tr><td>S</td><td>Repetition count</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></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]

- This instruction is a destination to which a jump is made from NEXTn having the same code number as this instruction.
  When repetition count S is greater than 0, the next instruction after FORn(S) is executed.
  When repetition count S is equal to 0, a jump is made to the next instruction after NEXTn.

- FORn(S) must be paired with NEXTn. NEXTn must be placed after FORn.

- FORn(S) must not be duplicated.

- FORn(S) and NEXTn must be in the same program area. (For example, a program containing FORn(S) in the normal scan area and NEXTn in the subroutine area is not allowed.)

- FORn(S)-NEXTn loops can be nested up to five levels. For details, see NEXTn.

[Precautions]

- This instruction is checked before operation starts. If an error is found , the following error code is set in internal special output register WRF001. In this case, 34 is set in the internal special output WRF000 as the CPU error code.

CPU error code	Internal special output	Error code	Nature of error	Action
34	WRF001	H0002	FOR is defined twice.	Delete the unnecessary FORn(S).

- If an error occurs during execution of this instruction, one of the following error codes is set in internal special output R7F3 and WRF015 and the next instruction is executed.

Internal special output		Error code	Nature of error	Action
R7F3 = " 1"	WRF015	H0017	No NEXT is defined.	Define NEXT corresponding to FORn.
		H0043	Invalid NEXT for FOR	Code NEXTn after FORn.
		H0044	NEXT in an invalid area containing FORn.	Place NEXTn in the area
		H0045	Invalid FOR-NEXT nesting	Use a FOR-NEXT nesting structure.
		H0046	Too many FOR nesting levels	Reduce the number of FOR-NEXT nesting levels to 5 or less.

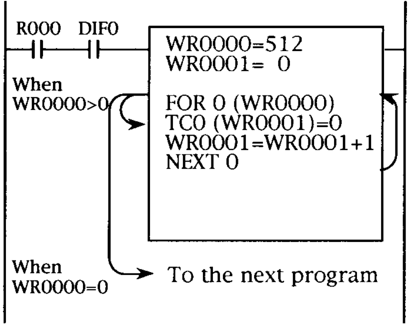
Name	NEXT							
Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
NEXT n	DER	ERR	SD	V	C	59.7 ~ 89.2	79.7 ~ 123.7	
	●	①	●	●	●			
Command format	No. of steps							
	Conditions			Step				
NEXT n	—			2				

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	Code No.												○	0 to 49 (decimal)

[Function]

- This instruction subtracts 1 from repetition count S in FORn(S) having the same code number as this instruction. Then, this instruction causes a jump to FORn(S).

[Program example]



- When R0 is turned on, the value (TCn) in the timer counter is zero-cleared for 512 points.
- Once the FOR-NEXT loop is executed, the execution continues until S becomes 0.
- FOR0 (WR0000) executes instructions below TC0 (WR0001) = 0 while WR0000 > 0. When NEXT0 is reached, 1 is subtracted from WR0000 and a jump is made to FOR0 (WR0000).
- FOR0 (WR0000) causes a jump to the instruction next to this box when WR0000 becomes 0.

[Precautions]

- This instruction is checked before operation starts. If an error is found, the following error code is set in internal special output WRF001. In this case, 34 is set in the internal special output WRF000 as the CPU error code.

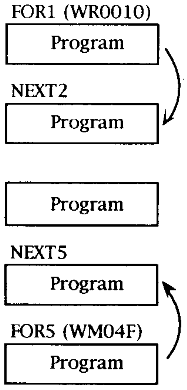
CPU error code	Internal special output	Error code	Nature of error	Action
34	WRF001	H0003	The NEXT instruction is defined twice.	Delete the unnecessary NEXTn.

- If an error occurs during execution of this instruction, one of the following error codes is set in internal special output registers R7F3 and WRF015 and the next program is executed.

Internal special output		Error code	Nature of error	Action
R7F3 = "1"	WRF015	H0016	FOR not defined.	Define FORn corresponding to NEXTn
		H0046	Too many FOR nesting levels	Reduce the number of FOR-NEXT nesting levels to 5 or less.

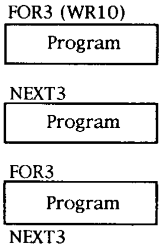
[Syntax of FOR-NEXT]

① A NEXT instruction having the same code number (n) as a FOR instruction is required after the FOR instruction.



- NEXT not defined
  A NEXT instruction corresponding to the FOR instruction is not coded in the user program.
- FOR not defined
  A FOR instruction is not coded before a NEXT instruction.
- Incorrect NEXT-FOR loop
  A NEXT instruction is coded before a FOR instruction.

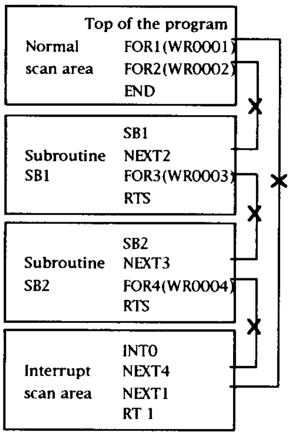
② FOR or NEXT instructions having the same code number (n) must not be duplicated.



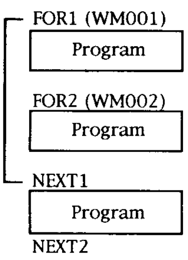
- Double definition error of FOR
  FOR instruction having the same code number (n) are coded.
- Double definition error of NEXT
  NEXT instruction having the same code number (n) are coded.

Note:
Double definition error of FOR, Double definition error of NEXT are occurred before operation starts.

③ FOR and NEXT instructions must be in the same program area.



④ Use a FOR-NEXT nesting structure.



The statements from FOR1 (WM001) to NEXT1 are executed normally.

- Nesting error
  When WM002 =0
  The statements from FOR1 (WM001) to NEXT1 are given priority. Therefore, a jump from FOR2 across NEXT1 to NEXT2 does not occur. NEXT2 causes an error due to an undefined FOR2.
  When WM002 ≠ 0
  FOR2 does nothing. NEXT2 causes an error due to an undefined FOR2.

⑤ A jump instruction can be used to exit from a FOR-NEXT loop.

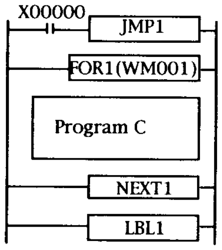
```
┌─ FOR1 (WM001)
│
│  CJMP10(X00000)
└─ NEXT1
   LBL10
```

- When X00000 is turned on before the FOR1-NEXT1 loop is executed by the specified number of repetitions (contens of WM001), the program can exit from the loop.

⑥ Up to five FOR-NEXT loops can be nested. If a subroutine is included, the FOR-NEXT loops in it are also counted.

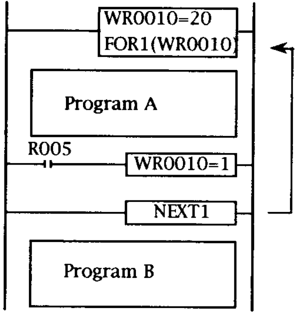
```
┌──────── FOR1 (WR0001)
│┌─────── FOR2 (WR0002)
││┌────── FOR3 (WR0003)
│││┌───── FOR4 (WR0004)
││││┌─── FOR5 (WR0005)
│││││    FOR6 (WR0006)   •Nesting overflow error
│││││
│││││    NEXT6
││││└─── NEXT5
│││└──── NEXT4
││└───── NEXT3
│└────── NEXT2
└─────── NEXT1
```

⑦ Do not include start conditions in a FOR-NEXT loop. (When a starting condition is necessary, make a circuit as shown below.)



[Explanation]
When X00000 is off, program C is executed by the number specified by WM1.
When X00000 is on , a jump occurs from JMP1 to LBL1. Program C is not executed.

⑧ The repetition count can be changed in the program.



The contents of WR0010 are subtracted by one and a jump is made to FOR1 (WR0010).

- When R005 is off
  After program A is executed 20 times, program B is executed.

- When R005 is on
  The number of repetitions indicated by WR0010 becomes 1 and 1 is subtracted in processing of NEXT1. So the contents of WR0010 become 0. This terminates repetition of program A and executes program B.

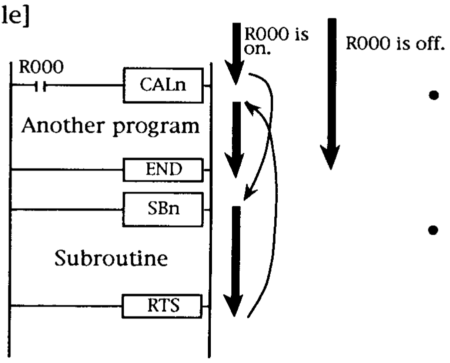
Name	Subroutine Call (CALL)							
Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
CAL n	DER	ERR	SD	V	C	57.7 ~ 72.9	76.6 ~ 99.4	
	●	①	●	●	●			
Command format	No. of steps							
	Conditions			Step				
CAL n	—			2				

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	Code No.												○	0 to 99 (decimal)

[Function]

- When the start condition in the CALn instruction is satisfied, the subroutine having the same code number (program bracketed by SBn and RTS) as the instruction is executed.
- When the start condition is not satisfied, the next program is executed.
- Up to five other nestd subroutines can be called from a subroutine. (The maximum nesting level is 5.)
- This instruction can also call a subroutine in the interrupt scan program.

[Program example]



- When R000 is on, CALn executes a subroutine.
  After the subroutine has been executed, the next program after CALn is executed.
- When R000 is off, the next program is executed without subroutine program being executed.

[Precautions]

- If an error is found during execution of this instruction, one of the following error codes is set in internal special output registers R7F3 and WRF015. In this case, the next program is executed.

Internal special output		Error code	Nature of error	Action
R7F3 = "1"	WRF015	H0013	SB not defined.	Define an SBn-RTS pair for CALn.
		H0041	Too many nesting levels	Reduce the number of subrou tine nesting levels to 5 or less.

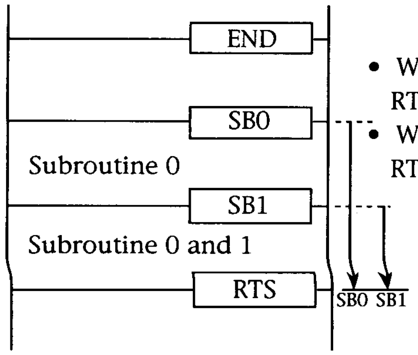
Name	Subroutine Start (SUBROUTINE)							
Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
SB n	DER	ERR	SD	V	C	0.2	0.6	
	●	●	●	●	●			
Command format	No. of steps							
	Conditions			Step				
SB n	—			1				

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
n	Code No.												○	0 to 99 (decimal)

[Function]

- This instruction indicates the beginning of a subroutine. (This statement processes nothing.)
- The number n in SBn must not be duplicated in the same program.
- Any start condition in SBn is ignored.
- SBn must be paired with RTS.
- Place the subroutines from SBn to RTS after an END instruction.

[Program example]



- When CAL0 is executed, the program from SB0 to RTS is executed as a subroutine.
- When CAL1 is executed, the program from SB1 to RTS is executed as a subroutine.

[Precautuions]

- This instruction is checked before operation starts. If an error is found, one the following error codes is set in internal special output WRF001. In this case, 34 is set in the internal special output WRF000 as the CPU error code.

CPU error code	Internal special output	Error code	Nature of error	Action
34	WRF001	H0004	Double definition of SB.	Delete the unnecessary SBn.
		H0013	No definition of SB.	Define SBn corresponding to RTS.

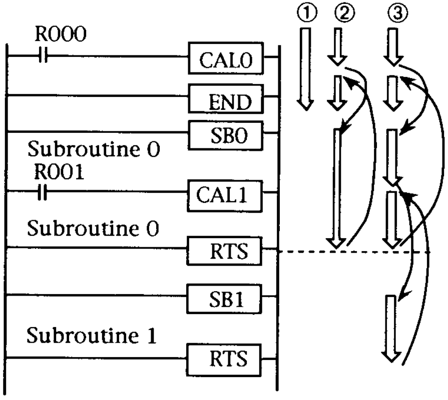
<table>
<tr><td>Name</td><td colspan="8">Subroutine End (RETURN SUBROUTINE)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">RTS</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">40.2
~
55.7</td><td rowspan="5">50.5
~
73.7</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">RTS</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">1</td></tr>
</table>

Usable I/O	Bit				Word				Double word			Constant	Others
	X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		

[Function]

- This instruction indicutes the end of a subroutine.
- When this instruction executed, the next program after the CALn instruction that called a subroutine is executed.
- Any start condition must not be entered in this instruction.

[Proram example]



① Program execution when both R000 and R001 are off.

② Program execution when R000 is on but R001 is off.
CAL0 is executed and control is passed to subroutine 0.
CAL1 is not executed.When subroutine 0 is terminated, control is returned to the next program after CAL0.

③ Program execution when both R000 and R001are on.
CAL0 is executed and control is passed to subroutine 0.
CAL1 is executed and control is passed to subroutine 1.
When subroutine 1 is terminated, control is returned to the next program after CAL1.
When subroutine 0 is terminated, control is returned to the next program after CAL0.

[Precautions]

- This instruction is checked before operation starts. If an error is found, one of the following error codes is set in internal special output WRF001. In this case, 34 is set in the internal special output WRF000 as the CPU error code.

CPU error code	Internal special output	Error code	Nature of error	Action
34	WRF001	H0011	RTS is not defiend.	Difine RTS corresponding to SBn
		H0020	RTS in an invalid area	Place RTS after END. RTS cannot be used in the interrupt program.
		H0030	Invalid start condition in RTS	Do not enter any startcondition in RTS.

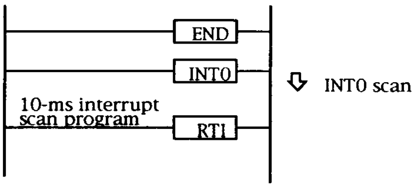
<table>
<tr><td>Name</td><td colspan="9">Starting Interrupt Scan Program (INTERRUPT)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">INT n</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">173
~
229</td><td rowspan="5">540
~
560</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">INT n</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">1</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>n</td><td>Interrupt priority</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>0 to 2</td></tr>
<tr><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></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]

- This instruction indicates the beginning of an interrupt scan program.
- A smaller value of n indicates a higher interruput priority.
- INTn must be paired with RTI.
- Any start condition in INTn is ignored.
- Place the interrupt scan program from INTn to RTI after an END instruction.
- The number n in INTn must not be duplicated in the same program.

[Program example]



- The program from INT0 to RTI is started and executed at 10 ms intervals.

[Precautions]

- This instruction is checked before operation starts. If an error is found, one of the following error codes is set in internal special output WRF001. In the case, 34 is set in the internal special output WRF000 as the CPU error code.

CPU error code	Internal special output	Error code	Nature of error	Action
34	WRF001	H0005	Double definition of INT.	Delete the unnecessary INTn.
		H0014	INT is not defined.	Delete INTn corresponding RTI.

Name	Ending Interrupt Scan Program (RETURN INTERRUPT)							
Ladder format	Condition code					Processing time (μ s)		Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	H-250	
RTI	DER	ERR	SD	V	C	74 ~ 162	360 ~ 370	
	●	●	●	●	●			

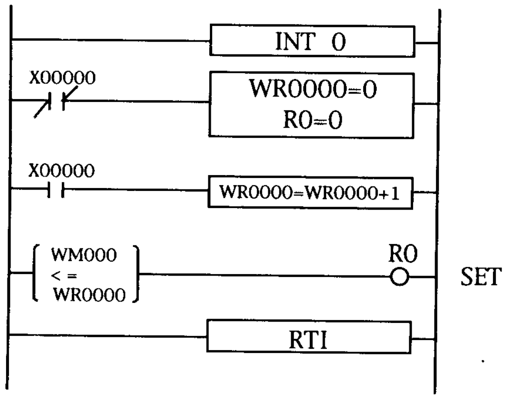
Command format	No. of steps	
	Conditions	Step
RTI	——	1

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		

[Function]

- This instruction indicates the end of an interrupt scan program.
- When this instruction is executed, control is returned to the program which was being executed before interrupt scan was performed.
- Any start condition must not be entered in this instruction.

[Program example]



- 0.01 - s timer is created using interrupts generated at 10 ms intervals. The setting is set in WM000. The elapsed time is set in WR0000. R0 is used as a timer coil.
- When X00000 is off, the elapsed time and timer coil are cleared.
- When X00000 is on, the elapsed time is incremented by one at 10 ms intervals.
- When WM000 < = WR0000, the timer coil is turned on.

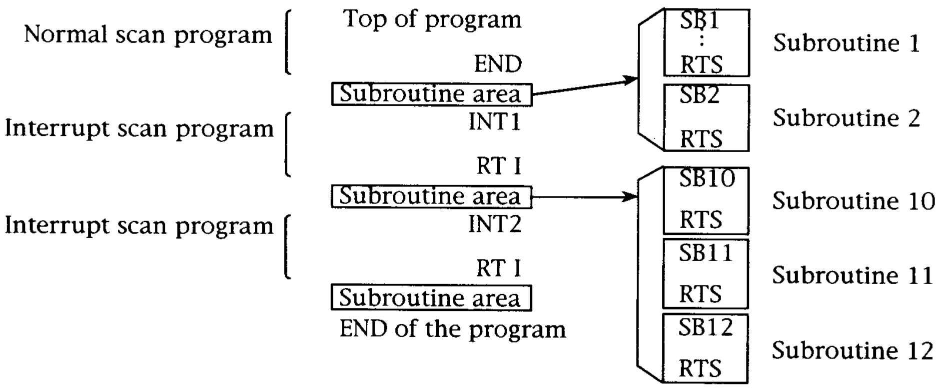
[Precautions]

- This instruction is checked before operation starts. If an error is found, one of the following error codes is set in internal special output WRF001. In this case, 34 is set in the internal output WRF000 as the CPU error code.

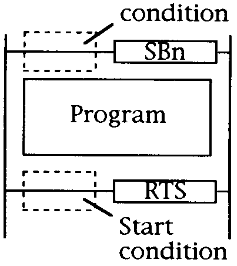
CPU error code	Internal special output	Error code	Nature of error	Action
34	WRF001	H0012	RTI is not defiend.	Difine RTI corresponding to INTn
		H0021	RTI in an invalid area	Place RTI after END. RTI cannot be used in SBn
		H0031	Invalid start condition in RTI	Do not enter any startcondition in RTI

[Syntax of SBn, RTS, INTn and RTI]

① Subroutines can be placed between a normal scan program and interrupt scan program, between interrupt scan programs, or after the last interrupt scan program.

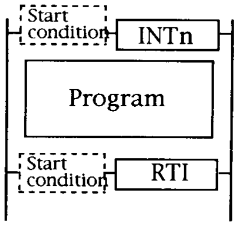


② The Subroutine Start (SBn) and Subroutine End (RTS) instructions must not contain any start condition.

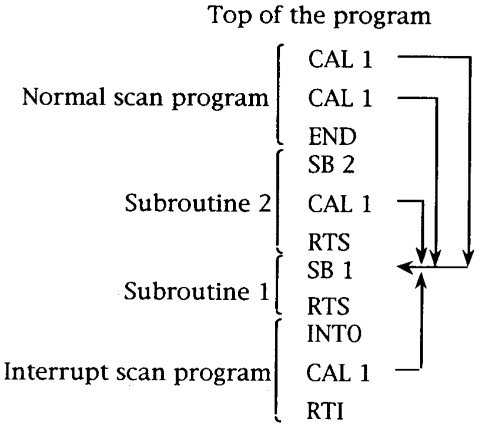


- Since a start condition is set in RTS, an error occurs in processing before operation.

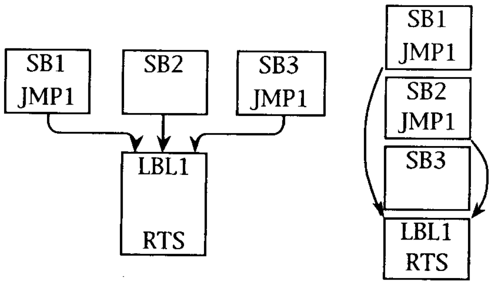
③ The Starting Interrupt Scan (INTn) and Ending Interrupt Scan (RTI) instructions must not contain any start condition.



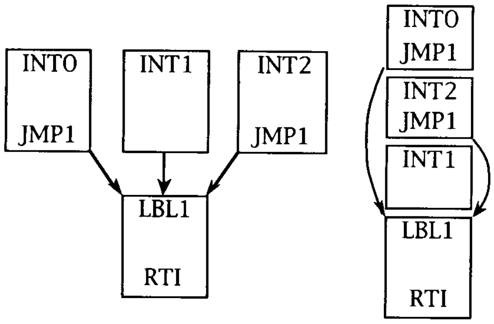
④ A same subroutine can be called from the normal scan program , the interrupt scan program, or another subroutine.



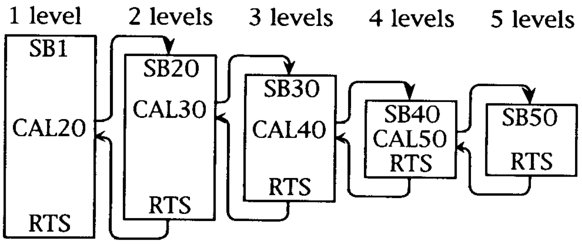
⑤ Subroutines having many entries and one exit can also be coded.



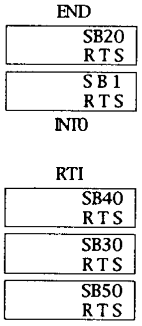
⑥ Interruptscan having many entries and one exit can also be coded.



⑦ Subroutines can be nested at up to five levels.



Top of the program



(1) There is no relationship between the order of subroutines and the nesting order, as shown on the left.

3.3 Details of Commands

(1) Basic Commands	
(2) Arithmetic Commands	
(3) Application Commands	
(4) Control Commands	
(5) Transfer Commands for Sophisticated Function Module	■
(6) FUN Commands	

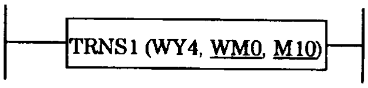
Name	SIH COMMUNICATION COMMAND

Ladder format	Condition code					Processing time (μ s)	Remarks
	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	
TRNS1(d,s,t)	DER	ERR	SD	V	C	271.0~378.9	During the communication
	↕	●	●	●	●		
Command format	No. of steps						
	Conditions			Step		806.5~931.8	At the start of the communication
TRNS1(d,s,t)	—			5			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Module installation location						○							
s	Top of parameter area							○						S : up to s + 16
t	Top of communications control bit area			○										t : up to t + 5

[Precautions]

- Array variables cannot be used in d, s and t.
- Use s + 16 and t + 5 so that they do not exceed the I/O range (R7FF, L03FFF, L13FFF, M3FFF, WR43FF, WL3FF, WL13FF, WM3FF). When they exceed the range, then DER is 1, and communications are not executed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.
- When the bit word commonly usable I/O M and L are used for t parameter by the TRNS1(d,s,t) instruction, the overlap check of t area of the bit and s area of the word is not correctly done. Therefore please pay attention not be overlapped s and t areas in the program of TRNS1 instruction.



Program example wrong that s area (s+1) and t area overlapped.

[Function]

1. This is a communication command for high function modules (Serial I/O SIH) and the CPU. This command is used to transmit the command and related data from the CPU, then receive the response data.
2. The smallest WY number of the I/O assignment WY for the installation location of SIH which communicates with this command is set in d. (Designate WY∗∗∗4.)
3. The top I/O numbers of the parameter area which sets the various parameters (time out value, top I/O number and size of the transmission and reception data area.)for communications is set in s.

4. The control bit which starts and initializes communications, and top I/O No. of the communication control bit area in which the results of communications will be set in t.

5. Description of the S area

S	(1) Return code
S+1	(2) Status
S+2	▨ (3) System area (not usable by users)
S+10	(4) Time out time
S+11	(5) Top I/O of the transmission data area
S+13	(6) Transmission data area size
S+14	(7) Top I/O of the reception data area
S+16	(8) Reception data area size

▨ : User access prohibit area
☐ : User setting area

(1) Return code:

The execution results of the TRNS1 command are set. (Refer to Table 3.3)

Normal end → 0

Abnormal end → ≠ 0

(2) Status:

The contents of the status words (WX∗∗∗0) of the SIH modules are set. For details of errors, refer to this area.

(3) System area:

When the TRNS1 command is executed, this area is used for system processing. <u>This area cannot be used by the user.</u>

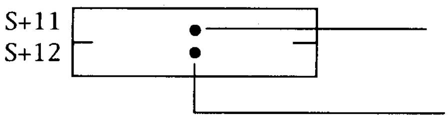
(4) Time out time:

This sets the time out time from the top of the TRNS1 command execution to its completion.

= 0: No time out checking.

≠ 0: × 10 ms time out check executed. (Maximum HFFFF).

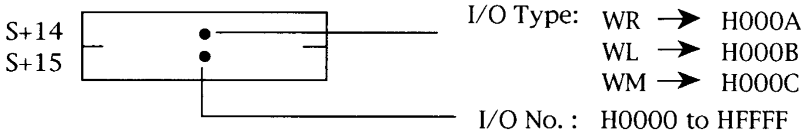
(5) Top I/O of the transmission data area: This designates top I/O type and number of the area holding the transmission data which is sent by the TRNS1 command.



I/O Type: WR → H000A
WL → H000B
WM → H000C
I/O No. : H0000 to HFFFF

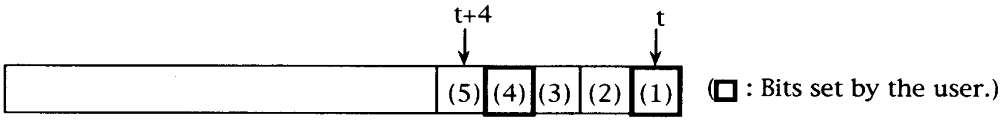
(6) Transmission data area size: This designates the size of the transmission data area in word units.

(7) Top I/O of the reception data area: This designates the top I/O type and number of the area which stores the corresponding response data, after the command and data are transmitted.



(8) Reception data area size: This designates the size of the reception data area in word units.

6. Description of t area



(1) Communication execution:

When communication is executed with the TRNS1 command, the user program should set "1." When communication is over, reset "0" by the command TRNS1.

(2) Normal end:

When communications with the TRNS1 command ends normally, "1" is set. When communication is started, this bit is reset to "0" by the command TRNS1.

(3) Abnormal end:

When communications with the TRNS1 command ends in an error or there is an error in a parameter, "1" is set. When communication is started, this bit is reset to "0" by the command TRNS1.

(4) Initialization request:

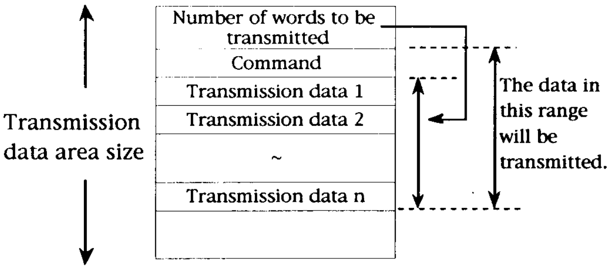
When the TRNS1 command is set to the initial state, this is set to "1". (This is not an initialization of the sophisticated function module).

(5) Initialization end:

When initialization of the TRNS1 command ends normally, this bit is set to "1." At this time the initialization request (4) is reset "0".

An initialization of the TRNS1 command will be executed also at a start of operations. Therefore, "1" will be maintained immediately after a start up.

8. Description of the transmission data area.
   This is the area which stores the command and data which is transmitted by the TRNS1 command. Set the data to be transmitted in SIH according to the configuration shown.



9. Description of the reception data area.

   This is the area for the reception data, to the command and data which is transmitted by the TRNS1 command. This is set according to the configuration shown.

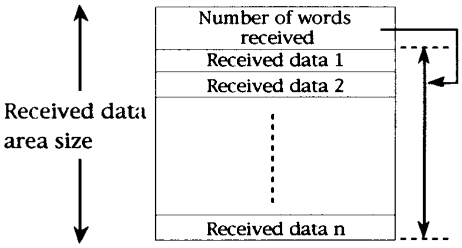


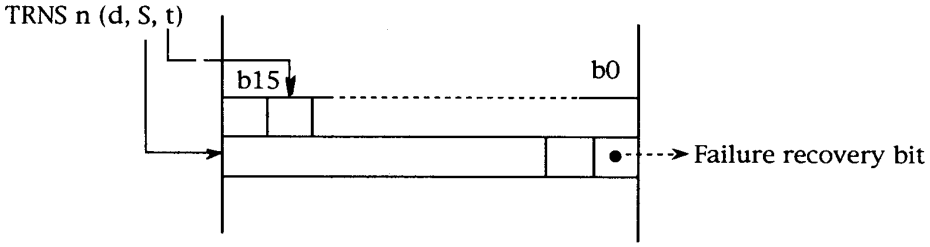
Table 3.3 Return Code Table

Return code	Name	Contents	Action
H00	Normal end	Transmission/reception has ended normally.	
H10	Module error * 1	Watchdog timer error	Check the fixture of the CPU to basic base, turn off/on the power supply, or replace the module.
H21	Range check error	The s parameter and t parameter of the TRNS and RECV commands exceed the I/O range (R7FF, L03FFF, L13FFF, M3FFF, W43FF, WL3FF, WL13FF, WM3FF). *3	Set the parameters to their correct ranges.
H22	Transmission area setting error	The setting for the top of the transmission area (type or corresponding I/O number) is not correct.	Set the top of the transmission area so that it is within the I/O range.
H23	Transmission area range error	The bottom area of the transmission area exceeds the I/O range (WR43FF, WL3FF,, WL13FF, WM3FF). *3	Set the transmission area so that is fits into the correct I/O range.
H24	Reception area setting error	The setting for the top of the reception area (type or corresponding I/O number) is not correct.	Set the top of the reception area so that it is within the I/O range.
H25	Reception area range error	The bottom area of the reception area exceeds the I/O range (WR43FF, WL3FF, WL13FF, WM3FF). *3	Set the the reception area so that it is fits into the correct I/O range.
H26	Transmission data length setting error	The transmission data length setting is longer than the transmission area length.	Set the transmission data length, so that it is within the transmission area range.
H27	Reception data length setting error	The reception data length is longer than the reception area lentgh. Or, a reception area which is not as long as necessary is not secured.	Set the reception data length so that it is within the reception area range.
H28	Area overlap error * 2	There are overlapping areas within the s parameters, t parameters, transmission areas, or reception areas of the TRNS commands.	Set the parameters or transmission /reception areas so that they do not overlap.
H30	Time out *1	The transmission/reception processes of the TRNS command was not completed within the specified time.	Increase the specified time, or modify the process to make the execution to be completed within the specified time
H40	Reception area full	The reception data has filled the data area, and there is no more open area to store the next reception data.	Increase the size of the reception area.
H80	Module unique error	The high function module has received transmission data and detected an error.	Refer to the instruction manual of the module.
H81	Logical discrepancy	System is in logical discrepancy. Meaningless command is sent.	Delete Write in the system area. Check for troubles in noise or hardware.
H82	Abnormal command	The command which can not be interpreted is sent.	Change the program so as to send the correct command.

*1: As the I/F specifications of the sophisticated function modules in terms of TRNS are the same over each module, the module types can not be distinguished in this respect. Therefore, an error code that would tell a mismatch in such combination of a command and a module will not be given. Instead, a time-out error (when a time-out is specified) or a module error will be given by an error code, if the communication is disabled because of the mismatches in register structures or in positions of hand shake bits.

Such error can not be detected in some combinations of commands and module types, where the register structures or positions of hand shake bits are the same. These relationship should be taken into considerations.

*2: Though the return code is H28 for the area overlap error, the area overlap as shown below may cause the return code to fail generate as H28. This should be taken into considerations.



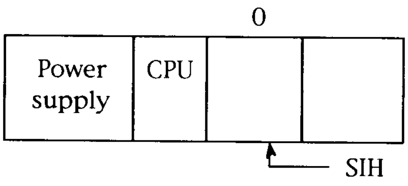
When the top area of S and the area of t are overlapped, H28 will be set on the top area of S, but the error ending bit will be "1". Therefore, the setting will be appeared to be H29.

*3: When the area exceeds WR3FF, an error will be caused too.

[Programming Example]

In this programming example, the serial I/O module (SIH) is used to output data from an external device to another external device. (Non-procedural mode)

1. SIH installation



SIH is installed in slot 0 of the basic base. Therefore, the I/O assignment of SIH is WX0000 to 3 and WY0004 to 7.

Designate WY0004 as the d parameter of the TRNS1 command.

2. Internal output assignment

The programming example will use the following assignments. In actual applications, be sure to modify the I/O numbers, etc.

I/O	No.	Purpose	Remarks
WM	010 to 020	TRNS1 command parameter area (s to s+16)	
M	0000 to 0004	TRNS1 command communication control bit area (t to t+4)	
WR	0100 to 011F	Transmission data area (32 words)	WR0100: Number of transmission data WR0101: Command code WR0102: Command ~ execution data
	0200 to 021F	Reception data area (32 words)	WR0200: Number of reception data WR0201: Reception data ~
R	000	TRNS1 command initialization request	
	001	Operation mode setting request	
	002	Data intput request	
	003	Request to calculate input data	
	004	Data output request	
	100	Communications request (TRNS1 execution)	

3. Data to be passed from the CPU to SIH

In regard to programming example , the command and data which are to be set in the transmission data area as the data to be passed from the CPU to SIH are shown below. Refer to the SIH manual for details.

(a) Initialization process command and data

I/O No.	Data	Contents	Remarks
WR0100	H0000	Number of transmission data	
WR0101	H0000	Command code	Initialization command

(b) Operation mode setting processor command and data

I/O No.	Data	Contents	Remarks
WR0100	H0004	Number of transmission data	
WR0101	H0001	Command code	Operation mode setting command
WR0102	H0000	Time out	No time out setting
WR0103	H0009	Mode setting	After executing the output command, clear the buffer.
WR0104	H0200	Output buffer empty flag operation conditiion	Controlled by the buffer empty byte number (H0200)
WR0105	H0001	Input buffer valid flag operation condition	Controlled by the received message length

(c) Input buffer command

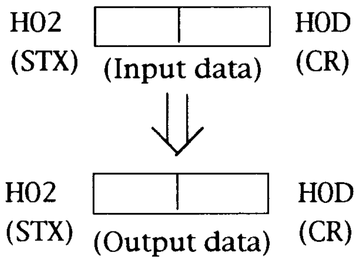
I/O No.	Data	Contents	Remarks
WR0100	H0002	Number of transmission data	
WR0101	H0010	Command code	Data inputs of the input buffer
WR0102	HF001	Message type	Message configuration
WR0103	H020D	Message configuration	STX.....CR

(d) Input buffer command

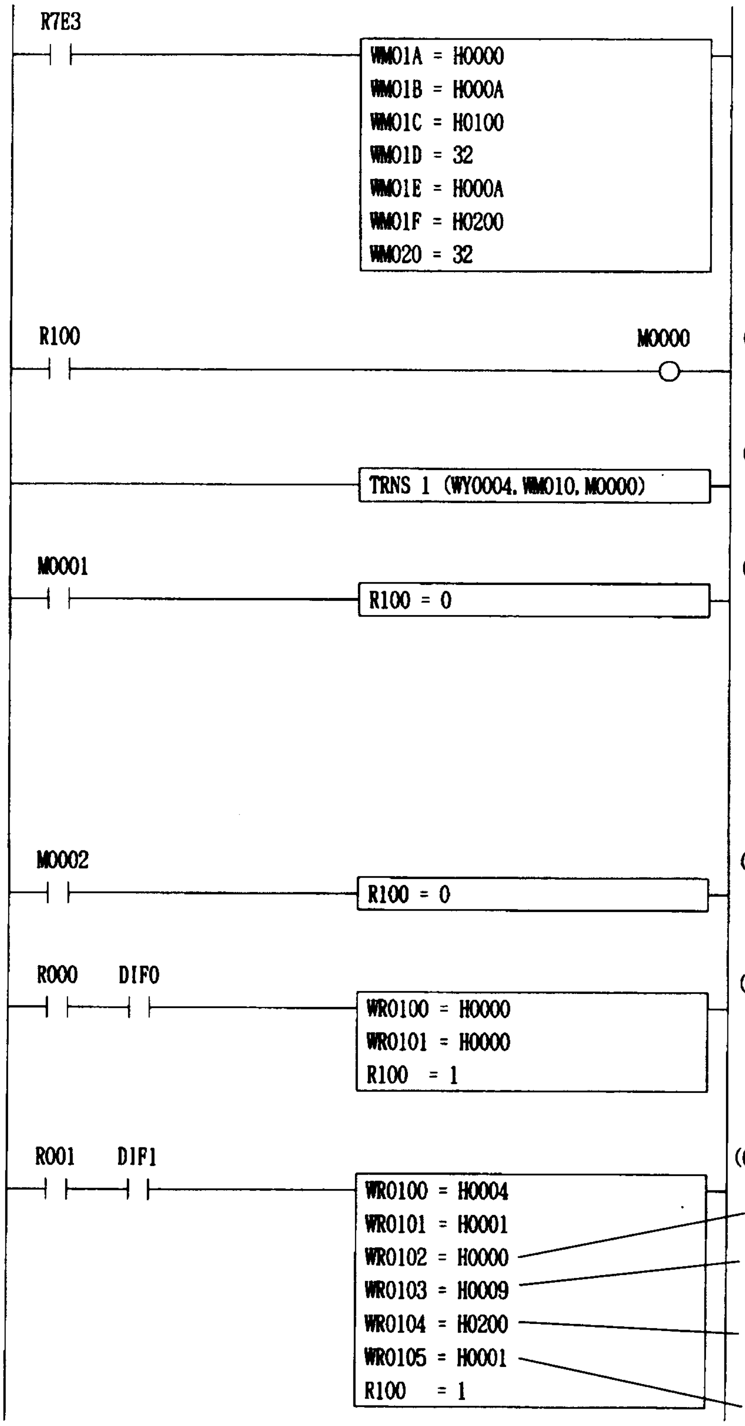
I/O No.	Data	Contents	Remarks
WR0100	H0004	Number of transmission data	
WR0101	H0020	Command code	Data ouput of the output buffer
WR0102	H0004	Number of data	4 bytes (2 words)
WR0103	H020D	Message configuration	STX.....CR
WR0104	WR201	Transmission data (for 1 word)	
WR0105	WR202	Transmission data (for 2 word)	

4. Description of operations.

Data consisting of 2 words from the serial I/O module will be transmited to an external device.



[Programm example]



```
WM01A = H0000
WM01B = H000A
WM01C = H0100
WM01D = 32
WM01E = H000A
WM01F = H0200
WM020 = 32
```

(00001) TRNS1 parameter setting.
No time out.
Specifies the transmission data I/O, WR.
Specifies the transmission data No., H100 ~ .
Transmission data size, 32 words.
Specifies the reception data I/O, WR.
Specifies the reception data No., H200 ~ .
Reception data size, 32 words.

(00002)
SET

```
TRNS 1 (WY0004.WM010.M0000)
```

(00003)

```
R100 = 0
```

(00004) Executes normally

```
R100 = 0
```

(00005) Abnormal end processing.

```
WR0100 = H0000
WR0101 = H0000
R100   = 1
```

(00006) Initialization processing.

```
WR0100 = H0004
WR0101 = H0001
WR0102 = H0000
WR0103 = H0009
WR0104 = H0200
WR0105 = H0001
R100   = 1
```

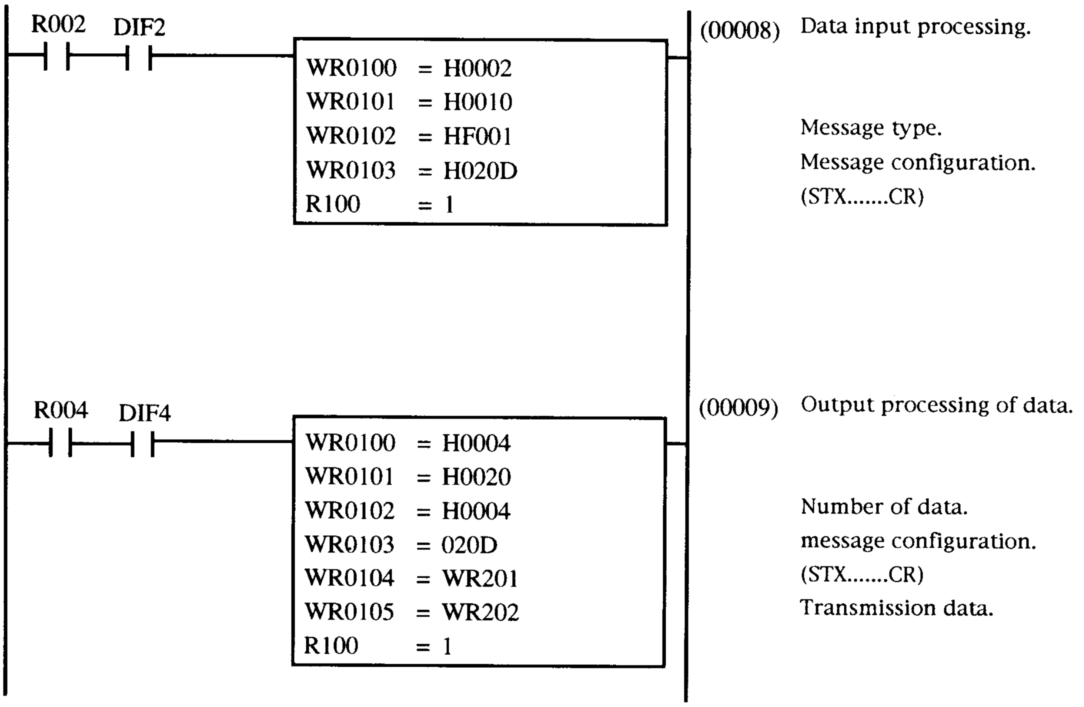
(00007) Operation mode setting processing.

No timeout setting

Clears the buffer after the ouput command execution.

The condition for the empty flag operation for an output buffer.

The condition for the effective flag operation for an input buffer.



```
R002  DIF2
─┤├──┤├──── WR0100 = H0002        (00008) Data input processing.
            WR0101 = H0010
            WR0102 = HF001                Message type.
            WR0103 = H020D                Message configuration.
            R100   = 1                    (STX.......CR)

R004  DIF4
─┤├──┤├──── WR0100 = H0004        (00009) Output processing of data.
            WR0101 = H0020
            WR0102 = H0004                Number of data.
            WR0103 = 020D                 message configuration.
            WR0104 = WR201                (STX.......CR)
            WR0105 = WR202                Transmission data.
            R100   = 1
```

[Program Description]

1. Special internal output R7E3 (1 scan on after RUN is started) to set the TRNS1 command parameters.

   No time out check.

   Transmission data area; WR0100 ~ WR011F. Reception data area; WR0200 ~ WR021F.

2. When the communications start condition R100 is set, communication execution flag M0000 will be ON.

3. Communications will be executed with the serial I/O module.

4. When the normal end flag M0001 is set, normal end processing will be executed and the communication start condition R100 will be cleared.

5. When the abnormal end flag is set, abnormal end processing will be executed and the communication start condition R100 will be cleared.

6. When the initialization start condition R000 is set, the initialization command of the serial I/O module is set in the transmission data area, and the communication start condition R100 is turned ON.

7. When the operation mode setting condition R001 is set, the operation mode setting command of the serial I/O module and its subcommand are set in the transmission data area, then communication start condition R100 is turned ON.

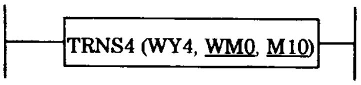
8. When data input condition R002 is set, the input buffer command of the serial I/O module and its subcommand are set in the transmission data area, then communication start condition R100 is turned ON.
9. When data output condition R004 is set, the output buffer command of the serial I/O module and its subcommand are set in the transmission data area, then communication start condition R100 is turned on.

<table>
<tr><td>Name</td><td colspan="7">Send/Receive Command for Two-axis Positioning Module (POSH)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">TRNS4 (d,s,t)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="4">During the communication</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="3">286.9~315.8</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">TRNS4 (d,s,t)</td><td colspan="3">Conditions</td><td colspan="2">Step</td><td rowspan="2">690.1~990.8</td><td rowspan="2">At the start of the communication</td></tr>
<tr><td colspan="3">—</td><td colspan="2">5</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
d	Module installation location						○							
s	Top of parameter area							○						S : up to s+16
t	Top of communications control bit area			○										t : up to t + 5

[Precautions]

- Use s + 16 and t + 5 so that they do not exceed the I/O range (R7FF, LO3FFF, L13FFF, M3FFF, WR43FF (note 1), WL3FF, WL13FF, WM3FF). If they do exceed the range, then DER = 1 will be true, and communications will not be executed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.
- When the bit word commonly usable I/O M and L are used for t parameter by the TRNS4(d,s,t) instruction, the overlap check of t area of the bit and s area of the word is not correctly done. Therefore please pay attention not be overlapped s and t areas in the program of TRNS4 instruction.


TRNS4 (WY4, WM0, M10)

Program example wrong that s area (s+1) and t area overlapped.

[Function]

1. This command is used between the two axis positioning module (POSH) and the CPU. This command is used to send the command and related data from the CPU, then receive the response data.
2. The smallest WY number of the I/O assignment WY for the installation location of POSH which communicates with this command is set in d. (Designate WY∗∗∗4.)
3. The top I/O numbers of the parameter area which sets the various parameters (time out value, top I/O number and size of the transmission and reception data area, control data) for communications is set in s.
4. The control bit which starts and initializes communications, and top I/O No. of the communication control bit area in which the results of communications will be set in t.

5. Description of the area S.

S	(1) Return code
S+1	(2) Status
S+2	(3) Status
S+3	▨ (4) System area (Not usable by the user.)
S+9	(5) Time out time
S+10	(6) Top I/O of the transmission area
S+12	(7) Transmission area size
S+13	(8) Top I/O of the reception area
S+15	(9) Reception area size
S+16	(10) Control data

(▨: User access prohibited area
☐: User setting area)

(1) Return code:

The execution results of the TRNS4 command are set.

Normal end → 0

Abnormal end → ≠ 0 (See TRNS1 for details.)

(2), (3) Status 0, 1:

The contents of the status words (WX∗∗∗0, WX∗∗∗1) of POSH are set.

(4) System area:

When the TRNS4 command is executed, this area is used for system processing. <u>This area cannot be used by the user.</u>

(5) Time out time:

This sets the time out time from the top of the TRNS4 command execution to its completion.

= 0: No time out checking.

≠ 0: x 10 ms time out check executed.
(Maximum, HFFFF).

(6) Top I/O of the transmission data area: This designates top I/O type and number of the area holding the transmission data when the TRNS4 command is sent. (Note)

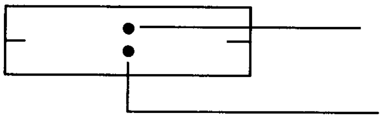
(7) Transmission data area size: This designates the size of the transmission data area in word units.

(8) Top I/O of the reception data area: This designates top I/O type and number of the area holding the reception data when the TRNS4 command is sent. (Note)

(9) Reception area size: This designates the size of the reception data area in word units.

(10) Control data: This sets the command for the two-axis positioning module. Refer to the two-axis positioning module manual or instruction manual for details on the command.

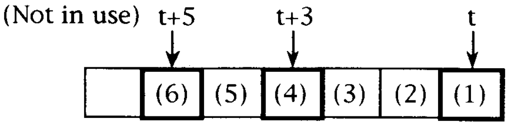
(Note)



I/O Type: WR → H000A
WL → H000B
WM → H000C

I/O No. : H0000 to HFFFF

6. Description of area t.



(□: Bits set by the user.)

(1) Communication execution: When communication is executed with the TRNS4 command, the user program should set to "1". When communication is ended, the TRNS4 command should be reset to "0".

(2) Normal end: When communications with the TRNS4 command ends normally, "1" is set. When communication is started, this bit is reset to "0".

(3) Abnormal end: When communications with the TRNS4 command ends in an error or there is an error in a parameter, this is set to "1." When communication is started, this bit is reset to "0."

(4) Initialization request: When the TRNS4 command is set to the initial state, this is set to "1". (This is not an initialization of the sophisticated function module.)

At a start of operation, the command TRNS4 will execute the initialization (to clear the system area, address calculations for transmission and reception areas) in reference to the special internal output R7E3 (one scan ON at a start up). Use this bit of the initialization request in the case of an abnormal operation as may be caused by an unexpected forced set to "0" in (1) Communication execution or by rewriting the system area of S, during the execution of the command TRNS4.

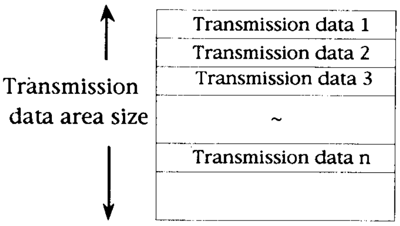
(5) Initialization end: When initialization of the TRNS4 command ends normally, this bit is set to "1." At this time, The initialization request, (4) will be reset to "0".

(6) Not in use: This bit is not used in this command.

7. Description of the transmission data area.

This area stores the data to be transmitted by the command TRNS4. Set the data to be sent to the sophisticated function modules in accordance with the structure shown.

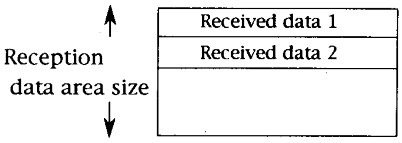
Set data in this area before making comunication executing bit (t) ON.



8. Description of the reception data area size.

This area stores the data corresponding to the command and data sent by TRNS4, in accordance with the structure shown.

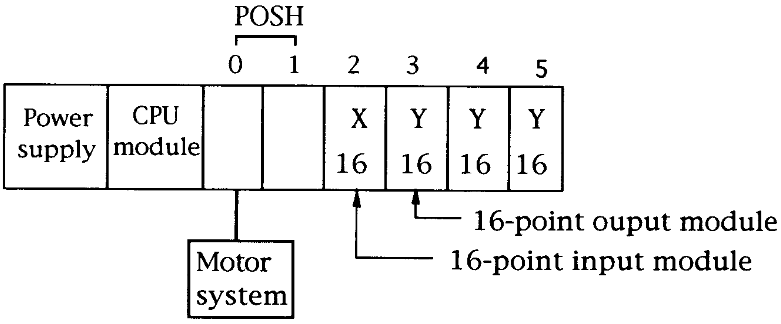
Use data in this area after the normally executed bit or abnormally executed bit is ON



[Programming Example]

This is a programming example starting from an initialization and ending in a pre-determined operation by utilizing a two axis positioning module (POSH).

(1) POSH installation



POSH is installed in slot 0 ~ 1 of the basic base. Therefore, the I/O assingments for POSH are WX0 to WX3 and WY4 to WY7 is designated for the parameter d of TRNS4.

(2) Internal output assignment

The programming example will use the following assignments. In actual applications, be sure to modify the I/O numbers, etc.

I/O	No.	Purpose	Remarks
WM	0 ~ 10	TRNS4 command parameter area(s to s+16)	
R	0000 ~ 0004	TRNS4 command communication control bit area (t to t+5)	
WR	100 ~ 10F	TRNS4 command transmission data area (16 words)	
WR	110 ~ 11F	TRNS4 command reception data area (16 words)	
WR	0	Variable for status control	Used for generationg continuous command.

(3) Assignment of external outputs

Design a program with the following assignment. In actual applications, be sure to modify the I/O numbers, etc.

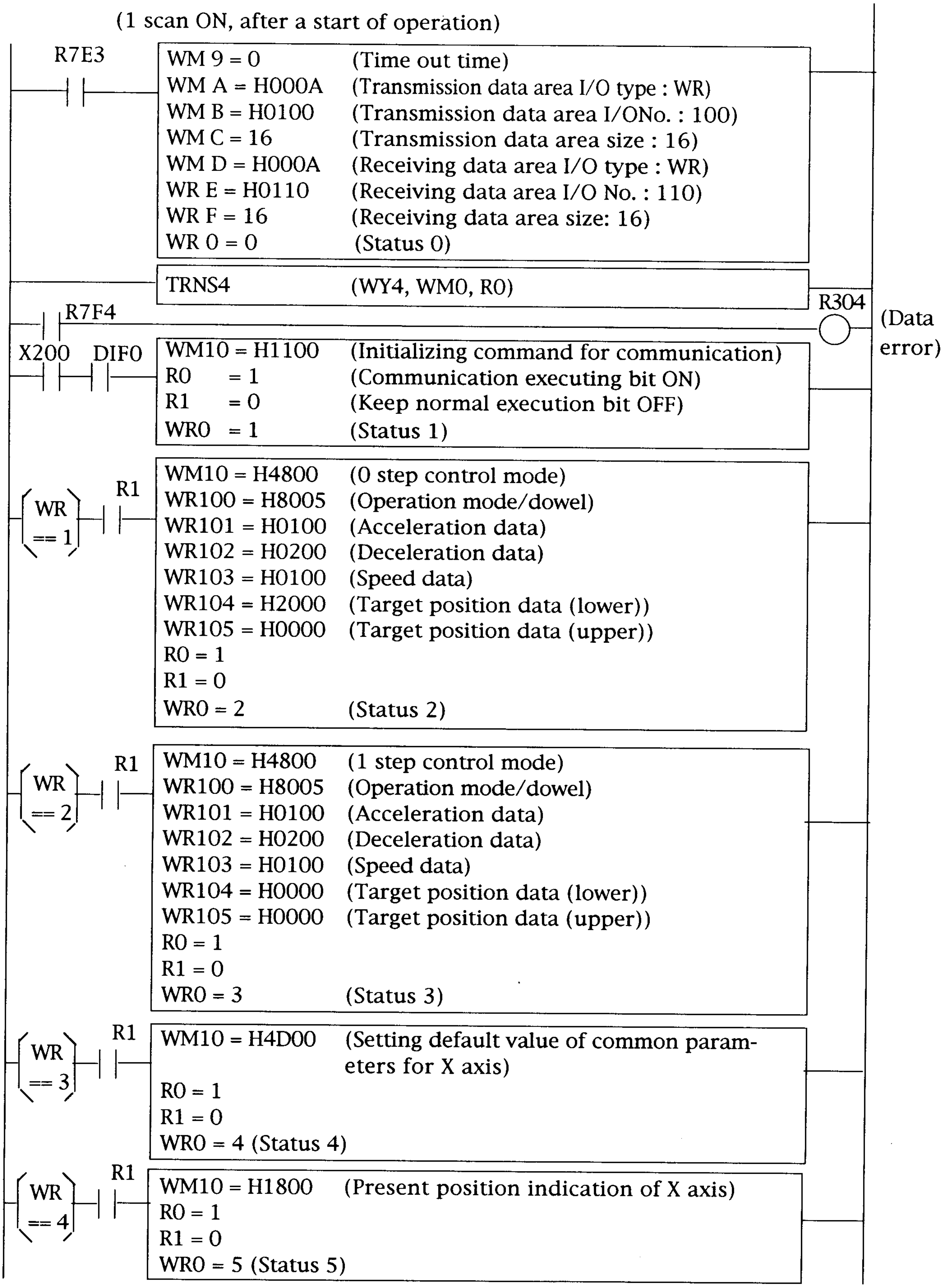
External I/O No.	Function/Description	External I/O No.	Function/Description
X200	•Initializing for communication •Position data setting for automatic operations of 0, 1 steps •Default setting of common parameters for X axis. •Present position indication of X axis. •Resetting to the original point.	X203	•Continuous cycle mode
		Y300	•Execution of communication
		Y301	•Normal end
		Y302	•Abnormal end
		WY40	•Indication area contents (WX2)
X201	•Manual operation mode	WY50	•Indication area contents (WX3)
X202	•Temporary stop		

(4) Position data for automatic operations of 0, 1

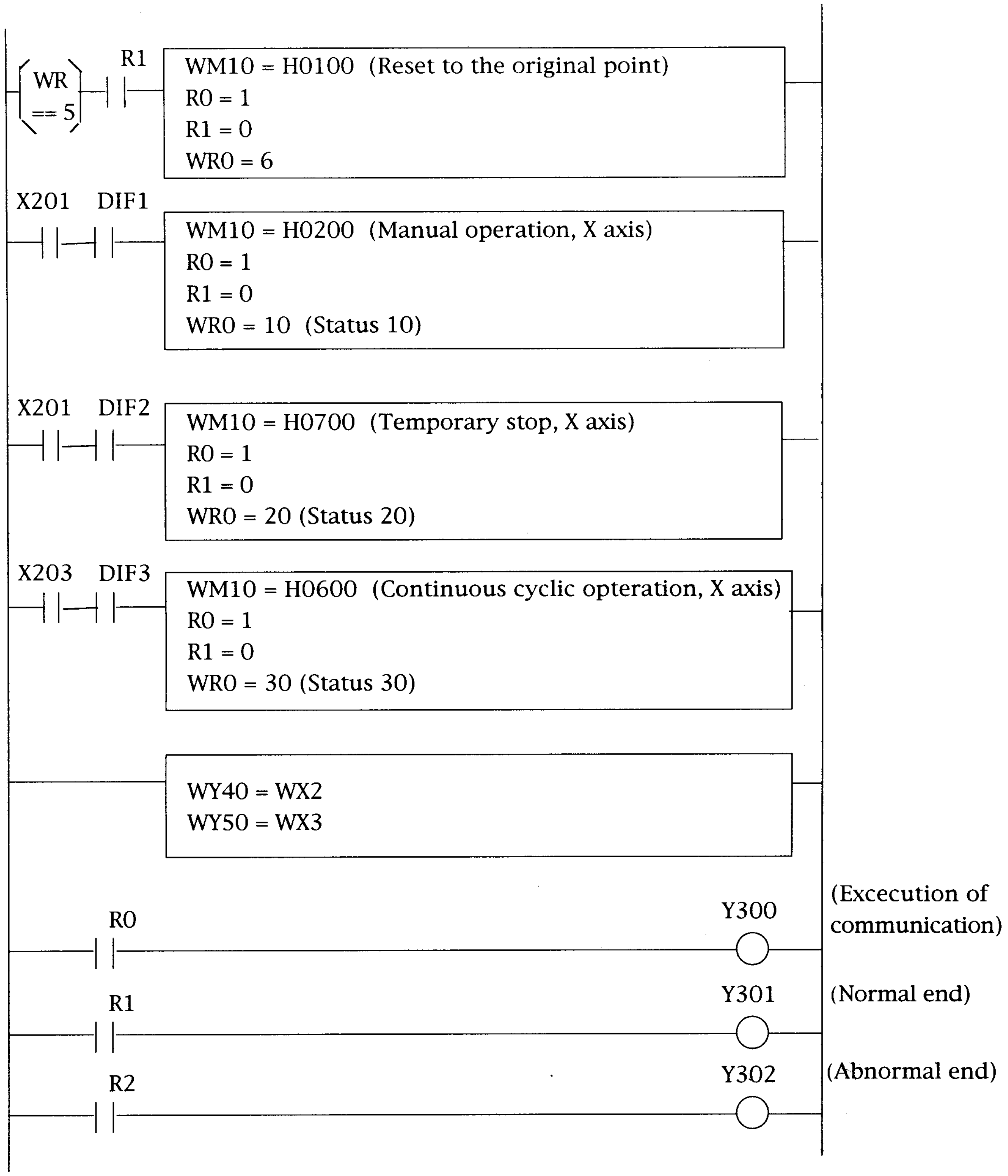
Data No.	Data (Hexadecimal)		Description
	Step 0	Step 1	
1	8005	C005	Operation effective, No variation in speed, Independent, Position mode, Absolute dowel 100ms
2	0100	0100	Acceleration (5kpps/s)
3	0200	0200	Deceleration (2.5kpps/s)
4	0100	0100	Speed data (1.6kpps)
5	2000	0000	Target position data (8192/0 pulse)

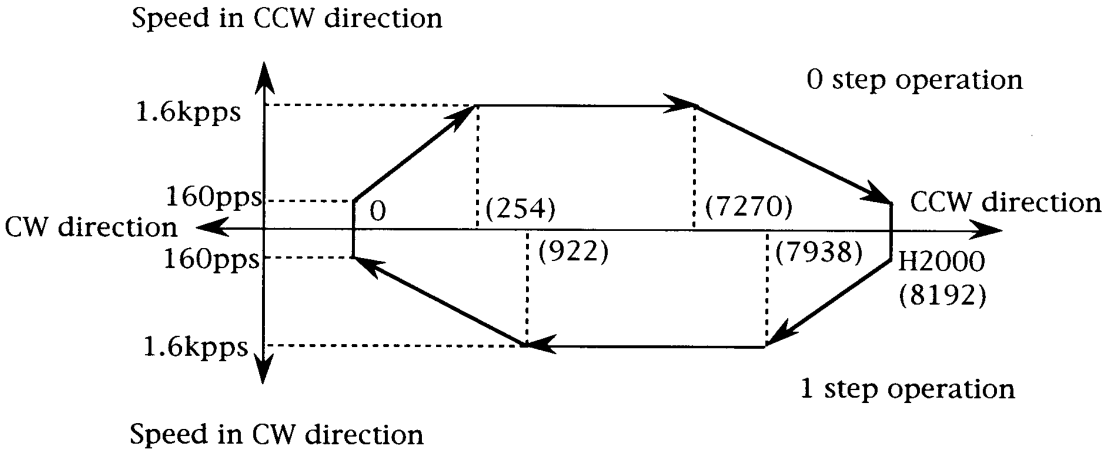
[Example program]

(1 scan ON, after a start of operation)



```
WM 9 = 0          (Time out time)
WM A = H000A      (Transmission data area I/O type : WR)
WM B = H0100      (Transmission data area I/ONo. : 100)
WM C = 16         (Transmission data area size : 16)
WM D = H000A      (Receiving data area I/O type : WR)
WR E = H0110      (Receiving data area I/O No. : 110)
WR F = 16         (Receiving data area size: 16)
WR 0 = 0          (Status 0)

TRNS4             (WY4, WM0, R0)

WM10 = H1100      (Initializing command for communication)
R0   = 1          (Communication executing bit ON)
R1   = 0          (Keep normal execution bit OFF)
WR0  = 1          (Status 1)

WM10 = H4800      (0 step control mode)
WR100 = H8005     (Operation mode/dowel)
WR101 = H0100     (Acceleration data)
WR102 = H0200     (Deceleration data)
WR103 = H0100     (Speed data)
WR104 = H2000     (Target position data (lower))
WR105 = H0000     (Target position data (upper))
R0 = 1
R1 = 0
WR0 = 2           (Status 2)

WM10 = H4800      (1 step control mode)
WR100 = H8005     (Operation mode/dowel)
WR101 = H0100     (Acceleration data)
WR102 = H0200     (Deceleration data)
WR103 = H0100     (Speed data)
WR104 = H0000     (Target position data (lower))
WR105 = H0000     (Target position data (upper))
R0 = 1
R1 = 0
WR0 = 3           (Status 3)

WM10 = H4D00      (Setting default value of common param-
                  eters for X axis)
R0 = 1
R1 = 0
WR0 = 4 (Status 4)

WM10 = H1800      (Present position indication of X axis)
R0 = 1
R1 = 0
WR0 = 5 (Status 5)
```


WR
== 5
R1
WM10 = H0100 (Reset to the original point)
R0 = 1
R1 = 0
WR0 = 6
X201
DIF1
WM10 = H0200 (Manual operation, X axis)
R0 = 1
R1 = 0
WR0 = 10 (Status 10)
X201
DIF2
WM10 = H0700 (Temporary stop, X axis)
R0 = 1
R1 = 0
WR0 = 20 (Status 20)
X203
DIF3
WM10 = H0600 (Continuous cyclic opteration, X axis)
R0 = 1
R1 = 0
WR0 = 30 (Status 30)
WY40 = WX2
WY50 = WX3
R0
Y300
(Excecution of communication)
R1
Y301
(Normal end)
R2
Y302
(Abnormal end)



① When X200 is ON, the following operation will be executed.
•Initializing for communication.
•Data setting for positioning of the automatic operations, 0 step and 1 step. (See example program (4))
•Indicating the present position of X axis.

② When X201 is ON, the manual operation mode will be set.
The motor will be started when the manual switches CW or CCW (in the motor system) are ON.

③ When X202 is ON, the temporary stop mode will be set. The motor will be stopped.

④ When X203 is ON, the continuous cyclic operation mode will be set. The motor will be operated continuously (See the illustration above).

⑤ The informations in TRNS4 commands will be indicated on Y300 = the communication performed , Y301 = the normal execution , Y302 = the abnormal execution , DER = Y304.

⑥ Contents of the indication area will be appeared on WY30 and WY40. The present position of the axis X will be indicated after the command is exported to indicate the present position of the axis X by ①.

3.3 Details of Commands

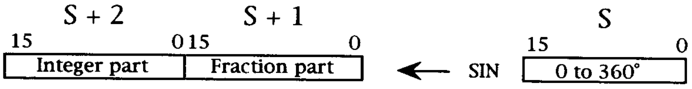
(1) Basic Commands
(2) Arithmetic Commands
(3) Application Commands
(4) Control Commands
(5) Transfer Commands for Sophisticated Function Module
(6) FUN Commands

<table>
<tr><td>Name</td><td colspan="7">SINE Function</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 10 (S)
* (SIN (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">117.9~208.6</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 10 (S)
* (SIN (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument							○						S to S+2 are used.

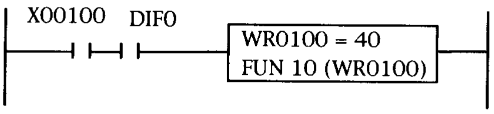
* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]



- The SIN value having a unsigned binary value specified by S or a binary value as an argument is calculated, and the integer part is set in S+2 and the fraction part is set in S+1.
- The SIN value is expressed by a binary value and a negative number is expressed by twos complement.
- When operations are performed normally, DER = 0.
- The data in the fraction part is the real value × 65535.

[Program example]



```
LD X00100
AND DIFO
[
WR0100=40
FUN10 (WR0100)
]
```

[Program explanation]

- An angle of 40° is set in WR0100.
- The SIN operation is performed at the rising edge of X00100, and the fraction part of the result is set in WR0101 in a binary value and the integer part is set in WR0102.

Execution result: WR0102 = H0000, WR0101 = HA48E, WR0100 = H0028

[Precautions]

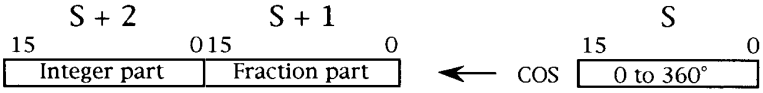
- The argument S is given by an angle (degree) and ranges from 0 to 360. When S is beyond the limits, DER = 1 and no operation is performed.
- When S+1 and S+2 are beyond the maximum I/O number, DER = 1 and no operation is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="7">COSINE Function</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 11 (S)
* (COS (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="6">121.0~207.9</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">FUN 11 (S)
* (COS (S))</td><td colspan="4">Conditions</td><td>Step</td></tr>
<tr><td colspan="4" rowspan="2">—</td><td rowspan="2">3</td></tr>
<tr></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument							○						S to S+2 are used

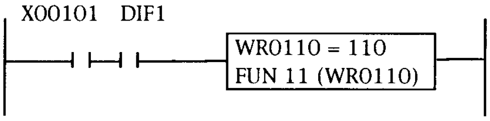
* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]



- The COS value having a unsigned binary value specified by S or a binary value as an argument is calculated, and the integer part is set in S+2 and the fraction part is set in S+1.
- The COS value is expressed by a binary value and a negative number is expressed by twos complement.
- When operations are performed normally, DER = 0.
- The data in the fraction part is the real value × 65535.

[Program example]



```
LD X00101
AND DIF1
[
WR0110=110
FUN11 (WR0110)
]
```

[Program explanation]

- An angle of 110° is set in WR0110.
- The COS operation is performed at the rising edge of X00101, and the fraction part of the result is set in WR0111 in a binary value and the integer part is set in WR0112.

Execution result: WR0112 = HFFFF, WR0111 = HA871, WR0110 = H006E

[Precautions]

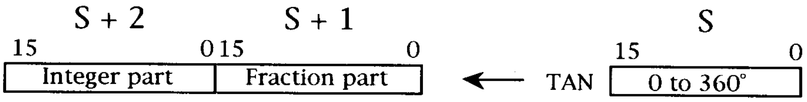
- The argument S is given by an angle (degree) and ranges from 0 to 360. When S is beyond the limits, DER = 1 and no operation is performed.
- When S+1 and S+2 are beyond the maximum I/O number, DER = 1 and no operation is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="7">TANGENT Function</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 12 (S)
* (TAN (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">117.8~206.7</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 12 (S)
* (TAN (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>s</td><td>Argument</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>S to S+2 are used</td></tr>
<tr><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></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>

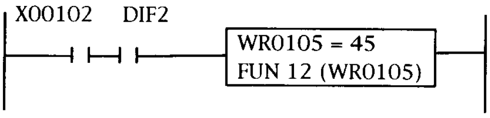
* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]



- The TAN value having a unsigned binary value specified by S or a binary value as an argument is calculated, and the integer part is set in S+2 and the fraction part is set in S+1.
- The TAN value is expressed by a binary value and a negative number is expressed by twos complement.
- When operations are performed normally, DER = 0.
- The data in the fraction part is the real value × 65535.

[Program example]



```
LD X00102
AND DIF2
[
WR0105=45
FUN12 (WR0105)
]
```

[Program explanation]

- An angle of 45° is set in WR0105.
- The SIN operation is performed at the rising edge of X00102, and the fraction part of the result is set in WR0106 in a binary value and the integer part is set in WR0107.

Execution result: WR0107 = H0001, WR0106 = H0000, WR0105 = H002D

[Precautions]

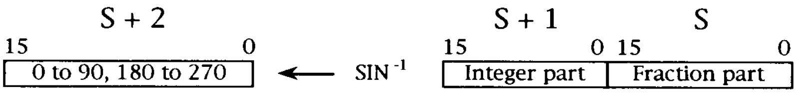
- The argument S is given by an angle (degree) and ranges from 0 to 360. When S = 90 or 270, H7FFF or HFFFF is set in S+2 and S+1. When S is beyond the limits, DER = 1 and no operation is performed. When S+1 and S+2 are beyond the maximum I/O number, DER = 1 and no operation is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

Name	ARC SINE Function						
Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 13 (S) * (ASIN (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	
	DER	ERR	SD	V	C	472.7~487.8	
	↕	●	●	●	●		
Command format	No. of steps						
FUN 13 (S) * (ASIN (S))	Conditions			Step			
	—			3			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument (fraction)							○						S to S+2 are used.
s+1	Argument (integer)							○						

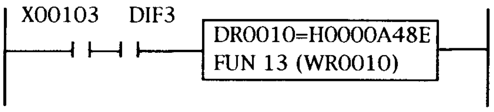
* The command in parentheses ( ) is for the Ladder Editor.

[Function]



- The SIN^{-1} value having a binary value specified by S (fraction part) and S+1 (integer part) as an argument is calculated and outputted to S+2.
- The SIN^{-1} value is an angle (degree) ranging from 0 to 90 or from 180 to 270 and expressed by a binary value.
- When operations are performed normally, DER = 0.
- The data in the fraction part is the real value x 65535.

[Program example]



LD X00103
AND DIF3
[
DR0010=H0000A48E
FUN13 (WR0010)
]

[Program explanation]

- Data is set in DR0010 (WR0010, WR0011).
- The SIN^{-1} operation is performed at the rising edge of X00103, and the result is set in WR0012 in a binary value.

Execution result: WR0012 = H0028, WR0011 = H0000, WR0010 = HA48E

[Precautions]

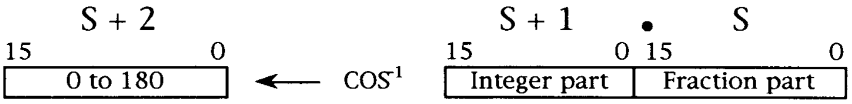
- When |S+1.S| > 1, DER = 1 and no operation is performed.
- When S+1 and S+2 are beyond the maximum I/O number, DER = 1 and no operation is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="7">ARC COSINE Function</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 14 (S)
* (ACOS (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="6">474.3~486.0</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">FUN 14 (S)
* (ACOS (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument (fraction)							○						S to S+2 are used
s+1	Argument (integer)							○						

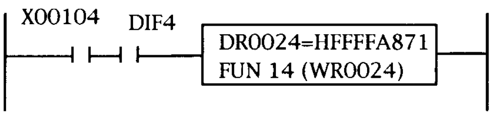
* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]



- The COS^{-1} value having a binary value spesified by S (fraction part) and S+1 (integer part) as an argument is calculated and outputted to S+2.
- The COS^{-1} value is an angle (degree) ranging from 0 to 180 and expressed by a binary value.
- When operations are performed normally, DER = 0.
- The data in the fraction part is the real value × 65535.

[Program example]



```
LD X00104
AND DIF4
[
DR0024=HFFFFA871
FUN14 (WR0024)
]
```

[Program explanation]

- Data is set in DR0024 (WR0024, WR0025).
- The COS^{-1} operation is performed at the rising edge of X00104, and the result is set in WR0026 in a binary value.

  Execution result: WR0026 = H006E, WR0025 = HFFFF, WR0024 = HA871

[Precautions]

- When |S+1.S| > 1, DER = 1 and no operation is performed.
- When S+1 and S+2 are beyond the maximum I/O number, DER = 1 and no operation is performed.

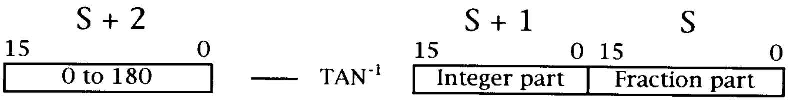
  In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

Name	ARC TANGENT Function						
Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 15 (S) * (ATAN (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	
	DER	ERR	SD	V	C	402.5~494.9	
	↕	●	●	●	●		
Command format	No. of steps						
FUN 15 (S) * (ATAN (S))	Conditions			Step			
	—			3			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument (fraction)							○						S to S+2 are used.
s+1	Argument (integer)							○						

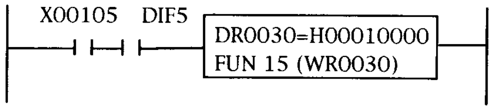
* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]



- The TAN-1 value having a binary value spesified by S (fraction part) and S+1 (integer part) as an argument is calculated and outputted to S+2.
- The TAN-1 value is an angle (degree) ranging from 0 to 180 and expressed by a binary value.
- When operations are performed normally, DER = 0.
- The data in the fraction part is the real value × 65535.

[Program example]



```
LD X00105
AND DIF5
[
DR0030=H00010000
FUN15 (WR0030)
]
```

[Program explanation]

- Data is set in DR0030 (WR0030, WR0031).
- The TAN-1 operation is performed at the rising edge of X00105, and the result is set in WR0032 in a binary value.

Execution result: WR0032 = H002D, WR0031 = H0001, WR0030 = H0000

[Precautions]

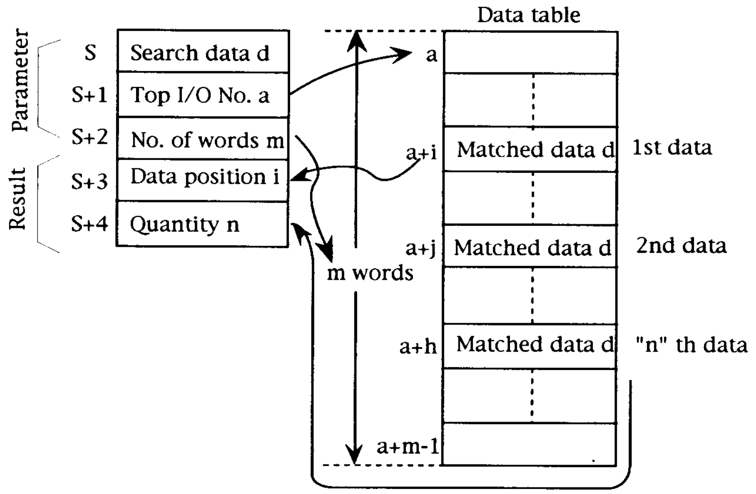
- When S+1 and S+2 are beyond the maximum I/O number, DER = 1 and no operation is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="7">Data Search Function</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 20 (S)
* (DSRCH (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="6">When the matched data are two in the table of 10 words.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">310.0</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 20 (S)
* (DSRCH (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
S	Search data							○						
S+1	Top I/O No. of data table							○						The real address is set.
S+2	No. of search words							○						S to S+4 are used.

* The command in parentheses ( ) is for the LADDER EDITOR.

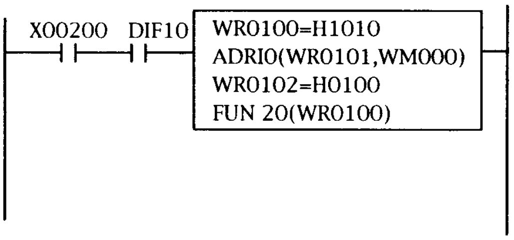
[Function]



- The search data specified by the world internal output is searched for within the word number range, specified by S+2, of the top I/O (the real address is specified) specified by S+1, and the data position (relative position from the top I/O), which is found first, is outputted to S+3 and the data quantity, which is found, is outputted to S+4.
- When no matched data is found, 0 is set in S+3.
- Set the real address of the search top I/O in S+1 using the ADRIO command. ADRIO (S+1, top I/O)
- When operations are performed normally, DER = 0.

[Program example]

Data H1010 is searched for among 256 words from WM000. After searching, the data position is set in WR0103 and the data quantity is set in WR0104.



```
LD X00200
AND DIF10
[
WR0100=H1010
ADRIO (WR0101,WM000)
WR0102=H0100
FUN20 (WR0100)
]
```

[Precautions]

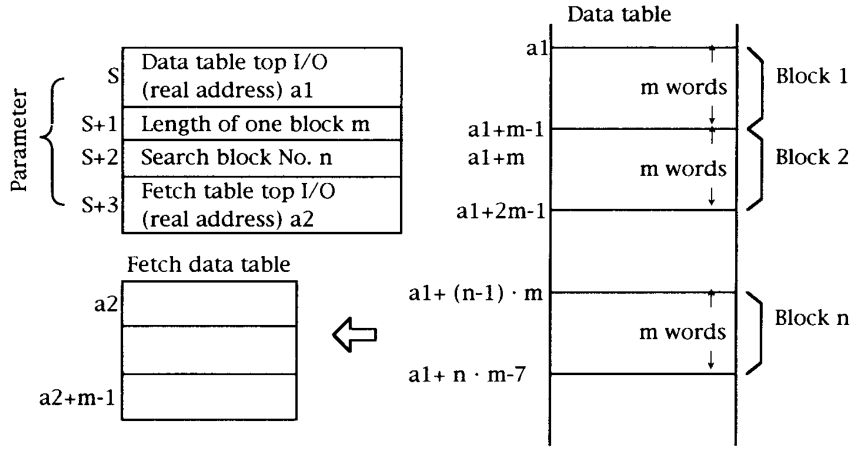
- Set the real address of WR, WL, or WM as a parameter of S+1 using the ADRIO command. When other addreses are set, DER = 1 and no processing is performed.
- When S to S+4 or the areas specified by them are overlapped, DER = 1 and no processing is performed.
- When S+1 and S+2 or the areas specified by them are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="8">Table Search Function</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μs)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 21 (S)
* (TSRCH (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="7">When 10th block is searched by 2 word per 1 block.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="6">330.3</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">FUN 21 (S)
* (TSRCH (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3" rowspan="2">—</td><td colspan="2" rowspan="2">3</td></tr>
<tr></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>S</td><td>Data table top I/O No.</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>The real address is set.</td></tr>
<tr><td>S+1</td><td>Length of one block</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>S to S+3 are used.</td></tr>
<tr><td>S+2</td><td>Search block No.</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>S+3</td><td>Fetch table top I/O No.</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>The real address is set.</td></tr>
</table>

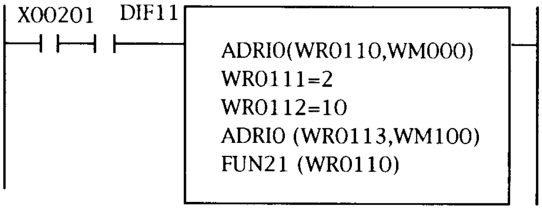
* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]



- The data block whose number is specified by S+2 is fetched from the data table, which is defined by the word internal output S and S+1, and stored from the I/O specified by S+3.
- Set the real addresses of top I/O in S and S+3 using the ADRIO command. ADRIO (S, top I/O)
- When operations are performed normally, DER = 0.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

[Program example]



```
ADRIO(WR0110,WM000)
WR0111=2
WR0112=10
ADRIO (WR0113,WM100)
FUN21 (WR0110)
```

```
LD X00201
AND DIF11
[
ADRIO (WR0110,WM000)
WR0111=2
WR0112=10
ADRIO(WR0113,WM100)
FUN21 (WR0110)
]
```

[Program explanation]

The 10th data block is fetched from the data table (one block consists of 2 words) of WM000 and set in the area of WM100.

[Precautions]

- Set the real addresses of WR, WL, or WM as parameters of S and S+3 using the ADRIO command. When other address are set, DER = 1 and no processing is performed.
- When S to S+3 or the areas specified by them are overlapped, DER = 1 and no processing is performed.
- When S to S+3 or the areas specified by them are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="7">16-Bit Unsigned Binary → Decimal ASCII Conversion
(Binary to Decimal ASCII)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 30 (S)
* (BINDA (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="6">199.1~291.1</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">FUN 30 (S)
* (BINDA (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>s</td><td>Argument (conversion data)</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>S to S+3 are used.</td></tr>
<tr><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></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></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>

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

16-bit unsigned binary data

S: 0 to 65535 ⇨

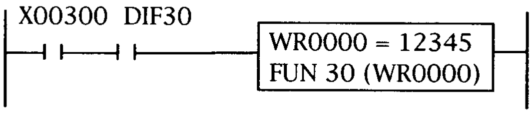
10^n: ASCII code in the 10^n place

Decimal ASCII data

	15 8	7 0
S + 1	10^4	10^3
S + 2	10^2	10^1
S + 3	10^0	NULL

- The 16-bit unsigned binary data specified by the argument S is converted to a 5-digit decimal ASCII code and set in S+1 to S+3.
- The conversion result is suppressed to 0 and their digits are H20 (space).
- The remaining digits after ASCII conversion are null, indicating the end of character string.
- When operations are performed normally, DER = 0.

[Program example]



```
LD X00300
AND DIF30
 [
WR0000=12345
FUN30 (WR0000)
 ]
```

[Program explanation]

- Binary data of 12345 which is set in WR0000 is converted to an ASCII. The conversion result is set in WR0001 to WR0003.

Execution result: WR0000 = 12345 (H3039), WR0001 = H3132, WR0002 = H3334,

WR0003 =H3500

[Precautions]

- When S+1 and S+2 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="7">32-Bit Signed Binary → Decimal ASCII Conversion
(Double Binary to Decimal ASCII)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 31 (S)
* (DBINDA (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">281.1~425.1</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 31 (S)
* (DBINDA (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">——</td><td colspan="2">3</td><td></td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument (low order)							○						-2147483648 to 2147483647
s+1	Argument (high order)							○						S to S+7 are used.

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

32-bit signed binary data

S	Low-order 16 bits
S+1	High-order 16 bits

⇨

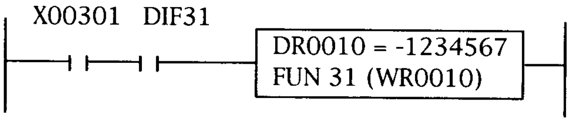
	15 8	7 0
S + 2	Sign	10^9
S + 3	10^8	10^7
S + 4	10^6	10^5
S + 5	10^4	10^3
S + 6	10^2	10^1
S + 7	10^0	NULL

Sign: Positive : H20 (space)
Negative: H2D (-)

10^n: ASCII code in the 10^n place

- The 32-bit signed binary data specified by S (low order) and S+1 (high order) is converted to a 10-digit decimal ASCII code and set in S+2 to S+7.
- When the sign is positive, H20 (space) is provided. When the sign is negative, H2D (-) is provided.
- The conversion result is suppressed to 0 and their digits are H20 (space).
- The remaining digits after ASCII conversion are null, indicating the end of character string.
- When operations are performed normally, DER = 0.

[Program example]



LD X00301
AND DIF31
[
DR0010= -1234567
FUN 31 (WR0010)
]

[Program explanation]

- Binary data of -1234567 which is set in DR0010(WR0010, WR0011) is converted to an ASCII.
- The conversion result is set in WR0012 to WR0017.

  Execution result: DR0010 = - 1234567 (HFFED2979), WR0012 = H2D20, WR0013 = H2020, WR0014 = H3132, WR0015 = H3334, WR0016 = H3536, WR0017 = H3700

[Precautions]

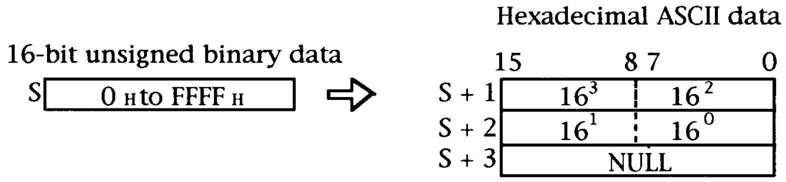
- When S+1 and S+7 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="7">16-Bit Binary → Hxadecimal ASCII Conversion
(Binary to Hexa ASCII)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 32 (S)
* (BINHA (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">125.5~203.9</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 32 (S)
* (BINHA (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">——</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument							○						S to S+3 are used

* The command in parentheses ( ) is for the LADDER EDITOR.

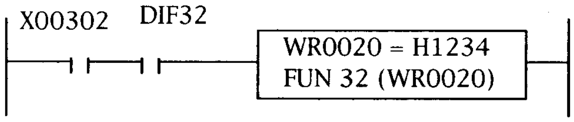
[Function]



16^n: ASCII code in the 16^n place

- The 16-bit unsigned binary data specified by the argument S is converted to a 4-digit hexadecimal ASCII code and set in S+1 to S+3.
- The conversion result is not suppressed to 0.
- Null behind the ASCII data means the end of character string.
- When operations are performed normally, DER = 0.

[Program example]



```
LD X00302
AND DIF32
[
WR0020=H1234
FUN32 (WR0020)
 ]
```

- Binary data of H1234 which is set in WR0020 is converted to an ASCII.
- The conversion result is set in WR0021 to WR0023.

Execution result: WR0020 = H1234, WR0021 = H3132, WR0022 = H3334, WR0023 = H0000

[Precautions]

- When S+1 to S+3 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

Name	32-Bit Binary → Hexadecimal ASCII Conversion(Double Binary to HEXA ASCII)

Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 33 (S) * (DBINHA (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	
	DER	ERR	SD	V	C	164.2~246.1	
	↕	●	●	●	●		

Command format	No. of steps	
	Conditions	Step
FUN 33 (S) * (DBINHA (S))	—	3

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument (low order)							○						H00000000 to HFFFFFFFF
s+1	Argument (high order)							○						S to S+6 are used

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

32-bit unsigned binary data

S	Low-order 16 bits
S+1	High-order 16 bits

⇨

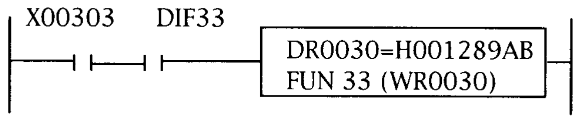
Hexadecimal ASCII data

	15 8	7 0
S + 2	16^7	16^6
S + 3	16^5	16^4
S + 4	16^3	16^2
S + 5	16^1	16^0
S + 6	NULL	

16^n: ASCII code in the 16^n place

- The 32-bit binary data specified by S (low order) and S+1 (high order) is converted to a 8-digit hexadecimal ASCII code and set in S+2 to S+6.
- The conversion result is not suppressed to 0.
- Null behind the ASCII data means the end of character string.
- When operations are performed normally, DER = 0.

[Program example]



```
LD X00303
AND DIF33
[
DR0030=H001289AB
FUN 33 (WR0030)
]
```

[Program explanation]

- Binary data of H001289AB which is set in DR0030 is converted to an ASCII.
- The conversion result is set in WR0032 to WR0036.

  Execution result: DR0030 = H001289AB, WR0032 = H3030, WR0033 = H3132, WR0034 = H3839, WR0035 = H4142, WR0036 = H0000

[Precautions]

- When S+1 to S+6 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

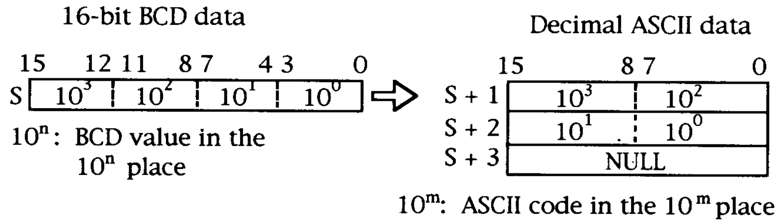
Name	16-Bit BCD → Decimal ASCII Conversion (BCD to Decimal ASCII)

<table>
<tr><th>Ladder format</th><th colspan="5">Condition code</th><th>Processing time (μ s)</th><th>Remarks</th></tr>
<tr><td rowspan="3">FUN 34 (S)
* (BCDDA (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">130.8~208.8</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 34 (S)
* (BCDDA (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument (BCD)							○						S to S+3 are used

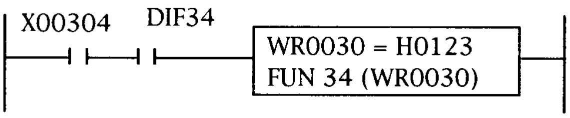
* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]



- The 16-bit BCD data specified by the argument S is converted to a 4-digit decimal ASCII code and set in S+1 to S+3.
- The conversion result is suppressed to 0 and their digits are H20 (space).
- Null behind the ASCII data means the end of character string.
- When operations are performed normally, DER = 0.

[Program example]



```
LD X00304
AND DIF34
 [
WR0030=H0123
FUN34 (WR0030)
 ]
```

[Program explanation]

- BCD data of H0123 which is set in WR0030 is converted to an ASCII.
- The conversion result is set in WR0031 to WR0033.

Execution result: WR0030 = H0123, WR0031 = H2031, WR0032 = H3233, WR0033 = H0000

[Precautions]

- When S is other than BCD data, DER = 1 and no processing is performed.
- When S+1 to S+3 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

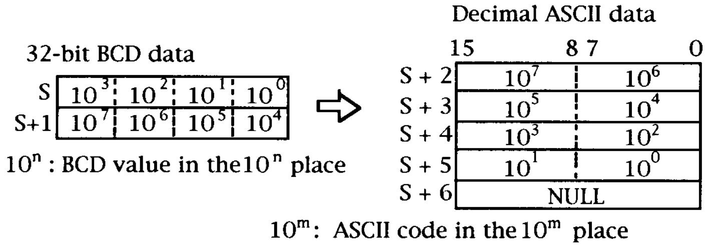
Name	32-Bit BCD → Decimal ASCII Conversion (Double BCD TO Decimal ASCII)

Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 35 (S) * (DBCDDA (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	
	DER	ERR	SD	V	C	175.3~286.2	
	↕	●	●	●	●		
Command format	No. of steps						
FUN 35 (S) * (DBCDDA (S))	Conditions			Step			
	—			3			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument (low order)							○						S indicates BCD data.
s+1	Argument (high order)							○						S to S+6 are used.

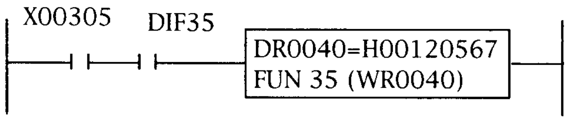
* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]



- The 32-bit BCD data specified by S (low order) and S+1 (high order) is converted to a 8-digit decimal ASCII code and set in S+2 to S+6.
- The conversion result is suppressed to 0 and their digits are H20 (space).
- Null behind the ASCII data means the end of character string.
- When operations are performed normally, DER = 0.

[Program example]



```
LD X00305
AND DIF35
 [
DR0040=H00120567
FUN 35 (WR0040)
 ]
```

[Program explanation]

- BCD data of H00120567 which is set in DR0040 is converted to an ASCII.

- The conversion result is set in WR0042 to WR0046.

  Execution result: DR0040 = H00120567, WR0042 = H2020, WR0043 = H3132, WR0044 = H3035, WR0045 = H3637, WR0046 = H0000

[Precautions]

- When S and S+1 are other than BCD data, DER = 1 and no processing is performed.
- When S+1 to S+6 are beyond the maximum I/O number, DER = 1 and no processing isperformed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

Name	Unsigned 5-Digit Decimal ASCII → 16-Bit Binary Conversion (Decimal ASCII to Binary)

Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 36 (S) * (DABIN (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	
	DER	ERR	SD	V	C	179.5~286.1	
	↕	●	●	●	●		
Command format	**No. of steps**						
FUN 36 (S) * (DABIN (S))	Conditions			Step			
	—			3			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument (high order)							○						S to S+2 are combinations of H00, H20, and H30 to H39.
s+1	Argument (middle order)							○						
s+2	Argument (low order)							○						S to S+3 are used.

* The command in parentheses ( ) is for the LADDER EDITOR.

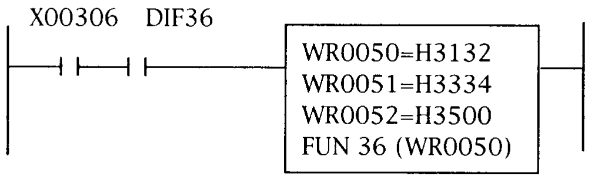
[Function]

	Unsigned decimal ASCII data				16-bit binary data
S	10^4	10^3	⇨	S+3	0 to 65535
S + 1	10^2	10^1			
S + 2	10^0	H00			

10^n: ASCII code in the 10^n place

- The unsigned 5-digit decimal ASCII data specified by S (high order), S+1 (middle order), and S+2 (low order) is converted to 16-bit binary data and set in S+3.
- H00 and H20 (null, space) in the high order positions are processed as H30 (0). (0-suppressed digits)
- When operations are performed normally, DER = 0.

[Program example]



```
LD X00306
AND DIF36
 [
WR0050=H3132
WR0051=H3334
WR0052=H3500
FUN 36(WR0050)
 ]
```

[Program explanation]

- ASCII data 1,2,3,4, and 5 which are set in WR0050 to WR0052 are converted to binary data.
- The conversion results are set in WR0053.

  Execution result: WR0050 = H3132, WR0051 = H3334, WR0052 = H3500, WR0053 = 12345 (H3039)

[Precautions]

- When 5-digit ASCII codes which are set in S to S+2 are other than H30 to H39 (0 to 9), DER = 1 and no processing is performed. This is not applied to H00 and H20 (null, space) in digits which are suppressed to 0.
- When S+1 to S+3 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- When data is more than 65 536, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="8">Signed 10-Digit Decimal ASCII → 32-Bit Binary Conversion (Double Decimal ASCII TO Binary)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 37 (S)
* (DDABIN (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td colspan="2">H-252</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td colspan="2" rowspan="6">270.6~367.4</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">FUN 37 (S)
* (DDABIN (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3" rowspan="2">—</td><td colspan="2" rowspan="2">3</td></tr>
<tr></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R,
L,
M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR,
WL,
WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR,
DL,
DM</td></tr>
<tr><td>s</td><td>Argument (ASCII code)</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 rowspan="3">The sign is H20 or H2D. The other digits are combinations of H00, H20, and H30 to H39.
S to S+7 are used.</td></tr>
<tr><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>s+5</td><td>Argument (ASCII code)</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></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>

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

Signed decimal ASCII data

S	Sign	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

⇨

Signed 32-bit binary data

S+6	Low-order 16 bits
S+7	High-order 16 bits

10^n: ASCII code in the 10^n place

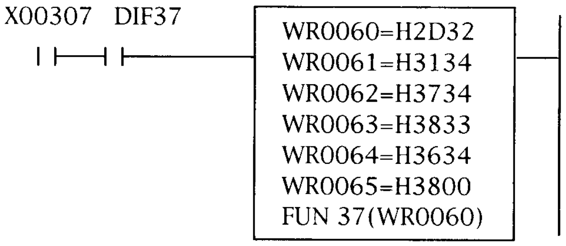
Sign: H20 (space): Positive
H2D (-): Negative

- The signed 10-digit decimal ASCII data specified by the arguments s to S+5 is converted to 32-bit binary data and set in S+7 (high order) and S+6 (low order).
- The argument is a combination of H00, H20, H30 to H39, and H2D (-).
- H00 and H20 (null, space) in the high order positions are processed as H30 (0). (0-suppressed digits)
- When operations are performed normally, DER = 0.
- Signed data ranges from -2147483648 to 2147483647.

[Precautions]

- In cases except for that the sign is H20 or H2D and the other digits are to H30 to H39 (0 to 9), DER = 1 and no processing is performed. This is not applied to H00 and H20 (null, space) in digits which are suppressed to 0.
- When data is other than -2147483648 to 2147483647, DER = 1 and no processing is performed.
- When S+1 to S+7 are beyond the maximum I/O number, DER = 1 and no processing is preformed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

[Program example]



```
LD X00307
AND DIF37
 [
WR0060=H2D32
WR0061=H3134
WR0062=H3734
WR0063=H3833
WR0064=H3634
WR0065=H3800
FUN 37(WR0060)
 ]
```

[Program explanation]

- ASCII data "-" ,2,1,4,7,4,8,3,6,4, and 8 which are set in WR0060 to WR0065 are converted to binary data.

  The conversion results are set in WR0067 (high order) and WR0066 (low order).

  Execution result: WR0060 = H2D32, WR0061 = H3134, WR0062 = H3734, WR0063 = H3833, WR0064 = H3634, WR0065 = H3800, DR0066 = -2147483648 (H80000000)

Name	4-Digit Hexadecimal ASCII → 16-Bit Binary Conversion (Hexa ASCII to Binary)

Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 38 (S) * (HABIN (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	
	DER	ERR	SD	V	C	168.6~272.6	
	↕	●	●	●	●		

Command format	No. of steps	
	Conditions	Step
FUN 38 (S) * (HABIN (S))	—	3

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
S	Argument (high order ASCII data)							○						Combinations of H00, H20, H30 to H39, and H41 to H46
S+1	Argument (low order ASCII data)							○						
														S to S+2 are used.

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

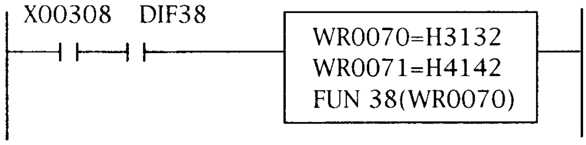
Hexadecimal ASCII data → 16-bit binary data

S	16^3	16^2	⇨	S+2	H0000 to HFFFF
S + 1	16^1	16^0			

16^n: ASCII code in the 16^n place

- The 4-digit hexadecimal ASCII data specified by the arguments S and S+1 is converted to binary data and outputted to S+2.
- H00 and H20 (null, space) in the high order positions are processed as H30 (0). (0-suppressed digits)
- The argument is a combination of H30 to H39 and H41 to H46 (0 to 9, A to F).
- When operations are performed normally, DER = 0.

[Program example]



```
LD X00308
AND DIF38
 [
WR0070=H3132
WR0071=H4142
FUN 38(WR0070)
 ]
```

[Program explanation]

- ASCII data 1,2,A, and B which are set in WR0070 and WR0071 are converted to binary data.
- The conversion results are set in WR0072.

  Execution result: WR0070 = H3132, WR0071 = H4142, WR0072 = H12AB

[Precautions]

- When 4-digit ASCII codes which are set in S and S+1 are other than H30 to H39 and H41 to H46, DER = 1 and no processing is performed. This is not applied to H00 and H20 (null, space) in digits which are suppressed to 0.
- When S+1 to S+2 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

Name	8-Digit Hexadecimal ASCII → 32-Bit Binary Conversion (Double Hexa ASCII to Binary)

Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 39 (S) * (DHABIN (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	
	DER	ERR	SD	V	C	279.5~407.4	
	↕	●	●	●	●		

Command format	No. of steps	
	Conditions	Step
FUN 39 (S) * (DHABIN (S))	—	3

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument (ASCII data)							○						Combinations of H00, H20, H30 to H39, and H41 to H46
~	~							~						
s+3	Argument (ASCII data)							○						S to S+5 are used.

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

Hexadecimal ASCII data

	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

⇨

32-bit binary data

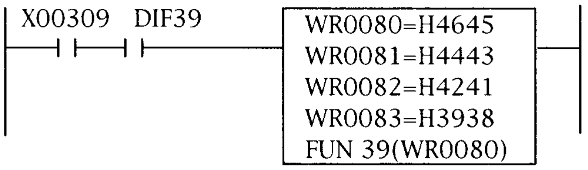
S+4	Low-order 16 bits
S+5	High-order 16 bits

H00000000 to HFFFFFFFF

16^n: ASCII code in the 16^n place

- The 8-digit hexadecimal ASCII data specified by the arguments S to S+3 is converted to binary data and outputted to S+4 and S+5.
- H00 and H20 (null, space) in the high order positions are processed as H30 (0). (0-suppressed digits)
- The argument is a combination of H30 to H39 and H41 to H46 (0 to 9, A to F).
- When operations are performed normally, DER = 0.

[Program example]



```
LD X00309
AND DIF39
 [
WR0080=H4645
WR0081=H4443
WR0082=H4241
WR0083=H3938
FUN 39(WR0080)
 ]
```

[Program explanation]

- ASCII data F,E,D,C,B,A,9 and 8 which are set in WR0080 to WR0083 are converted to binary data. The conversion results are set in WR0084 and WR0085.

  Execution result: WR0080 = H4645, WR0081 = H4443, WR0082 = H4241, WR0083 = H3938,DR0084 = HFEDCBA98

[Precautions]

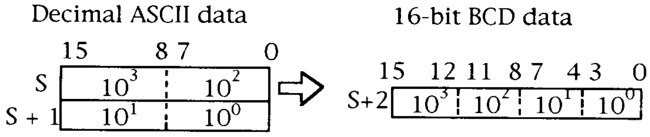
- When 8-digit ASCII codes which are set in S to S+3 are other than H30 to H39 and H41 to H46, DER = 1 and no processing is performed. This is not applied to H00 and H20 (null, space) in digits which are suppressed to 0.
- When S+1 to S+5 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="7">4-Digit Decimal ASCII → 16-Bit BCD Conversion (Decimal ASCII to BCD)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 40 (S)
* (DABCD (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="6">157.3~246.7</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">FUN 40 (S)
* (DABCD (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument (ASCII data)							○						Combinations of H00, H20, and H30 to H39 S to S+2 are used
s+1	Argument (ASCII data)							○						

* The command in parentheses ( ) is for the LADDER EDITOR.

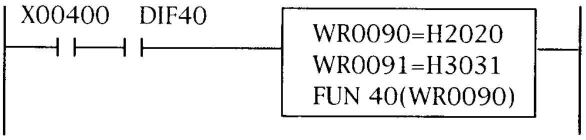
[Function]



10^m : ASCII code in the 10^m place 10^n : ASCII code in the 10^n place

- The 4-digit decimal ASCII data specified by the arguments S and S+1 is converted to 16-bit BCD data and set in S+2.
- H00 and H20 (null, space) in the high order positions are processed as H30 (0). (0-suppressed digits)
- The argument is a combination of H30 to H39(0 to 9).
- When operations are performed normally, DER = 0.

[Program example]



```
LD X00400
AND DIF40
 [
WR0090=H2020
WR0091=H3031
FUN 40(WR0090)
 ]
```

[Program explanation]

- ASCII data "␣", "␣", 0, and 1 which are set in WR0090 and WR0091 are converted to 16-bit BCD data. The conversion results are set in WR0092. ("␣" = H20)

  Execution result: WR0090 = H2020, WR0091 = H3031, WR0092 = H0001

[Precautions]

- When 4-digit ASCII codes which are set in S and S+1 are other than H30 to H39, DER = 1 and no processing is performed. This is not applied to H00 and H20 (null, space) in digits which are suppressed to 0.
- When S+1 and S+2 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

Name	8-Digit Decimal ASCII → 32-Bit BCD Conversion(Double Decimal ASCII to BCD)

Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 41 (S) * (DDABCD (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	
	DER	ERR	SD	V	C	256.8~352.3	
	↕	●	●	●	●		
Command format	No. of steps						
FUN 41 (S) * (DDABCD (S))	Conditions			Step			
	—			3			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument (ASCII data)							○						Combinations of H00, H20, and H30 to H39
	≀							≀						
s+3	Argument (ASCII data)							○						S to S+5 are used

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

Decimal ASCII data

	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^m : BCD code in the 10^m place

⇨

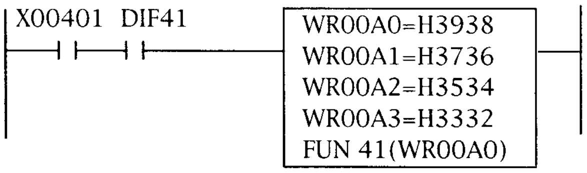
32-bit BCD data

	15 12	11 8	7 4	3 0
S+4	10^3	10^2	10^1	10^0
S+5	10^7	10^6	10^5	10^4

10^n : BCD code in the 10^n place

- The 8-digit decimal ASCII data specified by the arguments S to S+3 is converted to 32-bit BCD data and set in S+4(low order) and S+5(high order).
- H00 and H20 (null, space) in the high order positions are processed as H30 (0). (0-suppressed digits)
- The argument is a combination of H30 to H39 (0 to 9).
- When operations are performed normally, DER = 0.

[Program example]



```
LD X00401
AND DIF41
 [
WR00A0=H3938
WR00A1=H3736
WR00A2=H3534
WR00A3=H3332
FUN41(WR00A0)
 ]
```

[Program explanation]

- ASCII data 9,8,7,6,5,4,3 and 2 which are set in WR00A0 to WR00A3 are converted to 32-bit BCD data. The conversion results are set in WR00A4 and WR00A5.
- Execution result: WR00A0 = H3938, WR00A1 = H3736, WR00A2 = H3534, WR00A3 = H3332, DR00A4 = H98765432

[Precautions]

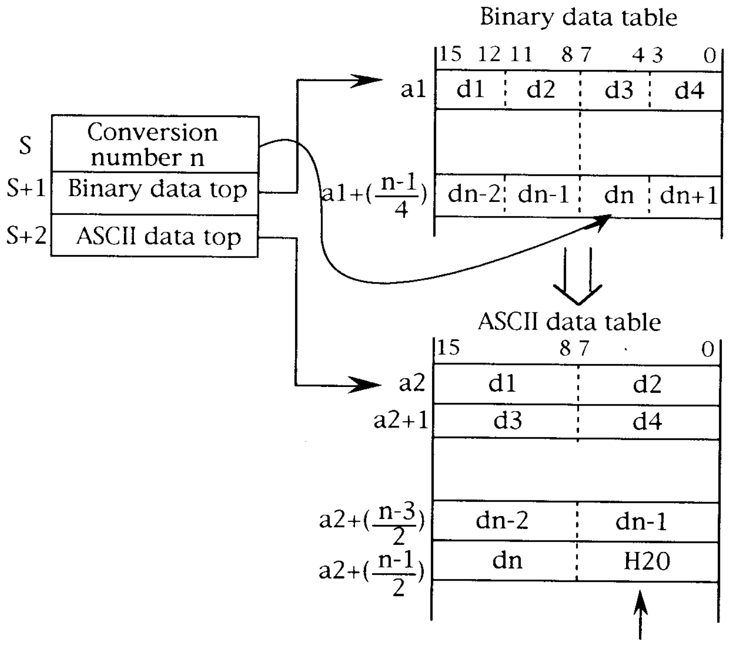
- When 8-digit ASCII codes which are set in S to S+3 are other than H30 to H39, DER = 1 and no processing is performed. This is not applied to H00 and H20 (null, space) in digits which are suppressed to 0.
- When S+1 to S+5 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="8">Hexadecimal Binary → Hexadecimal ASCII Conversion (Binary to ASCII)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 42 (S)
* (ASC (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td colspan="2">H-252</td><td rowspan="7">Conversion of 10 characters</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td colspan="2" rowspan="5">456.2~471.5</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 42 (S)
* (ASC (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	No. ofconversion characters							○						S to S+2 are used.
s+1	Binary data top I/O No.							○						The real address is set.
s+2	Conversion ASCII top I/O No.							○						The real address is set.

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]



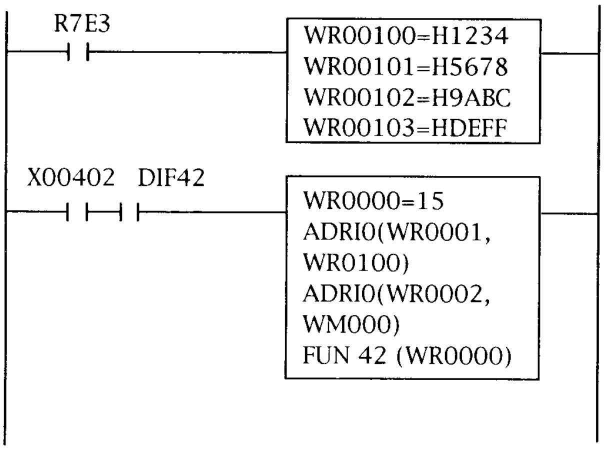
- The characters specified by S from the top I/O of the hexadecimal binary data specified by S+1 are converted to hexadecimal ASCII codes and set starting with the top I/O specified by S+2.
- When the number of characters is odd, the low-order 8 bits of the data of the output destination is H20 (space).
- Set the real addresses in the top I/O of S+1 and S+2 using the ADRIO instruction.

  Example: ADRIO (S+1, data top I/O)
- When operations are performed normally, DER = 0.

[Precautions]

- Set the real addresses in S+1 and S+2 using the ADRIO instruction. When any other addresses are set, DER = 1 and no processing is performed.
- When S to S+2 or the areas specified by them are overlapped, DER = 1 and no processing is performed.
- When S to S+2 or the areas specified by S+1 and S+2 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

[Program example]



```
LD   R7E3
[
WR0100=H1234
WR0101=H5678
WR0102=H9ABC
WR0103=HDEFF
]
LD   X00402
AND  DIF42
[
WR0000=15
ADRIO(WR0001, WR0100)
ADRIO(WR0002, WM000)
FUN 42 (WR0000)
]
```

[Program explanation]

1)Data is set in the data tables WR0100 and the following by the special internal output R7E3 (1 scan is turned ON after the running starts).

2)A hexadecimal number is converted to a hexadecimal ASCII at the rising edge of X00402 and the converted data is set in WM000 and the following.

```
Execution result:WR0100=H1234      WM000=H3132,WM001=H3334
                 WR0101=H5678      WM002=H3536,WM003=H3738
                 WR0102=H9ABC  ⇨   WM004=H3941,WM005=H4243
                 WR0103=HDEFF      WM006=H4445,WM007=H46 20
                                                         ↑
                                                       Space
```

Name	Hexadecimal ASCII → Hexadecimal Binary Conversion (ASCII to Binary)

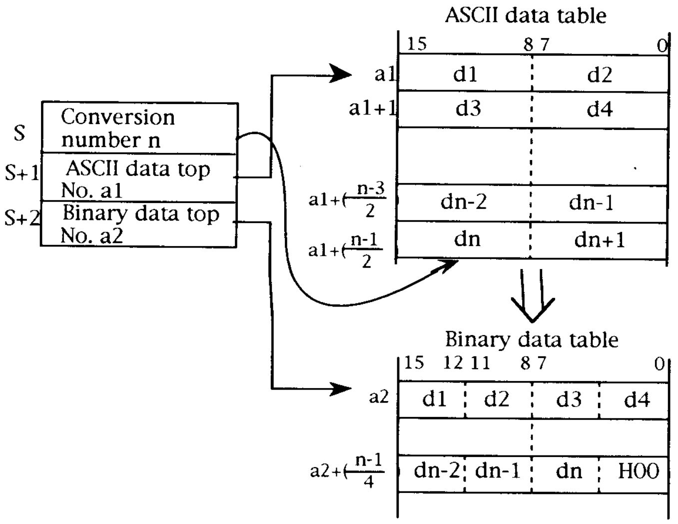
Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 43 (S) * (HEX (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	Conversion of 10 characters
	DER	ERR	SD	V	C	684.7~705.4	
	↕	●	●	●	●		
Command format	No. of steps						
FUN 43 (S) * (HEX (S))	Conditions				Step		
	—				3		

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	No. ofconversion characters							○						S to S+2 are used.
s+1	Conversion ASCII top I/O No.							○						The real address is set.
s+2	Binary data top I/O No.							○						The real address is set.

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

- The characters specified by S from the beginning of the hexadecimal ASCII data specified by S+1 are converted to binary data and set starting with the top I/O specified by S+2.
- When the number of characters is odd, the data of low-order 4 bits of the output destination is 0.



- Set the real addresses in the top I/O of S+1 and S+2 using the ADRIO instruction. ADRIO (S+1, data top I/O)
- H00 and H20 (null,space) in the high order positions are processed as H30 (0).

  (0-suppressed digits)
- When operations are performed normally, DER = 0.

[Precautions]

- Set the real addresses in S+1 and S+2 using the ADRIO instruction. When any other addresses are set, DER = 1 and no processing is performed.
- When S to S+2 or the areas specified by them are overlapped, DER = 1 and no processing is performed.
- When S to S+2 or the areas specified by S+1 and S+2 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

[Program example]

```
LD   R7E3
[
WM000=H3031
WM001=H3233
WM002=H3435
WM003=H3637
WM004=H3839
WM005=H4142
WM006=H4344
WM007=H4546
]
LD   X403
AND  DIF43
[
WR0000=15
ADRIO(WR0001,
WM000)
ADRIO(WR0002,
WR0100)
FUN 43 (WR0000)
]
```

[Program explanation]

1) Data is set in the data tables WM000 and the following by the special internal output R7E3 (1 scan is turned ON after the running starts).

2) A hexadecimal ASCII is converted to a hexadecimal binary number at the rising edge of X00403 and the converted data is set in WR0100 and the following.

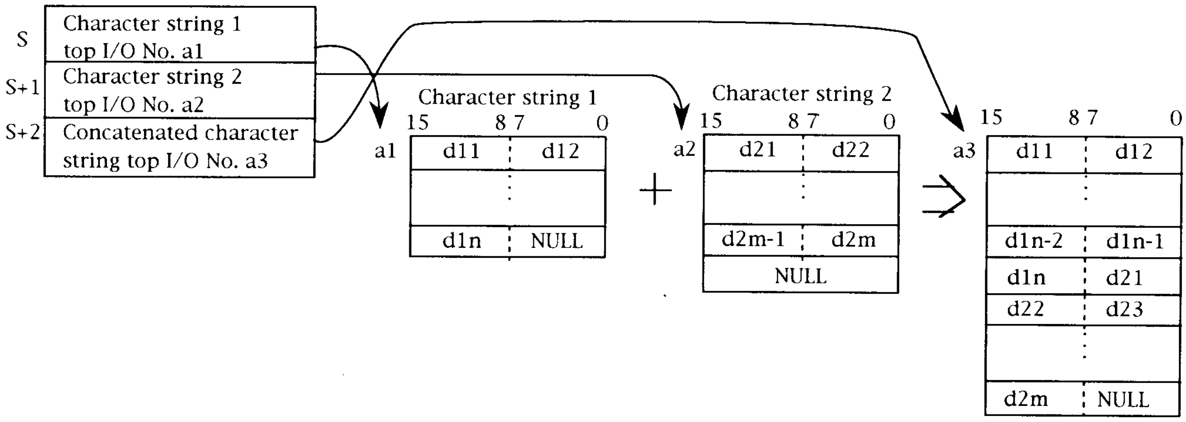
```
Execution result:  WM000=H3031, WM001=H3233         WR0100=H0123
                   WM002=H3435, WM003=H3637         WR0101=H4567
                   WM004=H3839, WM005=H4142    ⇨    WR0102=H89AB
                   WM006=H4344, WM007=H4546         WR0103=HCDE0
```

<table>
<tr><td>Name</td><td colspan="7">Character String Concatenation</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 44 (S)
* (SADD (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="6">10 characters plus 10 characters to 20 characters</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">668.5~859.0</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 44 (S)
* (SADD (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">——</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
S	Character string 1 top I/O No.							○						The real addresses are set in S to S+2.
S+1	Character string 2 topI/O No.							○						
S+2	Concatenated characterstring top I/O No.							○						

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

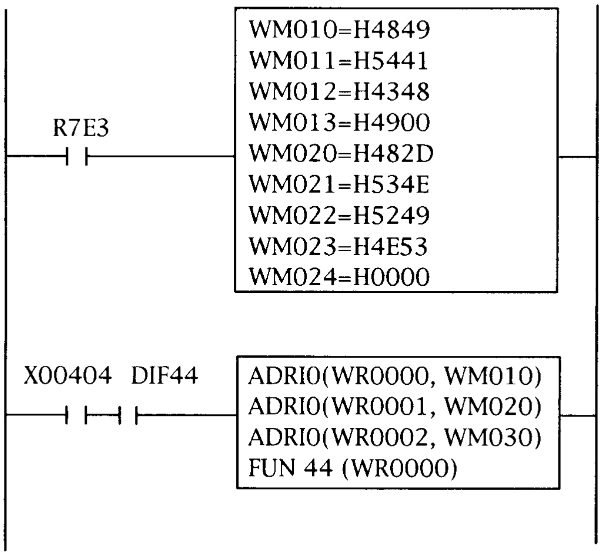


- The character string starting with the top I/O specified by S is concatenated with the character string starting with the top I/O specified by S+1 and set in the top I/O area specified by S+2.
- The character string to be concatenated is assumed to end prior to null (H00).
- Null is set behind the concatenated character string.
- Set the real addresses in the top I/O of S to S+2 using the ADRIO instruction.
- When operations are performed normally, DER = 0.

[Precautions]

- When S to S+2 or the areas specified by them are overlapped, DER = 1 and no processing is performed.
- When S+1 to S+2 or the areas specified by S to S+2 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- Set the real addresses in S to S+2 using the ADRIO instruction. When any other addresses are set, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

[Program example]



```
LD      R7E3
[
WM010=H4849    "H" , "I"
WM011=H5441    "T" , "A"
WM012=H4348    "C" , "H"
WM013=H4900    "I" , NULL
WM020=H482D    "H" , "-"
WM021=H534E    "S" , "E"
WM022=H5249    "R" , "I "
WM023=H4E53    "E" , "S"
WM024=H0000   NULL
]
LD      X00404
AND     DIF44
[
ADRIO(WR0000, WM010)
ADRIO(WR0001, WM020)
ADRIO(WR0002, WM030)
FUN 44 (WR0000)
]
```

[Program explanation]

1) WM010 or the first character string and WM020 or the second character string are set using the special internal output R7E3 (1 scan is turned ON after the running starts).

2) The character strings are concatenated at the rising edge of X00404 and outputted to WM030 and the following.

```
Execution result:  WM010=H4849     WM020=H482D     WM030=H4849
                   WM011=H5441     WM021=H534E     WM031=H5441
                   WM012=H4348  +  WM022=H5249  ⇨  WM012=H4348
                   WM013=H4900     WM023=H4E53     WM033=H4948
                                   WM024=H0000     WM034=H2D53
                                                   WM035=H4E52
                                                   WM036=H494E
                                                   WM037=H5300
```

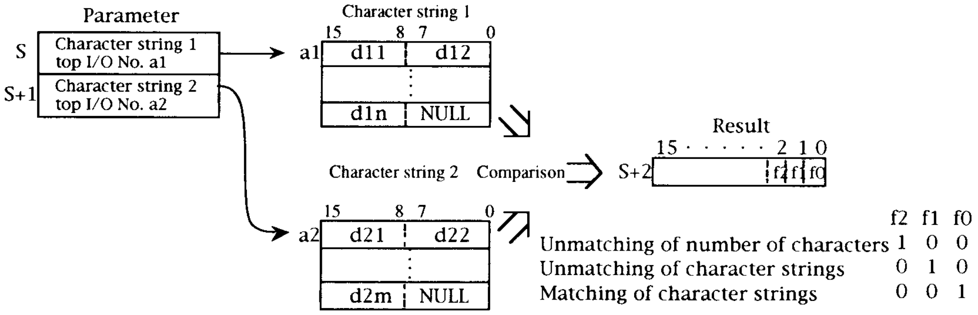
Name	Character String Comparison

Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 45(S) * (SCMP (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	Comparison between 10 characters and 10 characters
	DER	ERR	SD	V	C	501.1~616.6	
	↕	●	●	●	●		
Command format	No. of steps						
FUN 45 (S) * (SCMP (S))	Conditions			Step			
	—			3			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
S	Character string 1 top I/O No.							○						The real addresses are set in S and S+1. S to S+2 are used.
S+1	Character string 2 topI/O No.							○						

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

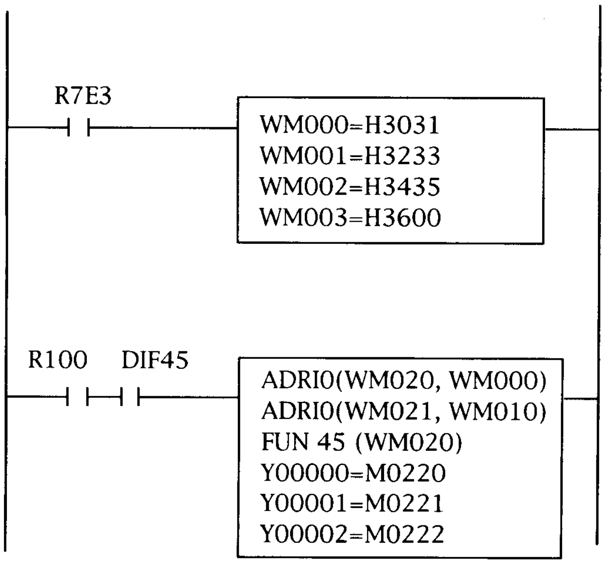


- The character string starting with the top I/O specified by S is compared with the character string starting with the top I/O specified by S+1 and the result is set in S+2.
- The character string to be compared is assumed to end prior to null (H00).
- Set the real addresses in the top I/O of S and S+1 using the ADRIO instruction.
- The character strings are compared with each other in matching of the number of characters.When the numbers of characters are not matched with each other, 1 is set in bit 2. When they are matched, the character strings are compared with each other. When the character strings are not matched with each other, 1 is set in bit 1. When the numbers of characters and the character strings are matched respectively, 1 is set in bit 0.
- When operations are performed normally, DER = 0.

[Precautions]

- Set the real addresses in S and S+1 using the ADRIO instruction. When any other addresses are set, DER = 1 and no processing is performed.
- When S and S+1 or the areas specified by them are overlapped, DER = 1 and no processing is performed.
- When S+1 and S+2 or the areas specified by S and S+1 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

[Program example]



```
LD      R7E3
[
WM000=H3031
WM001=H3233
WM002=H3435
WM003=H3600
]
LD      R100
AND     DIF45
[
ADRIO(WM020, WM000)
ADRIO(WM021, WM010)
FUN 45 (WM020)
Y00000=M0220
Y00001=M0221
Y00002=M0222
]
```

[Program explanation]

1) The comparison data is set in WM000 and the following using the special internal output R7E3 (1 scan is turned ON after the running starts).

2) The data of WM000 and the following is compared with the data of WM010 and the following at the rising edge of R100.

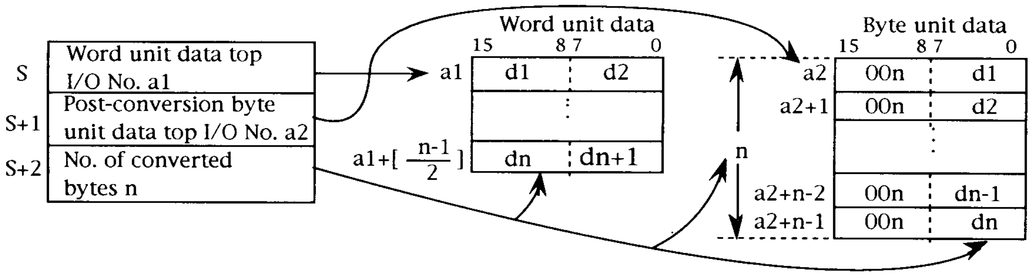
3) Y0000 to Y00002 are turned ON depending on the results.

<table>
<tr><td>Name</td><td colspan="7">Word Unit → Byte Unit Conversion
(Conversion Words to Bytes)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 46 (S)
* (WTOB (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="7">5 words to 10 bytes (word)</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="6">543.9~768.9</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">FUN 46 (S)
* (WTOB (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>s</td><td>Word data top I/O No.</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 rowspan="3">The real addresses are set in S and S+1.
S to S+2 are used.</td></tr>
<tr><td>s+1</td><td>Byte conversion data top I/O No.</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>s+2</td><td>No. of converted bytes</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></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>

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]



- The word character string data of the top I/O specified by S is split byte by byte for the number of bytes specified by S+2 and set in the top I/O area specified by S+1.
- Set the real addresses in the top I/O of S and S+1 using the ADRIO instruction.
- H00 is set in the high-order bytes of each split data.
- When operations are performed normally, DER = 0.

[Precautions]

- Set the real addresses in S and S+1 using the ADRIO instruction. When any other addresses are set, DER = 1 and no processing is performed.
- When S to S+1 or the areas specified by them are overlapped, DER = 1 and no processing is perfomed.
- When S+1 and S+2 or the areas specified by S to S+2 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

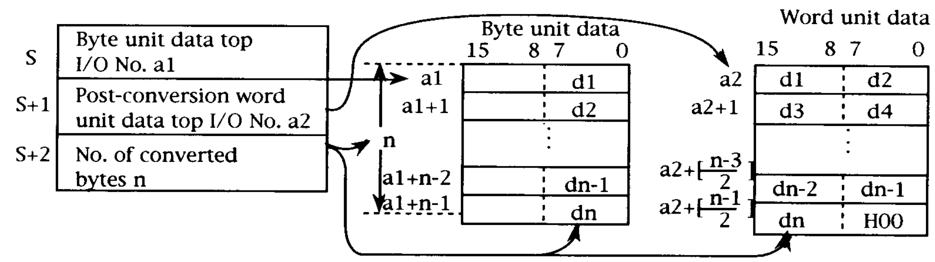
Name	Byte Unit → Word Unit Conversion (Conversion Bytes to Words)

Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 47 (S) * (BTOW (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	10 bytes (word) to 5 words
	DER	ERR	SD	V	C	370.7~595.7	
	↕	●	●	●	●		
Command format	No. of steps						
FUN 47 (S) * (BTOW (S))	Conditions			Step			
	—			3			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
S	Byte unit data top I/O No.							○						The real address are set in S and S+1.
S+1	Word unit data top I/O No.							○						
S+2	No. of converted bytes							○						S to S+2 are used.

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]



- The byte unit character string data in the number of bytes specified by S+2 is combined word by word from the data of the top I/O specified by S and set in the top I/O area specified by S+1.
- The high-order bytes of the byte unit data are ignored.
- When the number of converted bytes is odd, the low-order 8 bits at the end of the output destination are H00.
- Set the real address in the top I/O of S and S+1 using the ADRIO instruction.

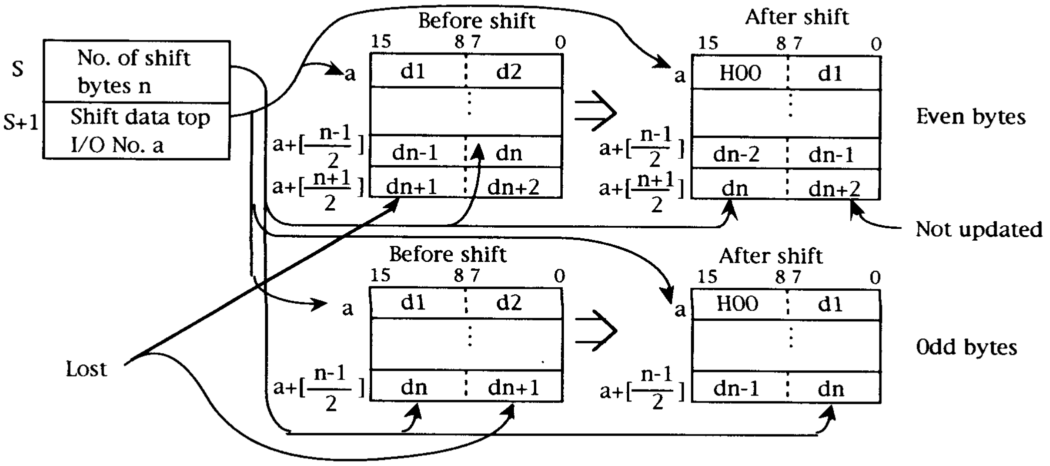
[Precautions]
- Set the real addresses in S and S+1 using the ADRIO instruction. When any other addresses are set, DER = 1 and no processing is performed.
- When S to S+2 or the areas specified by them are overlapped, DER = 1 and no processing is performed.
- When S+1 and S+2 or the areas spedified by S to S+2 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="7">Byte Right Shift</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 48 (S)
* (BSHR (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="7">Shifting 10 characters by 1 byte</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">266.4~416.4</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 48 (S)
* (BSHR (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R,
L,
M</td><td>TD, SS, WDT,
MS, TMR, CU,
RCU, CT</td><td>WX</td><td>WY</td><td>WR,
WL,
WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR,
DL,
DM</td></tr>
<tr><td>s</td><td>No. of shift bytes</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>S and S+1 are used.</td></tr>
<tr><td>s+1</td><td>Shift data top I/O No.</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>The real address is set.</td></tr>
<tr><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></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>

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

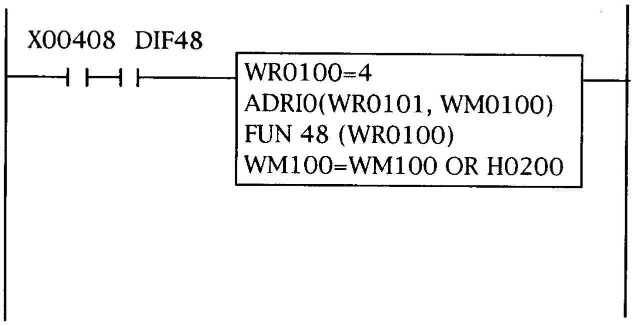


- The data in the number of bytes specified by S is shifted right one byte position from the top I/O data specified by S+1.
- In the empty area after shifting, H00 is set. The next data of the specified number of bytes is lost by shifting.
- Set the real address in the top I/O of S+1 using the ADRIO instruction.
- When operations are performed normally, DER = 0.

[Precautions]

- Set the real address in S+1 using the ADRIO instruction. When any other address is set, DER = 1 and no processing is performed.
- When S and S+1 or the areas specified by them are overlapped, DER = 1 and no processing is performed.
- When S+1 or the areas specified by S and S+1 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

[Program example]



```
WR0100=4
ADRIO(WR0101, WM0100)
FUN 48 (WR0100)
WM100=WM100 OR H0200
```

```
LD      X00408
AND   DIF48
 [
WR0100=4
ADRIO(WR0101, WM0100)
FUN 48 (WR0100)
WM100=WM100 OR H0200
  ]
```

[Program explanation]

- 4-byte transmission data is stored in WM100 and the following. A communication control code of H02 (STX) is added at the beginning of the data.

Execution result:

WM100	"T"	"E"
WM101	"X"	"T"
WM102		

⇨

WM100	H02	"T"
WM101	"E"	"X"
WM102	"T"	

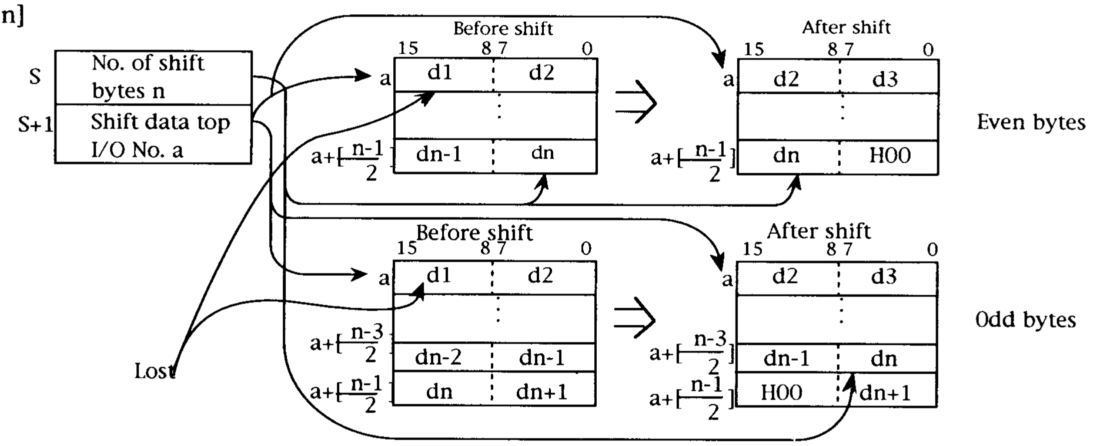
Name	Byte Left Shift

<table>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 49 (S)
* (BSHL (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="6">10 characters</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">240.2~390.2</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 49 (S)
* (BSHL (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	No. of shift bytes							○						S and S+1 are used.
s+1	Shift data top I/O No.							○						The real address is set.

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

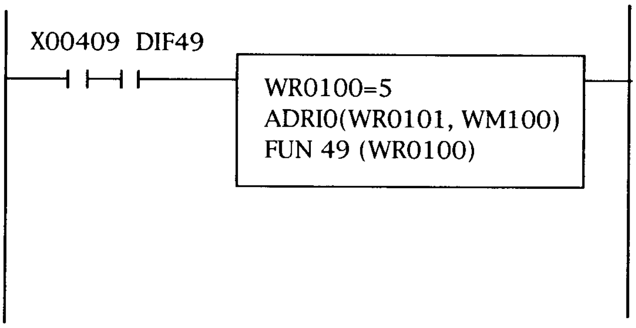


- The data in the number of bytes specified by S is shifted left one byte position from the top I/O data specified by S+1.
- In the empty area after shifting, H00 is set. The begininnig data is lost by shifting.
- Set the real address in the top I/O of S+1 using the ADRIO command.
  ADRIO (S+1, data top I/O)
- When operations are performed normally, DER = 0.

[Precautions]

- Set the real address in S+1 using the ADRIO instruction. When any other address is set, DER = 1 and no processing is performed.
- When S and S+1 or the areas specified by them are overlapped, DER = 1 and no processing is performed.
- When S+1 or the areas spedified by S and S+1 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

[Example program]



```
LD      X00409
AND     DIF49
[
WR0100=5
ADRIO(WR0101, WM100)
FUN 49 (WR0100)
]
```

[Progrm explanation]

- 5-byte data with a control code is stored in WM100 and the following. The control code is removed from the data and only the data forms a data string

Execution result:

WM100	H02	"T"
WM101	"E"	"X"
WM102	"T"	

⇨

WM100	"T"	"E"
WM101	"X"	"T"
WM102	H00	

<table>
<tr><td>Name</td><td>Sampling Trace Set</td></tr>
</table>

<table>
<tr><th>Ladder format</th><th colspan="5">Condition code</th><th>Processing time (μ s)</th><th>Remarks</th></tr>
<tr><td rowspan="3">FUN 50 (S)
* (TRSET (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="6">Relevant commands :
FUN 51 (S)
FUN 52 (S)</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">153.4 ~ 163.7</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 50 (S)
* (TRSET (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><th colspan="2" rowspan="2">Usable I/O</th><th colspan="4">Bit</th><th colspan="4">Word</th><th colspan="3">Double word</th><th rowspan="2">Constant</th><th rowspan="2">Others</th></tr>
<tr><th>X</th><th>Y</th><th>R, L, M</th><th>TD, SS, WDT, MS, TMR, CU, RCU, CT</th><th>WX</th><th>WY</th><th>WR, WL, WM</th><th>TC</th><th>DX</th><th>DY</th><th>DR, DL, DM</th></tr>
<tr><td>s</td><td>Argument (dummy constant)</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></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></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></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>

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

- The CPU instruction tracing is in the sampling execution enable state.
- When the instruction sampling is put into the execution state, the special internal output R7FD (sampling status) is turned ON.
- The instruction sampling is an end trigger function. When the sampling trace reset command (FUN52(S)) is normally executed, the special internal output R7FC (trigger matching flag) is turned ON. When the sampling trace set command is executed, the special internal output R7FC is turned OFF.

[Precautions]

- The special internal outputs R7FC (trigger matching flag) and R7FD (sampling status) are stored when the power fails.
- The word I/O (WR,WL,WM) specified by the argument (S) is used as a system area. Allocate one word to it. The word I/O (WR,WL,WM) specified as an argument (S) is not affected.
- When argument S is beyond the maximum I/O number, DER=1 and no operation is performed.

Name	Sampling Trace Execution

Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 51 (S) * (TRACE (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	Relevant commands FUN 50 (S) FUN 52 (S)
	DER	ERR	SD	V	C	—	
	↕	●	●	●	●		

Command format	No. of steps	
	Conditions	Step
FUN 51 (S) * (TRACE (S))	—	3

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument (dummy constant)							○						

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

- The set sampling trace data is sampled.
- When the instruction sampling is put into the execution state, the special internal output R7FD (sampling status) is turned ON.

[Precautions]

- The special internal outputs R7FC (trigger matching flag) and R7FD (sampling status) are stored when the power fails.
- The word I/O (WR,WL,WM) specified by the argument (S) is used as a system area. Allocate one word to it. The word I/O (WR,WL,WM) specified as an argument (S) is not affected.
- When argument S is beyond the maximum I/O number, DER=1 and no operation is performed.

Name	Sampling Trace Reset						
Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 52 (S) * (TRRES (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	Relevant commands FUN 50 (S) FUN 51 (S)
	DER	ERR	SD	V	C	120.6 ~ 130.9	
	↕	●	●	●	●		
Command format	No. of steps						
FUN 52 (S) *(TRRES (S))	Conditions			Step			
	—			3			

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
S	Argument (dummy constant)							○						

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

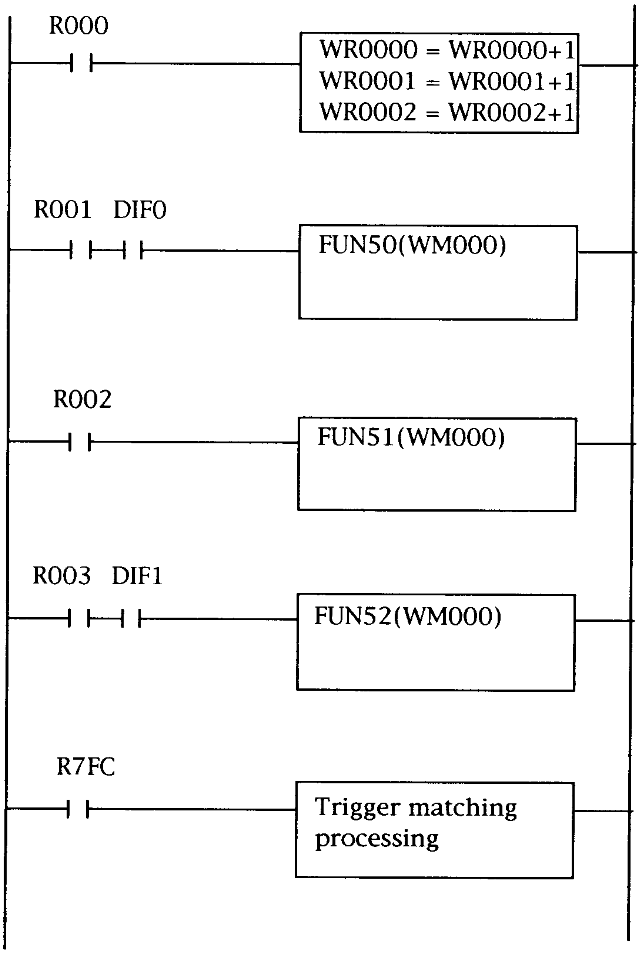
- The CPU instruction tracing is in the sampling execution disable state.
- When the instruction sampling is put into the execution state, the special internal output R7FD (sampling status) is turned ON.
- The command sampling is an end trigger function. When the sampling trace reset command (FUN52(S)) is normally executed, the special internal output R7FC (trigger matching flag) is turned ON.

[Precautions]

- The special internal outputs R7FC (trigger matching flag) and R7FD (sampling status) are stored when the power fails.
- The word I/O (WR,WL,WM) specified by the argument (S) is used as a system area. Allocate one word to it. The word I/O (WR,WL,WM) specified as an argument (S) is not affected.
- When argument S is beyond the maximum I/O number, DER=1 and no operation is performed.

[Program example]

- When WR0002 is sampled from I/O WR0000



```
LD   R000
[
WR0000=WR0000+1
WR0001=WR0001+1
WR0002+WR0002+1
]

LD   R001
AND DIF0
]
FUN 50 (WM000)
[

LD   R002
]
FUN 51 (WM000)
[

LD   R003
AND DIF1
]
FUN 52 (WM000)
[

LD   R7FC
]
Trigger matching
processing
[
```

[Program explanation]

1) The I/O type and number to be sampled from the peripheral equipment (LADDER EDITOR) are set in the CPU.

2) The CPU is operated.

3) When the contact R000 is forced to be set, the status of the I/O to be sample hereafter changes.

4) When the I/O status is sampled according to the sampling data (I/O type and nuumber) which is set by the peripheral equipment, the sampling trace set instruction is activated by forcing the contact R001 to set and the sampling enable status is set.

5) When the contact R002 is forced to be set, the sampling trace instruction is executed and the I/O status is sampled in the sampling buffer which is set by the peripheral equipment.

6) When the sampling is finished, the sampling trace reset instruction is executed by forcing the contact R3 to set and the I/O status is not sampled even if the sampling trace instruction is activated.

For further details, refer to LADDER EDITOR operation manual.

Name	Binary Square Root

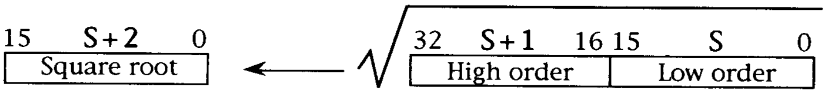
Ladder format	Condition code					Processing time (μ s)	Remarks
FUN 60 (S) * (BSQR (S))	R7F4	R7F3	R7F2	R7F1	R7F0	H-252	
	DER	ERR	SD	V	C	1212.7~1511.0	
	↕	●	●	●	●		

Command format	No. of steps	
	Conditions	Step
FUN 60 (S) * (BSQR (S))	—	3

Usable I/O		Bit				Word				Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR, WL, WM	TC	DX	DY	DR, DL, DM		
s	Argument (low order)							○						
s+1	Argument (high order)							○						S to S+2 are used.

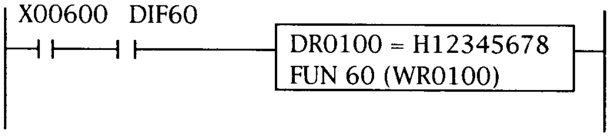
* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]



- The square root of the 32-bit binary value specified by S (low order) and S+1 (high order) is calculated and the result is set in S+2.
- When operations are performed normally, DER = 0.

[Program example]



```
LD     X00600
AND DIF60
 ]
DR0100 = H12345678
FUN 60 (WR0100)
 [
```

[Pogram explanation]

- 32-bit binary data H12345678 is set in WR0100 and WR0101.
- The operation result is set in WR0102.

  Resuult: DR0100 = H12345678

  WR0102 = H4444(17476)

[Precautions]

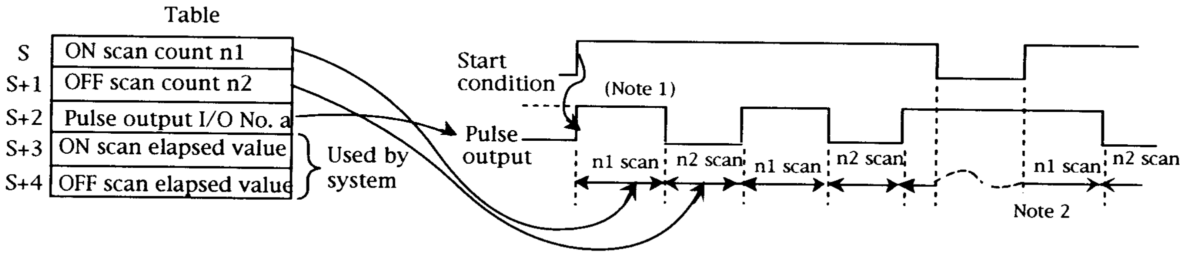
- When S+1 and S+2 are beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

<table>
<tr><td>Name</td><td colspan="8">Dynamic Scan Pulse Command
(Pulse Generating Function)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 61 (S)
* (PGEN (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td colspan="2">H-252</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td colspan="2" rowspan="6">286.5</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">FUN 61 (S)
* (PGEN (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3" rowspan="2">—</td><td colspan="2" rowspan="2">3</td></tr>
<tr></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R,
L,
M</td><td>TD, SS, WDT,
MS, TMR, CU,
RCU, CT</td><td>WX</td><td>WY</td><td>WR,
WL,
WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR,
DL,
DM</td></tr>
<tr><td>s</td><td>ON scan count</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>s+1</td><td>OFF scan count</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>s+2</td><td>Pulse output I/O</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>The real address is set.</td></tr>
<tr><td>s+3
s+4</td><td>System area</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>Cannot be used by the user.</td></tr>
</table>

* The command in parentheses ( ) is for the Ladder Editor.

[Function]



Relation between scan count and pulse operation

Scan count		Pulse operation
n1	n2	
n1 = 0	n2 = 0	The pulse output is turned OFF.
	n2 ≧ 1	
n1 ≧ 1	n2 = 0	The pulse output is turned ON.
	n2 ≧ 1	The pulse output turns n1 scan ON and n2 scan OFF.

- The operation that the bit internal output specified by S+2 is turned ON by the scan count specified by S and OFF by the scan count specified by S+1 is repeated.
- When this command is executed several times during one scan, the output is turned ON or OFF depending on the execution count.
- When both S and S+1 are 0, the output is left OFF.
- When the start condition is turned OFF, the output and the elapsed values of S+3 and S+4 are held.
- When S+3 and S+4 are not cleared at the time of initialization, the pulse width of the first one cycle may be changed.
- Set the real address in the pulse output I/O using the ADRIO instruction.

  ADRIO (S+2, bit internal output)
- When operations are performed normally, DER = 0.

[Precautions]

- The pulse output is changed a maximum of one scan behind the start condition. When the start condition is changed from ON to OFF and then to ON halfway, the pulse width during that time is changed ±1 scan.
- Set the real address in the pulse output I/O of S+2 using the ADRIO instruction. When any other address is set, no processing is performed.
- When S+1 to S+4 or the area specified by S+2 is beyond the maximum I/O number, DER = 1 and no processing is performed.
- In H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

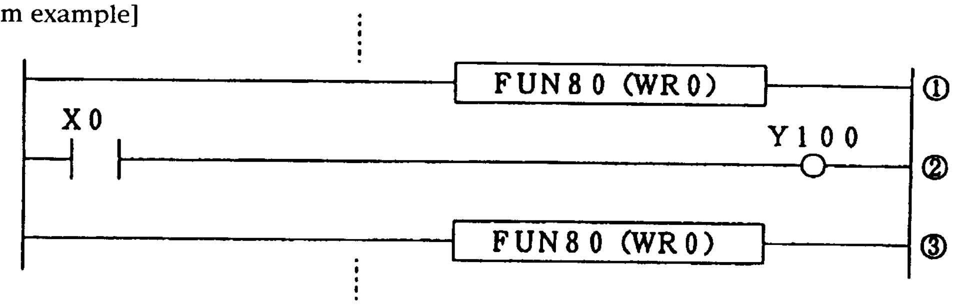
<table>
<tr><td>Name</td><td colspan="7">I/O Refresh (All points)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 80 (S)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">1303</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 80 (S)</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>S</td><td>Argument (Dummy)</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></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></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>

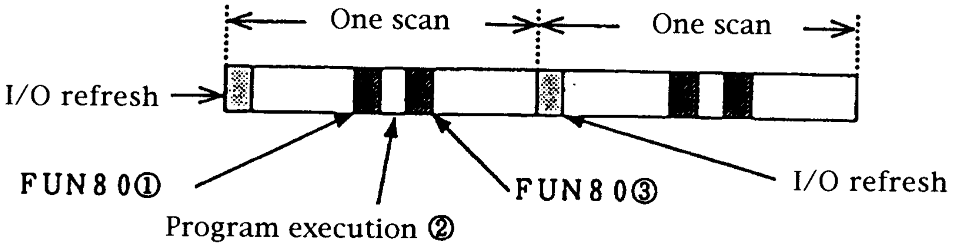
[Function]

- This instruction performs I/O refresh of all data of the external I/O during the scan.

[Program example]



[Program explanation]



[Precaution]

- This instruction performs the refresh of all external I/O.
  When the partial refresh is required, use FUN81 or FUN82.

* This instruction is valid only for H-252.

- Please do not execute the FUN80 instruction in the program which uses the TRNS instruction. The transmitting and receiving data by the TRNS instruction becomes irregular. Please use FUN81 or FUN82 when refreshing is necessary for one scan.

- Please note timing by which the FUN80 instruction is executed in case of mounting of counter module (CTH). The counter module does not operate correctly when the scan time from the FUN80 instruction to the END instruction is 2.5mS or less occasionally.

  Please program FUN80 to the step that the scan time becomes 2.5mS or more or input the program (dummy circuit etc.) so that the scan time may become 2.5mS or more.

<table>
<tr><td>Name</td><td colspan="7">I/O Refresh (for Specified I/O)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 81 (S)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="7">n is the number of all slots assigned.</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">1551+38.5 × n</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 81 (S)</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>S</td><td>Type</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></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></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]

S Type of I/O 00H : Input refresh
01H : Output refresh

- In accordance with area specified in S, the refresh of only the input module or the refresh of only the output module will be made.

- The refresh will be made by a slot unit in accordance with I/O assignment.
  I/O assignment subject to the input refresh : X16, X32, X48, X64, WX4W, WX8W
  I/O assignment subject to the output refresh : Y16, Y32, Y48, Y64, WY4W, WY8W
  For the modules having I/O assignment different from those as listed above, this instruction performs no refresh.

Type of I/O that can not be refreshed by FUN81.

- Mixed I/O : PHH-DT, PHM-TT
- Communication : LINK, REMOTE
- Special : COUNTER, POSITIONING, SERIAL I/O, Thermocouple

- When the refresh processing is executed normally, DER=0 will be made.

[Precaution]

- When the type of I/O is other than 00H and 01H, the processing will not be executed caused by DER=1.

- This instruction is valid only for H-252.

<table>
<tr><td>Name</td><td colspan="7">I/O Refresh (for Voluntary I/O)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 82 (S)</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="6"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">According to the formula shown below</td></tr>
<tr><td>↕</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 82 (S)</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="4">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR, WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>S</td><td>No. of points</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>S+1 and after</td><td>No. of slot position</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>Slot position is specified</td></tr>
<tr><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]

S	No. of refresh points n
S + 1	Refresh slot position No.
S + 2	Refresh slot position No.
⋮	⋮
S + n	Refresh slot position No.

n ≤ 29

Position No. of refresh slot is specified by the unit No. and the slot No.

(Note) Only for the analog module, the channel No. is also specified.

- The specified modules is refreshed by the number as specified by S from the area S+1.
- The refresh is made in a slot unit. (by a channel unit in the case of the analog module.)
- The slot position No. ,to be stored in the area after S+1, is specified by the unit No. and the slot No.

  (Example) The position of the slot 3 in the basic base is expressed as "H0003".
  The position of the slot 2 at 1st stage expansion base is expressed as "H0012".
  To specify the channels 1, 2, 3 of the analog module in the slot position 4 of the basic base is expressed as "H1304".
- Maximum number of the refresh points n is 29.

  For the range exceeding 29 points, the refresh processing is disabled.

  When the refresh processing is excuted normaly, DER becomes"0".
- Processing speed is calculated by the formula:

  (51+(processing speed of each I/O assignment)+10 ∗s) ∗n+284μs

  Where, n: No. of refresh points, S; Slot position (Absolute position from the unit 0 slot 0)

  Processing speed of each I/O assignment is shown below. (C is No.of channels)

X16 :	88μs	Y16 :	94μs
X32 :	37μs	Y32 :	38μs
X48 :	46μs	Y48 :	47μs
X64 :	54μs	Y64 :	55μs
WX4W :	98+111 ∗Cμs	WY4W :	37+109 ∗Cμs
WX8W :	98+111 ∗Cμs	WY8W :	37+109 ∗Cμs

[Precaution]

- For the range after (S+1), set the unit No. (0~2), the slot position No. (0~9), the channel No. (0~7). If other numbers are specified, DER=1 and such slot will not be processed.

  For channel No., when the top channel No. > the tail channel No.,DER=1 and such channel will not be processed.

(Processing will be made for slots and channels that are correctly set.)

- When number of S+ points exceeds the maximum I/O No., processing will not be made caused by DER=1.
- When the number of points exceeds the maximum 29, DER=1 will be made, and the processing will not be made for the range exceeding 29 points. (Refresh processing will be made as far as 29 points are reached.)

  However, in H-252, the external output area of WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, the processing will also be disabled causing DER=1 when the processing exceeds WR3FF.

  * This instruction is valid only for H-252.

[Type of refresh module]

Types of I/O assignment that are able to refresh by this instruction are shown below.

I/O assignment possible to refresh : X16, X32, X48, X64, WX4W*, WX8W*
Y16, Y32, Y48, Y64, WY4W*, WY8W*

* For WX4W, WX8W, WY4W, WY8W, the processing will be made as the analog module.

No refresh will be made for the I/O assignment slots other than those indicated above.

[Slot position No.]

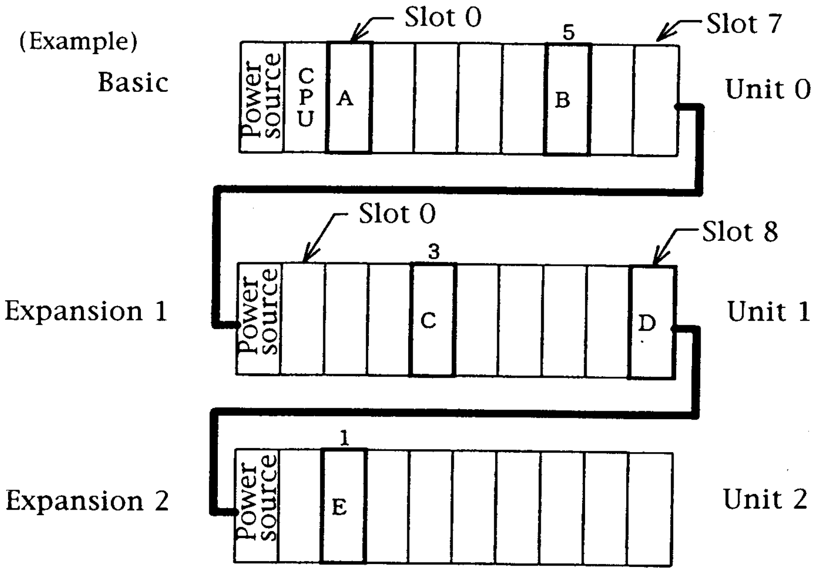
Slot No. is specified by the unit No., the slot No., and the channel No.

Set the unit No., the slot No., and the channel No. so as to accommodate them in one word as shown in the following illustration.

b15	b12	b7	b3
Top channel No.	Tail channel No.	Unit No.	Slot No.

* Channel No. is specified only for the analog module.

Note) Should be: Top channel No. ≤ Tail channel No



In the drawing above, the slot positions from A to E become as shown below.

Position of A : H0000
Position of B : H0705 (Analog module : All channels 0~7 are refreshed)
Position of C : H0013
Position of D : H3318 (Analog module : Only the channel 3 is refreshed)
Position of E : H0021

<table>
<tr><td>Name</td><td colspan="8">BOX COMMENT</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μs)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 254 (S)
* (BOXC (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">15.9</td><td rowspan="5">28.9</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 254 (S)
* (BOXC (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="3">Word</td><td colspan="4">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Other</td></tr>
<tr><td>X</td><td>Y</td><td>R,
L,
M</td><td>TD,SS,WDT,
MS,TMR,CU,
RCU,CT</td><td>WX</td><td>WY</td><td>WR,
WL,
WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR,
DL,
DM</td></tr>
<tr><td>s</td><td>Argument (dummy constant)</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>

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

- This instruction executes nothing but prints comments on the right of the arithmetic box in combination with LADDER EDITOR.
- The maximum number of comment characters is 32.

<table>
<tr><td>Name</td><td colspan="8">MEMO COMMENTS</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μs)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 255 (S)
* (MEMC (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td>H-250</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="5">15.9</td><td rowspan="5">28.9</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 255 (S)
* (MEMC (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="3">Word</td><td colspan="4">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Other</td></tr>
<tr><td>X</td><td>Y</td><td>R,
L,
M</td><td>TD,SS,WDT,
MS,TMR,CU,
RCU,CT</td><td>WX</td><td>WY</td><td>WR,
WL,
WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR,
DL,
DM</td></tr>
<tr><td>s</td><td>Argument (dummy constant)</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>

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

- This command executes nothing but prints comments between the circuits under the arithmetic box in combination with LADDER EDITOR.
- A maximum of one screen (66 characters × 16 lines) can be used for comments.

3.4 PID function

The H-252 has instructions for PID control.

For the outline of the PID function, refer to Section 3.4 of the hardware edition.

1. PID operation instruction

There are three PID instructions available as shown below.

* The commands in parentheses ( ) are for the LADDER EDITOR.

FUN 0 (S) * (PIDIT (S))	Initialization of the PID operation function
FUN 1 (S) * (PIDOP (S))	Execution management of PID operation
FUN 2 (S) * (PIDCL (S))	Execution of PID opertion

2. Details of the PID instructions

Details are shown on the next and subsequet pages.

<table>
<tr><td colspan="2">Name</td><td colspan="6">PID Initialization (PID INITIALIZE)</td></tr>
<tr><td colspan="2">Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td colspan="2" rowspan="3">FUN 0 (S)
* (PIDIT (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="7">n
= Loop count</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="6">1501.6+234.4 x n</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td colspan="2">Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td colspan="2" rowspan="3">FUN 0 (S)
* (PIDIT (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="5">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR</td><td>WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>s</td><td>PID management table</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></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></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>

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

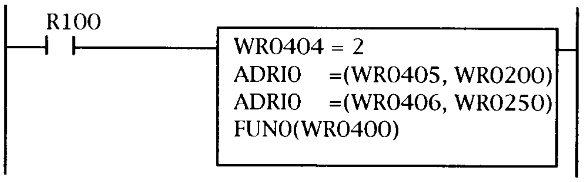
- FUN0(S) initializes the area for storing the initialization data necessary for PID operations.
- S of FUN0(S) specifies the WR top No. of the PID management table.
- When any error is found in the contents shown in the PID management table, an error code is set in Error Code 0 of the PID management table and the area is not initiialized.
- When FUN0 is executed once again after the area is correctly initialized (FUN0 normal termination (1) of the PID management table)once, an error is caused. (However, the normal end flag remains "1").

For the PID management table, loop word table, and loop bit table, see Section 3.4(3), " Details of PID management table ".

[Precauutions]

If a control fault occurs when the area to be used for the PID operation is cleared at the time of operation start or start after power failure, specify Power Failure Storage.

[Program example]



```
WR0404 = 2
ADRIO   =(WR0405, WR0200)
ADRIO   =(WR0406, WR0250)
FUNO(WR0400)
```

1) The loop count is specified as 2.
2) } The real address of the WR top I/O of
3) } the loop word table area is aet.
4) The PID area is initialized.

[Program explanation]

The following is a program example that WR0400 and the following of the PID managementtable are used, the loop count is set to 2, the loop 1 word table area ranges from WR0200 to WR022F (48 words), and the loop 2 word table area ranges from WR0250 to WR027F (48words).

PID management table

WR0400	Error code 0	(R)	
WR0401	Error code 1	(R)	
WR0402	Error code 2	(R)	
WR0403	FUN 0 normal termination	(R)	
WR0404	Loop count	(W)	WR404 = 2
WR0405	Real address of WR top No. of the loop 1 word table	(W)	ADRIO = (WR405, WR0200)
WR0406	Real address of WR top No. of the loop 2 word table	(W)	ADRIO = (WR406, WR0250)

(R) Word for reading.
(W) Word for setting and reading

When FUNO (WR0400) is executed, the areas from WR0200 to WR022F and from WR0250 to WR027F are checked and initialized.

<table>
<tr><td>Name</td><td colspan="7">PID Execution Management (PID OPERATION CONTROL)</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td>Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 1 (S)
* (PIDOP (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td>H-252</td><td rowspan="7"></td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td rowspan="6">454.4</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="3">FUN 1 (S)
* (PIDOP (S))</td><td colspan="4">Conditions</td><td>Step</td></tr>
<tr><td colspan="4">——</td><td>3</td></tr>
</table>

Usable I/O		Bit				Word					Double word			Constant	Others
		X	Y	R, L, M	TD, SS, WDT, MS, TMR, CU, RCU, CT	WX	WY	WR	WL, WM	TC	DX	DY	DR, DL, DM		
s	PID management table top							○							

* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

- FUN1(S) reads the PID execution flag and the PID constant change flag of the loop bit table area and determines the loop for performing operations.
- Set the WR top No. of the PID management table in S of FUN1(S). When any other number is set, an error is caused. An error code is set in Error code is 0 and 1 of the PID management table and FUN1 is not executed.
- Create a program so that FUN1(S) is executed once during the periodic scan of 20 ms.

For the PID management table, loop word table, and loop bit table, see Section 3.4(3), " Details of PID management table ".

<table>
<tr><td>Name</td><td colspan="8">PID CALCULATION</td></tr>
<tr><td>Ladder format</td><td colspan="5">Condition code</td><td colspan="2">Processing time (μ s)</td><td>Remarks</td></tr>
<tr><td rowspan="3">FUN 2 (S)
* (PIDCL (S))</td><td>R7F4</td><td>R7F3</td><td>R7F2</td><td>R7F1</td><td>R7F0</td><td colspan="2">H-252</td><td rowspan="6">n = Loop count</td></tr>
<tr><td>DER</td><td>ERR</td><td>SD</td><td>V</td><td>C</td><td colspan="2" rowspan="5">2065+17.5 × n</td></tr>
<tr><td>●</td><td>●</td><td>●</td><td>●</td><td>●</td></tr>
<tr><td>Command format</td><td colspan="5">No. of steps</td></tr>
<tr><td rowspan="2">FUN 2 (S)
* (PIDCL (S))</td><td colspan="3">Conditions</td><td colspan="2">Step</td></tr>
<tr><td colspan="3">—</td><td colspan="2">3</td></tr>
</table>

<table>
<tr><td colspan="2" rowspan="2">Usable I/O</td><td colspan="4">Bit</td><td colspan="5">Word</td><td colspan="3">Double word</td><td rowspan="2">Constant</td><td rowspan="2">Others</td></tr>
<tr><td>X</td><td>Y</td><td>R, L, M</td><td>TD, SS, WDT, MS, TMR, CU, RCU, CT</td><td>WX</td><td>WY</td><td>WR</td><td>WL, WM</td><td>TC</td><td>DX</td><td>DY</td><td>DR, DL, DM</td></tr>
<tr><td>s</td><td>Word table top</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></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></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>

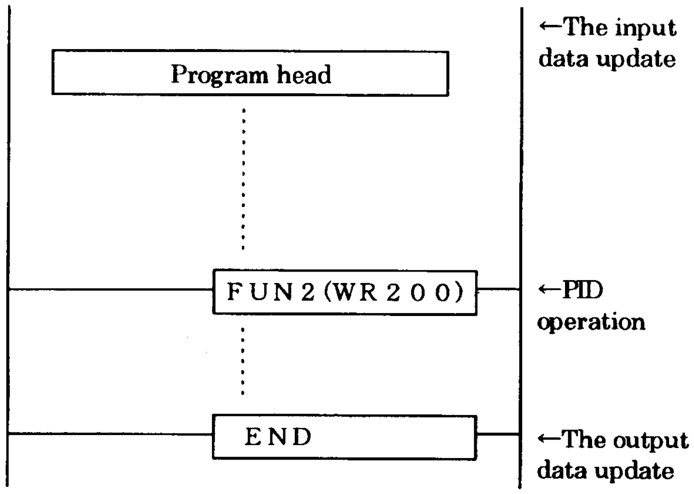
* The command in parentheses ( ) is for the LADDER EDITOR.

[Function]

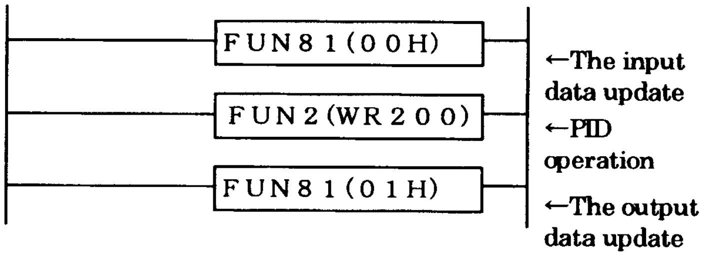
- Whether or not to calculate PID is determined by the sampling time set in each loop word table.
- FUN2(S) turns the loop PID calculation flag to be calculated ON.
- Set all the WR top numbers of each PID loop word table by FUN2(S).
- FUNS(2) checks each loop output upper llimit value, lower limit value, set value bit pattern range, and output value bit pattern range. When an error occurs, the FUN2 error flag of the loop bit table is turned ON and an error code is set in Error Code 2 of the PID management table. Even if an error occurs, the FUN2 processing is continued.
- Create a program so that FUN2(S) is executed once during the periodic scan of 20 ms.

For the PID management table, loop word table, and loop bit table, see Section 3.4(3), " Details of PID management table ".

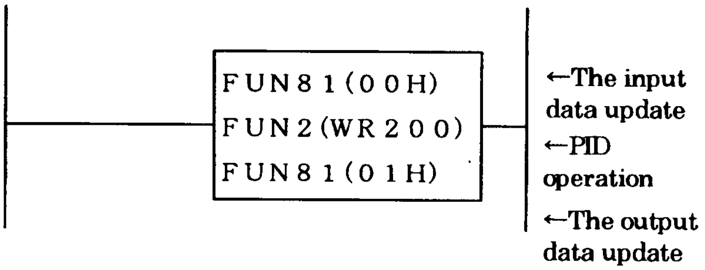
- When only the FUN2 instruction is programmed when PID is processed, the I/O data is updated by the refreshing processing executed by the END instruction.



- The response quickens by assuming the program which combines refreshing instructions which are FUN81 or FUN82 with FUN2 as follows when it is necessary to respond faster.



or



- Please specify the channel which wants to be refreshed in the I/O No. from S+1 in case of using FUN82(S).

3. Details of PID management table.

(a) Explanation for an example.

[Example] FUN0 (WR0400)

{1} PID management table

WR0400	Error code 0 (Read)	{2}
WR0401	Error code 1 (Read)	
WR0402	Error code 2 (Read)	
WR0403	Normal termination 1 of FUN0 (Read)	{5}
WR0404	Number of loops (Write)	{3}
WR0405	Top address of word table (WR) for the loop 1 (Write)	{4}
WR0406	Top address of word table (WR) for the loop 2 (Write)	
WR0407	Top address of word table (WR) for the loop 3 (Write)	
⋮	⋮	
WR0444	Top address of word table (WR) for the loop 64 (Write)	

{1} PID management table

1. "S" of FUN0(S) should be the top number of WR to be used for the PID management table.

2. The PID management table consists of the four parts: {2}, {3}, {4}, {5}. The size of the table will be increased with the increase in number of the loops {3}. The address must not exceed the maximum permissible number of WR(WR43FF) *. If the number of the loops exceeds this limit ,the error code H0004 will be written on "{2} Error code 0".

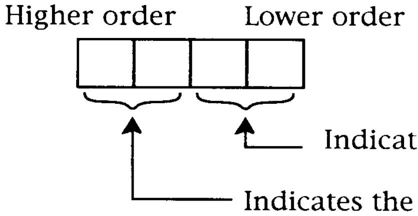
   Avoid simultaneous assignment on both areas WR0 ~ 3FF and WR400 ~ 43FF.

   [Note] (Write) in the above table means the area for writing and reading by a user's program.

   * In H-252, the WR is divided into the two areas, WR0~3FF and WR400~43FF. Therefore, an error will also be caused when the processing the processing exceeds WR3FF.

{2} Error code 0, 1, 2 (Read only)

1. Error code 0 ---An error code will be set in this area when the error occurs in the execution of FUN0 processing or FUN1 (partly) processing. While no error is detected the area will hold the last status.
2. Error code 1 ---An error code will be set in this area when the error occurs in the execution of FUN1 processing. While no error is detected the area will hold the last status.
3. Error code 2 ---An error code will be set in this area when the error occurs in the execution of FUN2 processing. While no error is detected the area will hold the last status.
4. An error code is expressed by a hexadecimal number in 4 digits.



Indicates the cause of the error.

Indicates the loop number.
H00: Error not related to loop numbers.
H01~ H40: Error occurred in a loop with its number announced.

For details of the error code 0, 1 and 2, refer to "(6) Details of PID error codes". In the table for the error codes, the upper 2 digits denoted by" xx "will accommodate a loop number in an error.

{3} Number of loops (Write)

Set the number of loops used (1 to 64).

When "0" is set in this area, the error code 0002 will be set in the area "{2} Error code 0" and PID control function will be disabled (even when FUN1 and FUN2 are already coded).

{4} Address of (WR) for each loop of the word table (Write)

1. One loop requires a WR of 48 words long for an input of PID constants and for the internal calculation of PID. Set the top real address of the WR in this area by ADRIO instruction. The address must not exceed WR43FF. If the address exceeds WR43FF, the error code xx05 will be written in the area "[2] Error code 0". Avoid double numbering of WR.
2. By using ADRIO instruction, set the top numbers of WR addresses as many as the number of loops set in the area {3} Number of loops to be specified.

For instance, when you write "5" in the area {3} Number of loops, set the address of WR for the loop 1 to the address of WR for the loop 5 (a total of 5 addresses).

For the ADRIO instruction, refer to 3.3 (3) for the application commands.

{5} Normal termination of FUNO (Read only)

Code 0001 will be set in this area when the FUNO command ends normally. When an error occurs in the execution of the FUNO command, an error code will be set in the area "{2} Error code 0".

(b) The word table and the bit table for each loop.
This example has two control loops, Loop 1 and Loop 2.
A word table and a bit table are necessary for each loop.
The followings are the details of the tables.

{1} PID management table

WR0400	Error code 0	{2}
WR0401	Error code 1	
WR0402	Error code 2	
WR0403	Normal end flag, FUN0	{5}
WR0404	2 (number of loops)	{3}
WR0405	ADRIO (WR0405, WR0200)	{4}
WR0406	ADRIO (WR0406, WR0250)	

Bit table for the loop 1
(The top address of this bit table is given in the area {11})

Word table for the loop 1

Bit table for the loop 1

WR0200	ADRIO (WR0200, R100)		{11} ←	R100	Execution flag	(W)	{50}
WR0201	Sampling time TZ	(W)	{12}	R101	Non bumpless flag	(W)	{51}
WR0202	Proportional gain KP	(W)	{13}	R102	Change flag, PID constants	(W)	{52}
WR0203	Integral constant TI/TZ	(W)	{14}	R103	S flag	(W)	{53}
WR0204	Differential constant TD/TZ	(W)	{15}	R104	R flag	(W)	{54}
WR0205	Differential lag constant Tn/TZ	(W)	{16}	R105	D-FREI flag	(W)	{55}
WR0206	Output upper limit UL	(W)	{17}	R106			
WR0207	Output lower limit LL	(W)	{18}	R107			
WR0208	Initial value INIT	(W)	{19}	R108	PID RUN flag	(R)	{58}
WR0209	I/O No. of the set value (Address)	(W)	{20}	R109	PID in operation flag	(R)	{59}
WR020A	I/O No. of the measuring value (Address)	(W)	{21}	R10A	PID constant OK flag	(R)	{60}
WR020B	I/O No. of the output value (Address)	(W)	{22}	R10B	Over high limit flag	(R)	{61}
WR020C	Bit pattern of the set value	(W)	{23}	R10C	Under low limit flag	(R)	{62}
WR020D	Bit pattern of the measuring value	(W)	{24}	R10D	Error flag, FUN2	(R)	{63}
WR020E	Bit pattern of the output value	(W)	{25}	R10E			
WR020F	Operation area for PID (Not to be used by the user)			R10F			
	⋮		{26}				
WR022F	Operation area for PID (Not to be used by the user)						

Bit table for the loop 2

WR0250	ADRIO (WR0250, R110)	{11} ←	R110	Execution flag	(W)
WR027F	Same as for the loop 1	The word table for loops	⋮	Same as for the loop 1	
	⋮		R11F	⋮	

(W) Bits or words for setting or writing.
(R) Bits or words for reading.

(c) Details of the word table for each loop.

{11} Address of the Top of the Bit Table for Loop (Write)

"R" of 16 bits is used for controlling and monitoring each loop. Specify the address by the top No. (0 ~ R7B0) of "R" using the ADRIO instruction. The number (address) must not exceed R7BF (the end of R).

Set {12} to {19} within the specified by the following table.

<table>
<tr><th>No.</th><th>Parameters</th><th>Specification</th><th>Remarks</th></tr>
<tr><td>{12}</td><td>Sampling Time
T Z</td><td>1 to 200 (time is T Z × 20ms)
(When the analog input output module is installed on the basic or expansion unit)</td><td>○ The sampling time must be a multiple of a minimum set value.
○ The minimum set value is what is set in the Number of Loops area {3}.
(Example) Setting of samplinng times (TZ) of 4 loops
<table>
<tr><td>Loop 1</td><td>4</td><td>4</td><td>8</td><td>4</td><td>0 *</td><td>3 *</td></tr>
<tr><td>Loop 2</td><td>4</td><td>4</td><td>24</td><td>4</td><td>4</td><td>12</td></tr>
<tr><td>Loop 3</td><td>4</td><td>8</td><td>16</td><td>4</td><td>0 *</td><td>48</td></tr>
<tr><td>Loop 4</td><td>4</td><td>8</td><td>40</td><td>10 *</td><td>4</td><td>24</td></tr>
<tr><td></td><td>Ex. 1</td><td>Ex. 2</td><td>Ex. 3</td><td>Ex. 4</td><td>Ex. 5</td><td>Ex. 6</td></tr>
</table>
Ex.1 to Ex.3: Valid
Ex.4 to Ex.6: Not Valid (*)</td></tr>
<tr><td>{13}</td><td>Proportional Gain KP</td><td>-1000 to +1000</td><td>Corresponds to -10.00 to +10.00</td></tr>
</table>

No.	Parameters	Specification	Remarks
{14}	Integral Constant T_I / TZ	1 to 32767	Set each time (T_I, T_D, Tn) divided by sampling time (TZ × 20ms)
{15}	Differential Constant T_D / TZ		
{16}	Differential lag Constant T_n / TZ		
{17}	Output upper limit UL	-32767 to +32767	These values must satisfy the following condition: LL ≦ INIT ≦ UL
{18}	Output low er limit LL		
{19}	Initial Value INIT		

{20} I/O No. of the set value (Write)

This area sets the real address of I/O word No. which stores a set value.

(Example) When a set value is set in WX500 and the following ADRIO (WR0209, WX0500)

{21} I/O No. of the measuring value (Write)

This area sets the address of I/O word No. which stores the measuring value.

(Example) ADRIO (WR020A, WX0510)

{22} I/O No. of the output value (Write)

This area sets the address of I/O word No. from which the result of a PID operation will be output.

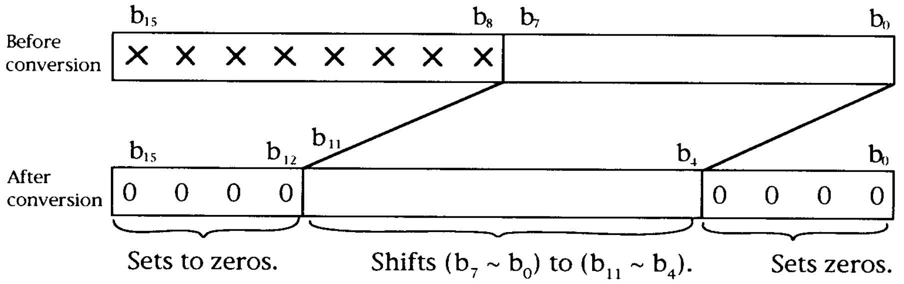
(Example) ADRIO (WR020B, WY0520)

{23} Bit Pattern of the set value (Write)

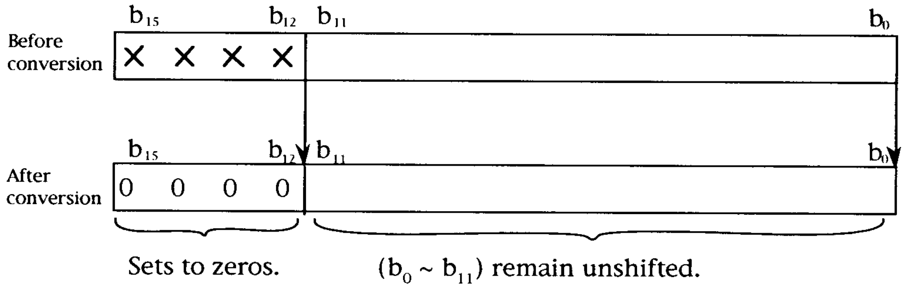
This area determines a method of coverting a set value into 16-bit data for PID operation.

Select and write one of H0001 to H0004 according to the conditions shown below. See Table 3.4.1 for bits patterns for analog input modules.

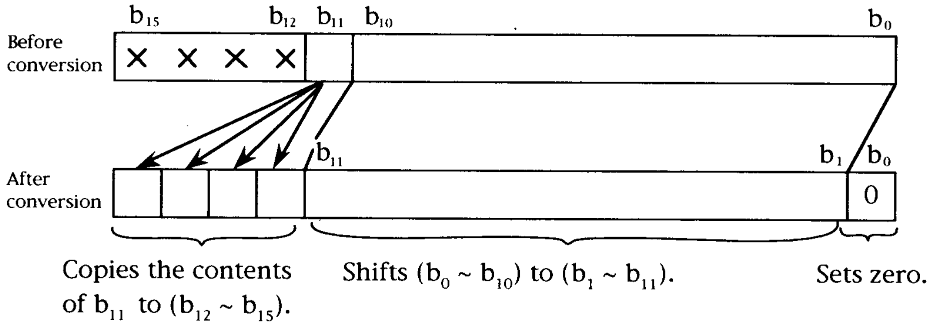
- H0001: 8 bits to 16 bits



- H0002: Unsigned 12 bits to 16 bits



- H0003: Signed 12 bits to 16 bits (Sign expansion)



- H0004: No conversion

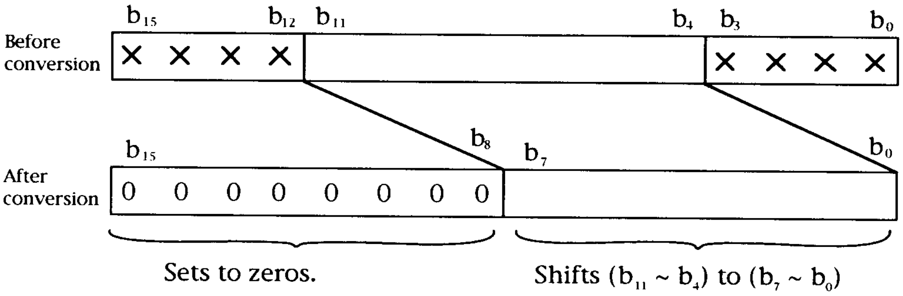
{24} Measuring Value Bit Pattern (Write)

This area sets one method of H0001 to H0004 of converting data read from the Measuring Value Address area {21} into 16-bit data. See Table 3.4.1 for bit pattern for analog input modules. See {23} " Set Value Bit Patterns" for the converting methods.

{25} Output Value Bit Patterns (Write)

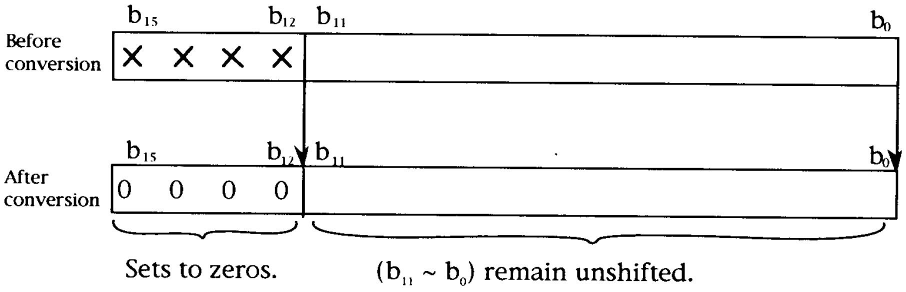
The result of FUN2 processing or PID operation is converted according to the bit pattern set in this area and written in the Output Value Address area {22} .See Table 3.4.1 for bit patterns for analog output modules.Select and write one of H0001 to H0004 in this area {25} according to the type of the {22} Output Value I/O (Analog Output WY, WR, WL, or WM), as shown below.

- H0001: 16 bits to 8 bits



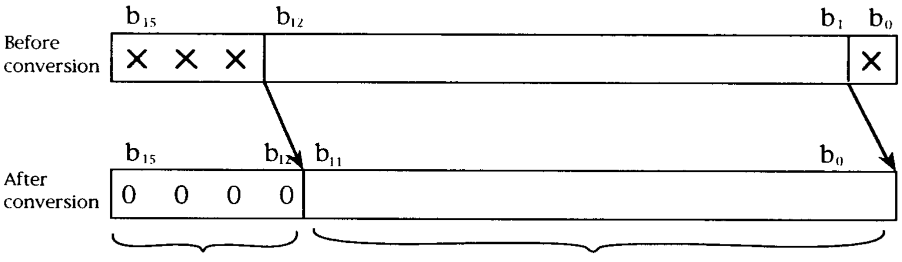
HOFFF to H7FFF (before conversion) is converted to HOOFF.
H8000 to HF000 (before conversion) is converted to H0000.

- H0002: 16 bits to Unsigned 12 bits



HOFFF to H7FFF (before conversion) is converted to HOOFF.
H8000 to HF000 (before conversion) is converted to H0000.

- H0003: Signed 16 bits to Signed 12 bits



Sets to zeros.　　Shifts (b_1 ~ b_{12}) to (b_0 ~ b_{11})

HOFFF to H7FFF (before conversion) is converted to H07FF.

H8000 to HF000(before conversion) is converted to H0800.

- H0004: No coversion

(Note) Bit patterns for {23} Set value, {24} Measuring value and {25} Output value vary depending on a type of an analog I/O module that are subject to the reading and writing of these values. Assign in accordance with Table 3.4.1.

Table 3.4.1 Analog I/O modules and bit patterns

Module	Type	Specification	Set value •Measuring value /Output	Bit pattern
Analog Input Module	AGH-I	4 ~ 20 mA DC, 8 bits	Set value • Measuring value	H0001
	AGH-IV	0 ~ 10V DC, 8 bits		H0001
	AGH-IV2	4 ~ 20 mA DC, 0 ~ 10 VDC, 12 bits		H0002
Analog Output Module	AGH-O	4 ~ 20 mA DC, 8 bits	Output value	H0001
	AGH-OD	4 ~ 20 mA DC, 8 bits		H0001
	AGH-OV	0 ~ 10 VDC, 8 bits		H0001
	AGH-ODV	0 ~ 10 VDC, 8 bits		H0001

{26} Work area for PID Operation (Not to be used by the user)

The FUN0, FUN1, and FUN2 processings use this area for PID operations. The user program is not allowed to use this area.

If the user program uses this area, the result of the PID operation will not be assured.

{50} Execution flag (Write)

1. When the Execution flag turns on ("0" to "1"), the PID constants at the time are checked and the PID calculation values are initialized. When the PID constants are all valid, the PID RUN flag {58} is turned on ("1").

   When any PID constant is found invalid, the PID RUN flag {58} is turned off ("0"). The PID operation will not be performed.
2. The PID operation is performed while the Execution flag is on ("1").
3. When the PID RUN flag {58} is turned off ("0"), the PID operation ends and the output becomes 0.

{51} Non-Bumpless flag (Write)

"Non-bumpless" operation is to store the PID calculation result internally without stopping the PID operation when the S or R flag is turned on. Therefore, when the S or R flag is turned off, the output value abruptly changes according to the intermediate operation value.

Contrary, in the Bumpless status, the PID operation is suppressed while the S or R flag is on. When the flag is turned off, the output value changes in sequence starting from the value at the time point. See (5) (c) Time chart example 3.

When the Non-Bumpless = 1, non-bumpless processing will be performed.

When the Non-Bumpless = 0, bumpless processing will be performed.

{52} PID Constant Change Flag (Write)

1. When any of PID constants ({12} to {16}) is changed, the PID Constant Change flag {52} must be turned on. When this flag is turned on, the change of PID constant ({12} to {16}) is reflected on the result (precedent value).
2. When the change of the PID constant is completed, this flag must be set to OFF by the user.
3. When any PID constant is not valid (when the "PID Constant OK Flag 60 is "0"), the PID operation will be performed with the precedent PID constant.

{53} S Flag (Write)

When the S flag is on ("1"), the output value is reset to the initial value. The output value is determined according to the relationship of the initial value {19} , output upper limit {17} and output lower limit {18} , as shown below.

Output lower limit {18} > Output upper limit {17} ....No output

Output lower limit {8} ≦ Initial value {19} ≦ Output upper limit {17}

..............The output value is the Initial value {19 } .

Output lower limit {18} ≦ Output upper limit {17} ≦ Initial value {19}

..............The output value is the Output upper limit {17} .

Initial value {19} ≦ Output lower limit {18} ≦ Output upper limit {17}

..............The output value is the Output lower limit {18} .

The S flag is given a higher precedence than the R flag.

{54} R Flag (Write)

When this flag is on ("1"), the output value is cleared to all zeros.

{55} D-FREI Flag (Write)

1. When this flag is on ("1"), the P, I, and D operations are performed. (The differentiating operation and integrating operation will be performed.)
2. When this flag is off ("0"), the P, and I operations are performed. (The differentiating operation will not be performed.)

In the case of D-FREI = 1, PID operation will be performed by differentiating and integrating operations.

$$\text{The output} = \frac{KP^{\{13\}}}{100} \bullet XD + \frac{KP^{\{13\}}}{100} \bullet \frac{XD}{T_I^{\{14\}}/TZ} + Y1(n-1)$$

$$+ \frac{Tn^{\{16\}}\ /TZ}{1+Tn^{\{16\}}/TZ}\ YDT1\ (n-1) + \frac{T_D^{\{15\}}/TZ}{Tn^{\{16\}}/TZ} \times \frac{KP^{\{13\}}}{100}\ (XD-XD(n-1))$$

In the case of D-FREI = 0, PI operation will be performed without differentiating operations.

$$\text{The output} = \frac{KP^{\{13\}}}{100} \bullet XD + \frac{KP^{\{13\}}}{100} \bullet \frac{XD}{T_I^{\{14\}}/TZ} + Y1(n-1)$$

Where;
XD is the deviation: XD = Set value - Measuring value
KP : Proportional gain
Tn / TZ: Differentiating constant
TI/TZ : Integrating constant
XD (n-1) : The precedent deviation
Y1 : Integrated value

$$Y1 = \frac{KP^{\{13\}}}{100} \bullet \frac{XD}{T_I^{\{14\}}/TZ} + Y1(n-1)$$

Y1 (n-1) : The precedent integrated value
YDT1 : Differentiated value

$$YDT1 = \frac{Tn^{\{16\}}/TZ}{1+Tn^{\{16\}}/TZ}\ [YDT1\ (n-1) + \frac{T_D^{\{15\}}/TZ}{Tn^{\{16\}}/TZ} \times \frac{KP^{\{13\}}}{100}\ (XD(n-1))]$$

YDT1 (n-1): The precedent differentiated value

Do not change the content of the flags ({58} to {63}) (for reference purposes only). If changed, the result of the PID operation may be impaired.

{58} PID RUN Flag (Read-only)

1. When FUN1 detects that the Execution flag {50} is on, the validities of the parameters {12} to {16} and {20} to { 22} will be checked and the result will be set in the PID RUN Flag {58} .

   PID RUN Flag = "1" : Valid ....The PID operation starts.

   ="0" : Invalid.......The PID will not be operated. The error identification will be set on {2} Error code 1 of the PID management table.

2. When FUN1 detects that the execution flag {50} is reset while PID RUN flag = 1, {58} PID RUN will become "0"and PID operation will be terminated.

{59} PID-in-operation Flag (Read only)

The FUN2 turns on this flag of a loop when performing the PID operation on it and terns off the PID-in-execution flags of other loops.

{60} PID constant OK flag (Read only)

When FUN1 detects that the PID constant change flag {52} is on, the validity of each PID constant ({12} to {16}) will be checked and the result will be set in the PID constant OK flag {60}.

1 : PID constants are valid.

0 : PID constants are invalid. PID operation will be continued with the precedent PID constants. Details of the errors will be set on {2} Error code 1.

{61} Over high limit flag MV > UL (Read only)

When the output value calculated by FUN2 is greater than the Output upper limit UL {17}, this flag will be turned on "1".

{62} Under low limit flag MV< LL (Read only)

When the output value calculated by FUN2 is less than the Output lower limit LL {12}, this flag will be turned on "1".

{63} FUN2 error flag (Read only)

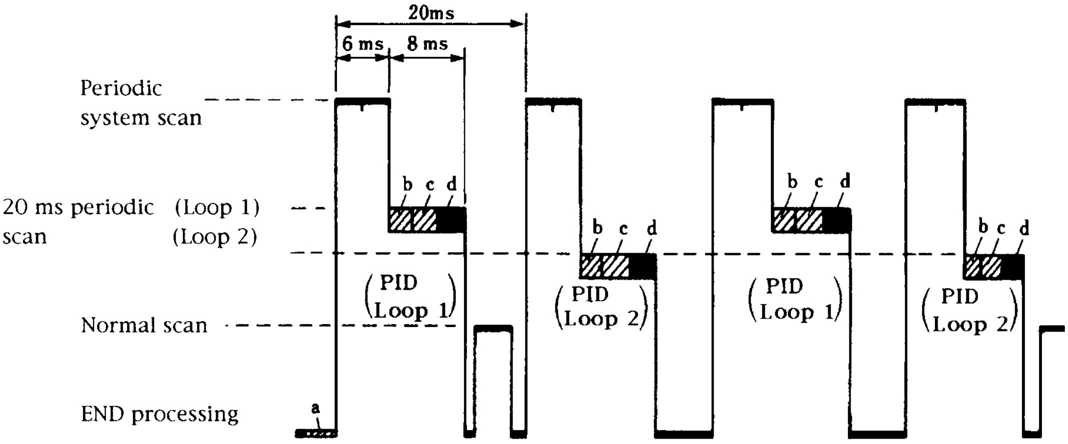
When an error is found in the Output upper limit {17}, the Output lower limit {18} or bit patterns {23} to {25}, {63} FUN2 error will turns on "1". the error identification will be set on {2} Error code 2. PID operation will not be stopped by the errors of this type.

When no error exists, {63} FUN2 error flag = 0. Nothing will be set on {2} Error code 2.

4. Execution principle of PID operation

[Example 1] PID operations on two loops with the setting TZ = 2 (× 20 ms)

Note: For an easier understanding, two interruption with 10 ms each are drawn as one combined cycle.



a: FUN0 processing c: FUN2 processing

b: FUN1 processing d: Other periodic interruption processing

Figure 3.4.1 Management of PID operation (two loops)

[Example 2] PID operations on three loops with the following settings.

Loop 1 : TZ = 3 (× 20 ms)
Loop 2 : TZ = 6 (× 20 ms)
Loop 3 : TZ = 12 (× 20 ms)

Note: For an easier understanding, two interruption with 10 ms each are drawn as one combined cycle.

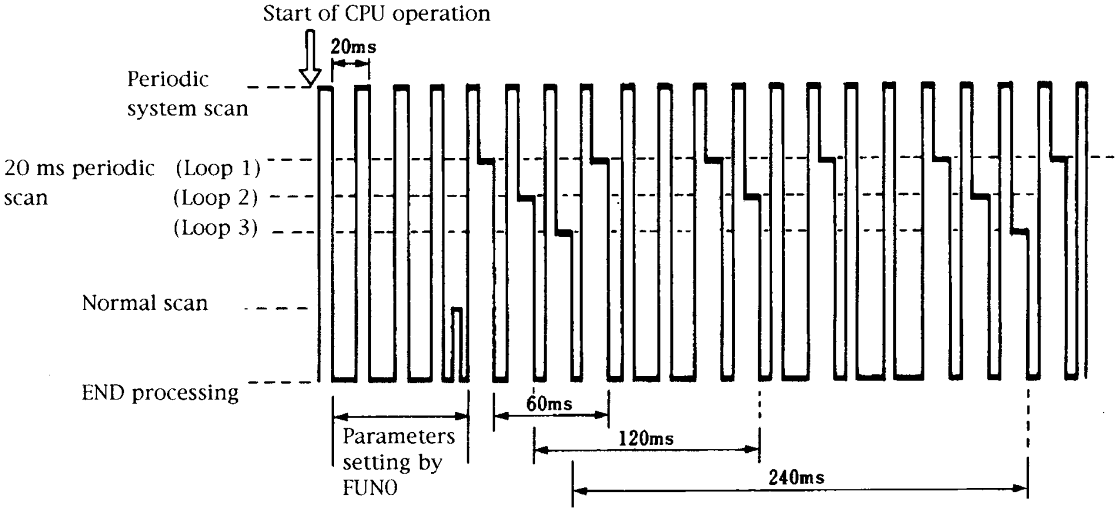
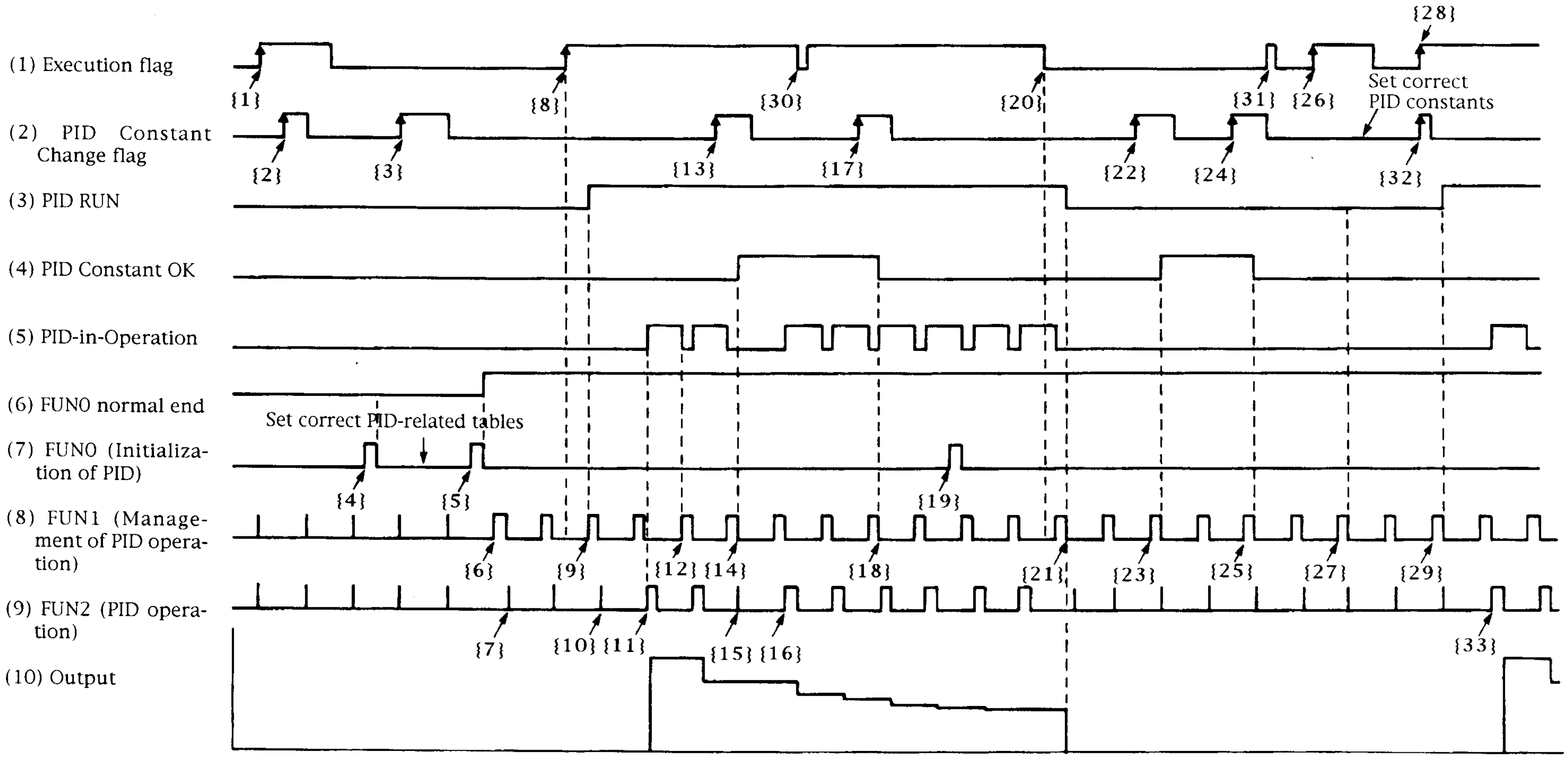


Figure 3.4.2 Management of PID operation (three loops)

5. Time chart in PID operation.

(a) [Example 1]

This example shows timing of the operation of the flags (PID RUN, PID Constant OK, PID-in-Operation) and the processings (FUN0, FUN1, FUN2) at the ON-to-OFF transition of both the execution flag and the PID Constant Change flag for a loop.



[Explanations for the example 1]

{1}, {2}, {3} These OFF-to-ON transitions are ignored as FUN0 is not executed normally.

{4} As the FUN0 processing finds an error in the PID-related table, FUN1 is disabled.

{5} FUN0 processing has ended normally and

{6} FUN1 processing starts.

{7} FUN2 does not perform the PID operation as the Execution flag is OFF.

{9} FUN1 {9} detects the OFF-to-ON transition {8} of Execution flag and checks validities of PID constants. As the PID constants are valid, the PID RUN flag is turned ON.

{10} The PID operation of FUN2 starts at {11} as the first scan is not executed.

{11} The PID-in-Calculation flag is turned ON before the PID operation of the FUN2.

{12} FUN1 turns OFF the PID-in-Calculation flag.

{13} At the OFF-to-ON transition of the PID Constant Change flag,

{14} The FUN1 turns ON the PID Constant OK flag (as the check on the PID constant is valid) and changes the PID constants.

{16} The PID operation starts with the new PID constants, as FUN2 does not perform the PID operation at {15}.

{17} At the OFF-to-ON transition of the PID Constant Change flag,

{18} The FUN1 turns OFF the PID Constant OK flag (as the FUN1 detects an error in the PID constants). The PID Constant flag remains unchanged.

{19} Another FUN0 instruction is ignored when it is executed while the PID operation is executed.

{21} As the ON-to-OFF transition {20} of the Execution flag is detected by FUN1 {21}, the PID RUN flag is turned OFF and the output becomes "0".

{23} As the OFF-to-ON transition {22} of the PID Constant Change flag is detected by FUN1 while the Execution flag is OFF, the PID constants are checked for validities. As they are valid, the PID constants are changed and the PID constant OK flag is turned ON.

{25} As the OFF-to-ON transition {24} of the PID Constant Change flag is detected by FUN1 while the Execution flag is OFF, the PID constants are checked for validities. As they are not valid (error), the PID constant OK flag is terned OFF.

{27} As the OFF-to-ON transition {26} of the Execution flag is detected by FUN1 and PID constants are checked. As they have an error, the PID RUN flag is turned OFF.

{29} As the OFF-to-ON transition {28} of the Execution flag and the OFF-to-ON transition {32} of the PID Constant Change flag are detected simultaneously by FUN1 {29} the OFF-to-ON transition {32} of the PID Constant Change flag is ignored. As the PID constants are checked by the FUN1 and valid, the PID RUN flag is turned ON. The PID operation FUN2 starts from {33}.

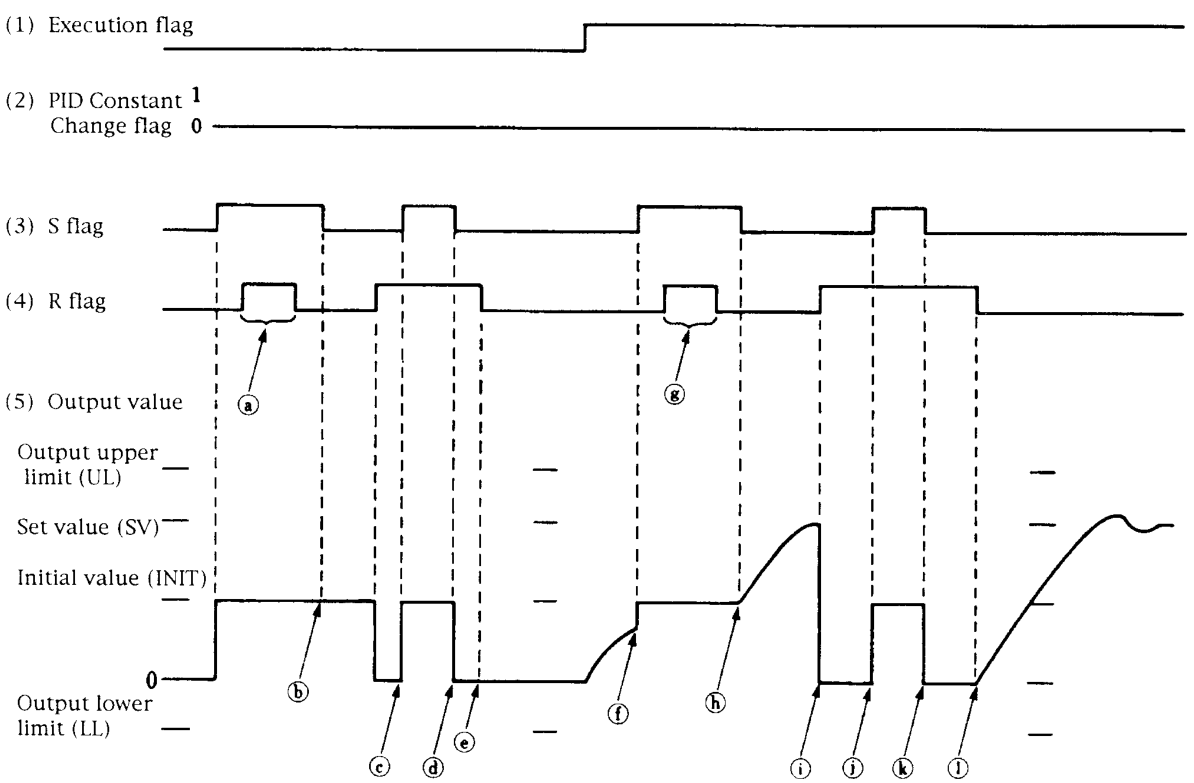
{30}, {31} The ON-to-OFF transition of the Execution flag are ignored if they are not detected in normal periodic scans.

(b) [Example 2]

This example shows timing of the operation of the S flag and the R flag (in the Bumpless mode).

S flag : Resets the output value to the initial value.

R flag : Resets the output value to zero.

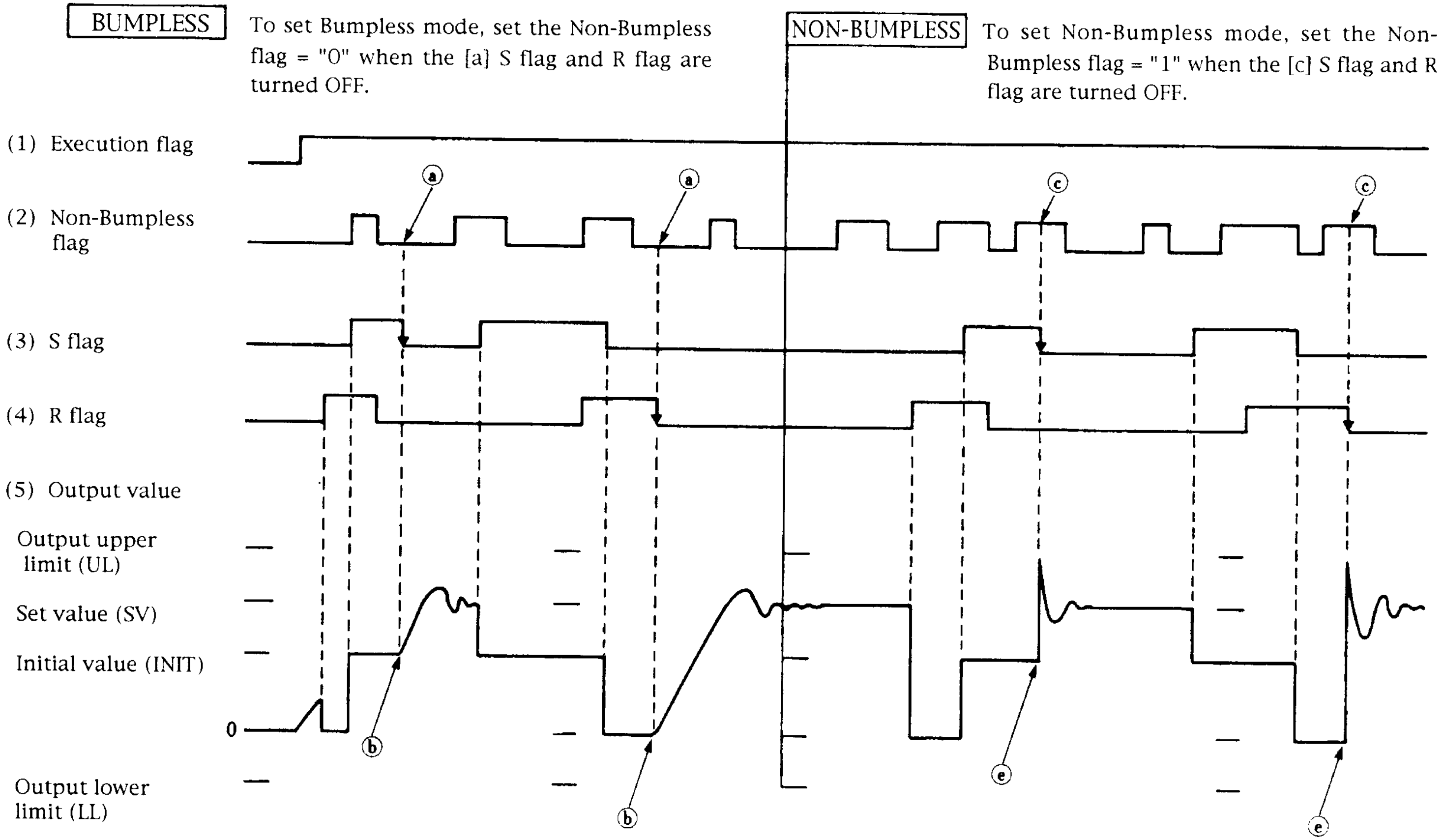


ⓐⓖ	The output value keeps the initial value (INIT) as the S flag has the priority.
ⓑⓔ	The output value is held as the Execution flag is OFF.
ⓒⓙ	The output value becomes the initial value (INIT) as the S flag has the priority.
ⓓⓚ	The output value becomes "0" as the R flag is ON when the S flag becomes OFF.
ⓕ	The output value becomes the initial value (INIT).
ⓗⓛ	The output value continuously approaches the set value as the Execution flag is ON and the operation is in the Bumpless mode.
ⓘ	The output value becomes "0".

├→ The execution of the FUNO has ended normally.

(c) [Example 3]

This example shows timing of the bumpless operation and the non-bumpless operation.



To set Bumpless mode, set the Non-Bumpless flag = "0" when the [a] S flag and R flag are turned OFF.

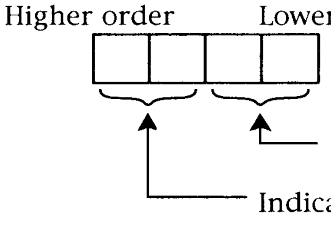
To set Non-Bumpless mode, set the Non-Bumpless flag = "1" when the [c] S flag and R flag are turned OFF.

ⓑ When S flag and R flag are turned OFF, the output value continuously approaches the set value.

ⓔ When S flag and R flag are turned OFF, the output value makes a sudden approach to the set value .

6. Details of the error codes for the PID instructions

The error code is expressed by a hexadecimal number in 4 digits.



Indicates the cause of the error.

Indicates the loop number.
H00: Error not related to loop numbers.
H01~ H40: Error occurred in a loop with its number announced.

In Error code lists, a loop number of H01 to H40 is set in the upper 2 digits of an error code.

Figures in parentheses { } correspond to the figures used in the PID management table shown in 3.4.3.

(a) Error code 0

An error code will be set in this Error code 0 area when the error occurs in the processing of FUN0 or, the processing FUN1 (partly). When no error is detected, the last status will be held.

Table 3.4.2 Details of Error code 0 in PID (1/2)

Error code	Causes	Remedies	Remarks
0001	Although FUN0 has been ended normally, The same instruction is repeated.	Do not try to execute FUN0 again after it is ended normally.	{5} The Normal Termination of FUN0 will hold the last value.
0002	{3}The number of loops is 0.	Specify {3} the number of loops 1 to 64	
0003	{3}The number of loops is greater than 64.	Specify {3} the number of loops 1 to 64.	
0004	The PID management table exceeds the maximum No. of WR.	Change the top address of the PID management table or {3}the number of loops to avoid the number exceeding the maximum No. of WR.	The size of the PID management table is variable. When {3} the number of loops exceeds the I/O limit {5} the Normal termination of FUN0 will hold the last value.

Table 3.4.2 Details of Error code 0 Area in PID (2/2)

Error code	Causes	Remedies	Remarks
xx05	The word table of loop xx exceeds the maximum No. of WR.	Specify {4} the correct No. of WR for the loop of the word table.	Size of a word table is 48 words per a loop.
xx06	The bit table of the loop xx exceeds the maximum No. of R.	Specify {11} the correct No. of R for the loop of the bit table.	Size of a bit table is 16 bits per a loop.
xx07	{17} the output upper limit of the loop xx is outside the permissible range.	{17} the output upper limit must be in -32767 to 32767.	
xx08	{18} the output lower limit of the loop xx is outside the permissible range.	{18} the output lower limit must be in -32767 to 32767.	
xx09	{19} the initial value of the loop xx is outside the permissible range.	{19} the initial value must be in -32767 to 32767.	
xx0A	{17}the output upper limit, {18}the output lower limit and {19}the initial value for the loop xx are not properly related.	Make the relation; {18} the output lower limit ≤{19} the initial value ≤{17} the output upper limit	
xx0B	{23} the bit pattern of the set value for the loop xx is outside the permissible range.	Specify the set value 1 to 4 in {23} the bit pattern of the set value.	
xx0C	{24} the bit pattern of the measuring value for the loop xx is outside the permissible range.	Specify the set value 1 to 4 in {24} the bit pattern of the measuring value.	
xx0D	{25} the bit pattern of the output value for the loop xx is outside the permissible range.	Specify the set value 1 to 4 in {25} the bit pattern of the output value.	
0020 (Note)	FUN1 is executed before FUN0 is normally completed.	Execute FUN1 after a completion of a normal execution of FUN0.	This error code will be set on the area of <u>Error code 0</u> that is specified by the S of FUN1(S).
0021 (Note)	The argument S of FUN1 (S) is not equal to {1} that of FUN0 of the PID management table.	Use the same WR for S in FUN1(S) and FUN0(S).	This error code will be set on the area of <u>Error code 0</u> that is specified by the S of FUN1(S).

(Note) The error codes 0020 and 0021 are written over previous error codes (0001 ~xx0D). Therefore, make sure that FUN0 has been normally ended before FUN1 is executed. Refer to the example program in 3.4.7.

(b) Error code 1

An error code will be set in this Error code 1 area when the error occurs in the processing of FUN1. When no error is detected, the last status will be held.

Table 3.4.3 Details of Error code 1 in PID (1/2)

Error code	Causes	Remedies	Remarks
0020	FUN1 is executed before FUN0 is normally completed.	Execute FUN1 after a completion of a normal execution of FUN0.	This error code will be set on the area of Error code 0 that is specified by the S of FUN1(S).
0021	The argument S of FUN1 (S) is not equal to {1} that of FUN0 of the PID management table .	Use the same WR for S in FUN1(S) and FUN0(S).	This error code will be set on the area of Error code 0 that is specified by the S of FUN1(S).
xx22	{20}The set value I/O No. for the loop xx is invalid.	Set correct set value I/O No. {20} by the ADRIO instruction.	These errors will likely be occurred on a rising of the Execution flag.
xx23	{21}The measuring value I/O No. for the loop xx is invalid.	Set correct {21} measuring value I/O No. by the ADRIO instruction.	
xx24	{22}The output value I/O No. for the loop xx is invalid.	Set correct {22} output value I/O No. by the ADRIO instruction.	
xx25	{12} The sampling timefor the loop xx is outside the permissible range.	Set {12} the sampling time in 1 ~ 200.	These error will likely be occurred on a rising of the Execution flag or the PID Constant Change flag.
xx26	{12} The sampling time for the loop xx is not a multiple of {3} the number of loops.	Set {12} the sampling time to be a multiple of {3} the number of loops.	
xx27	{13} The proportional gain for the loop xx is outside the permissible range.	Set the {13} proportional gain in -1000 ~ 1000.	
xx28	{14} The integral constant for the loop xx is outside the permissible range.	Set the integral constant {14} in 1 ~ 32767.	
xx29	{15} The differential constant for the loop xx is outside the permissible range.	Set {15} the differential constantin 1 ~ 32767.	
xx2A	{16} The differential lag constantfor the loop xx is outside the permissible range.	Set {16} the differential lag constantin 1 ~ 32767.	

Table 3.4.3 Details of Error code 1 in PID (2/2)

Error code	Causes	Remedies	Remarks
xx30	{18}the output lower limit and {17}the output upper limit for the loop xx are not properly related.	Set {18} the output lower limit ≤ {17} the output upper limit.	This error will likely be occurred when {53} S flag is ON during {58}PIDRUN flag is OFF.
xx31	{22} the output value I/O No. for the loop xx is invalid.	Set correct output value I/O No. by the ADRIO instruction.	This error will likely be occurred when {53} S flag or {54} R flag is ON during {58} PIDRUN flag is OFF.
xx32	{25} the bit pattern of the output value for the loop xx is outside the permissible range.	Set {25} the bit pattern of the output in 1 ~ 4.	

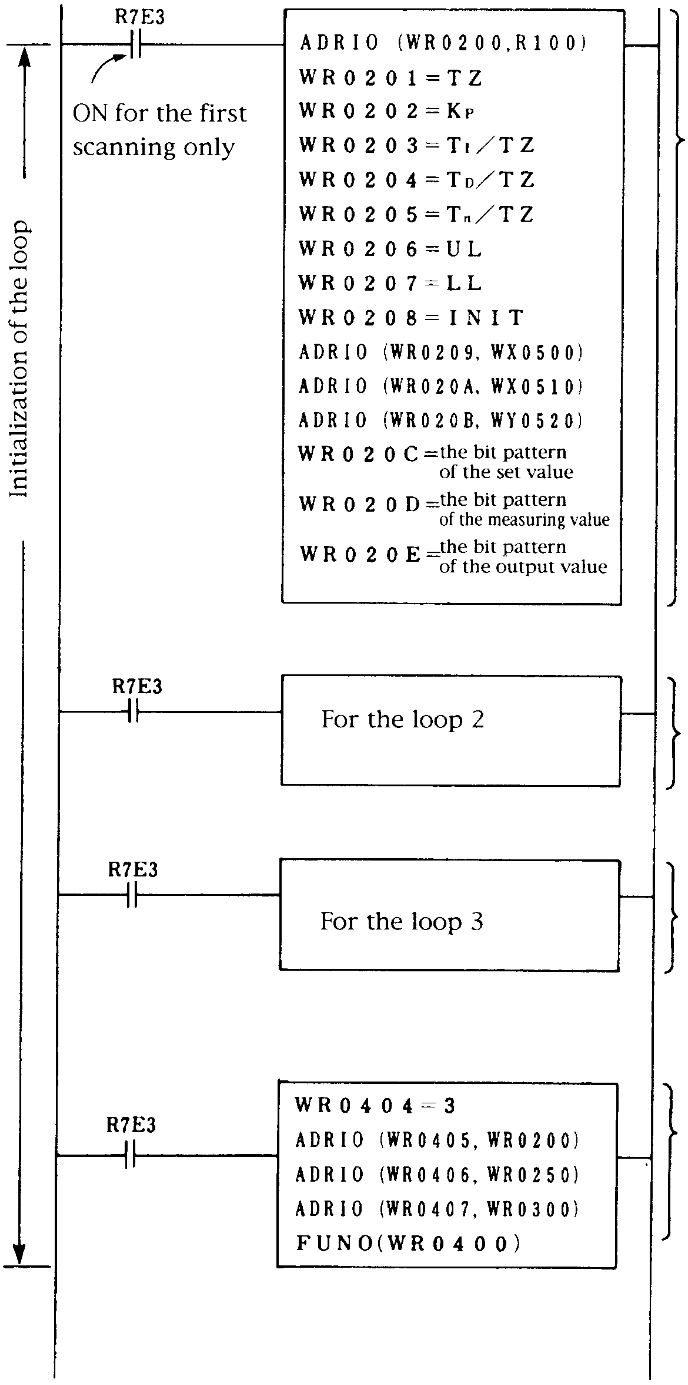
(C) Error code 2

An error code will be set in this Error code 1 area when the error occurs in the processing of FUN 2. When no error is detected, the last status will be held.

Error code	Causes	Remedies	Remarks
xx40			(Reserved)
xx41	{23} the bit pattern of the set value for the loop xx is outside the permissible range.	Set {23} the bit pattern of the output in 1~4.	When the bit pattern is outside the permissible range, execution will be proceeded as "4:no conversion".
xx42	{24} the bit pattern of the measuring value for the loop xx is outside the permissible range.	Set {24} the bit pattern of the measuring value in 1~4.	
xx43	{25} the bit pattern of output value for the loop xx is outside the permissible range.	Set {25} the bit pattern of the output value in 1~4.	
xx44	{18} the output lower limit and {17} the output upper limit for the loop xx are not properly related.	Set {18} the output lower limit ≤ {17} the output upper limit.	When they are not properly related, outputs will not be made though the execution continues.

7. Example program

This example is comprised of three loops. In this example, PID constants will be rewritten every time CPU stars RUN.



```
ADRIO (WR0200,R100)
WR0201=TZ
WR0202=Kp
WR0203=Tɪ/TZ
WR0204=Tᴅ/TZ
WR0205=Tₙ/TZ
WR0206=UL
WR0207=LL
WR0208=INIT
ADRIO (WR0209, WX0500)
ADRIO (WR020A, WX0510)
ADRIO (WR020B, WY0520)
WR020C=the bit pattern
       of the set value
WR020D=the bit pattern
       of the measuring value
WR020E=the bit pattern
       of the output value
```

PID constants are set for the loop 1 (Refer to the word table and the bit table for each loop in Section 3.4.3 (b))

```
For the loop 2
```

PID constants are set for the loop 2

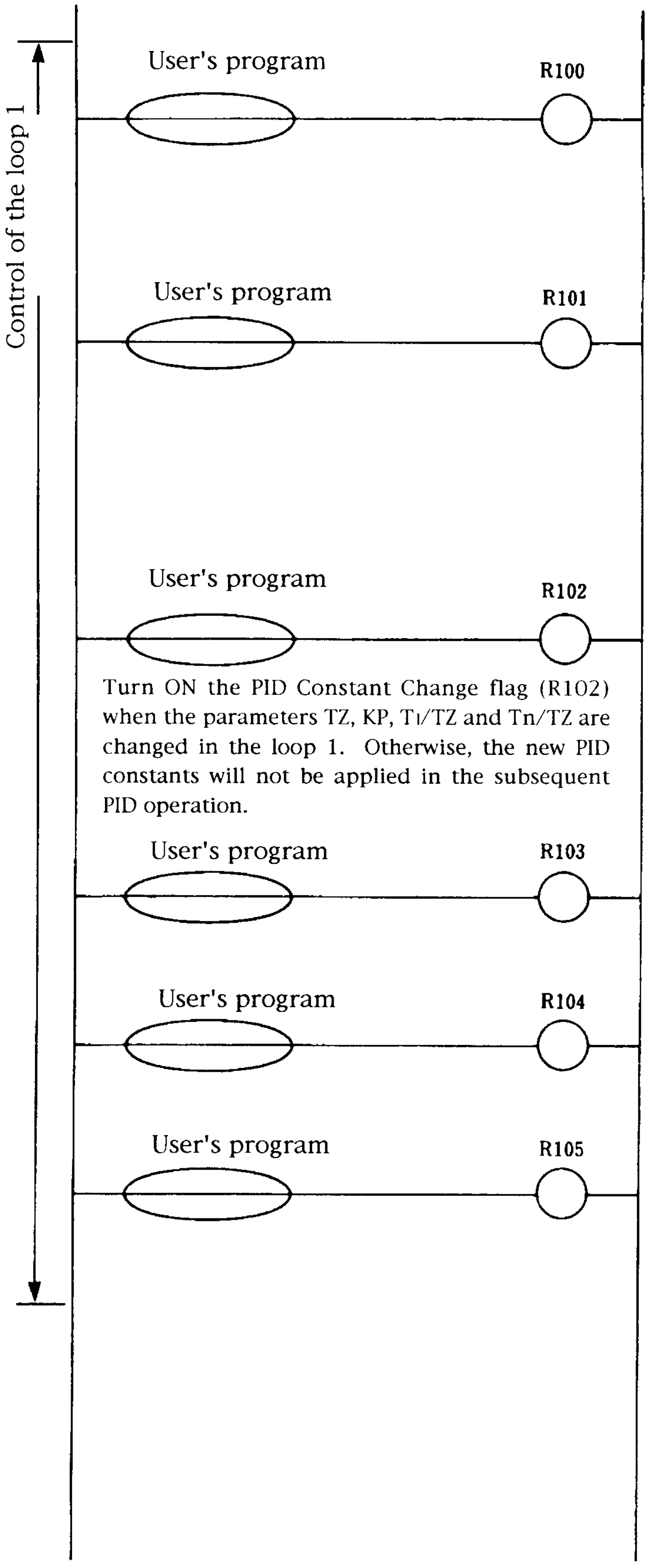
```
For the loop 3
```

PID constants are set for the loop 3

```
WR0404=3
ADRIO (WR0405, WR0200)
ADRIO (WR0406, WR0250)
ADRIO (WR0407, WR0300)
FUNO(WR0400)
```

The management table for the loops are set.
(Refer to Details of PID management table in Section 3.4.3)

- PID constants are checked for their validities in the loops 1, 2, 3.
- Set the PID constants before an execution of FUNO.



Turn ON the PID Constant Change flag (R102) when the parameters TZ, KP, Ti/TZ and Tn/TZ are changed in the loop 1. Otherwise, the new PID constants will not be applied in the subsequent PID operation.

Execution flag
1 : Execution of PID operation in the loop 1.
0 : Termination of PID operation in the loop 1.
The output value will become 0.

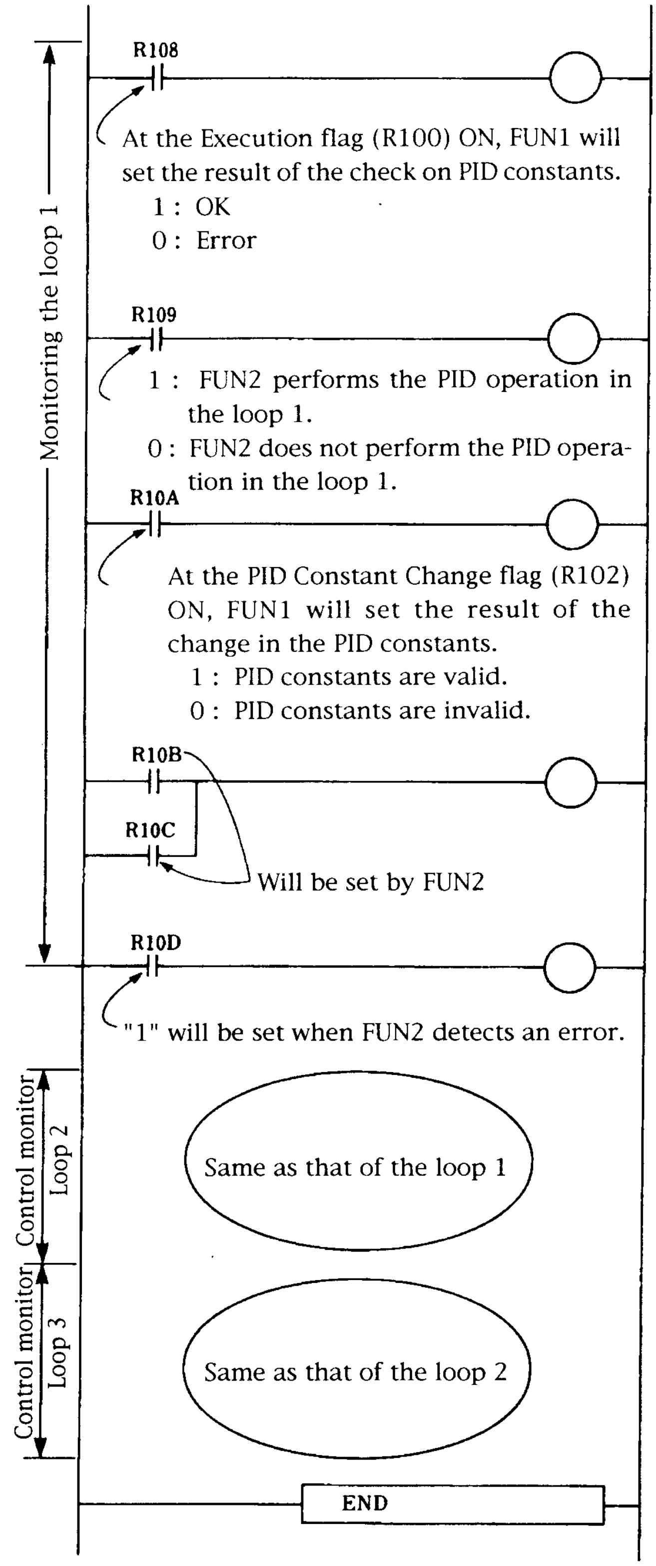
Non-Bumpless flag
1 : Non-Bumpless.
0 : Bumpless

PID Constant Change flag
_↑‾ : Calculating to change the PID constants at the OFF-to ON transition of this flag.

S flag
1 : {19} The initial value is outputted.
S flag has the priority over R flag.

R flag.
1 : "0" is outputted.

D-FREI flag
1 : The PID operation is performed.
0 : The PI operation is performed.



At the Execution flag (R100) ON, FUN1 will set the result of the check on PID constants.
1 : OK
0 : Error

1 : FUN2 performs the PID operation in the loop 1.
0 : FUN2 does not perform the PID operation in the loop 1.

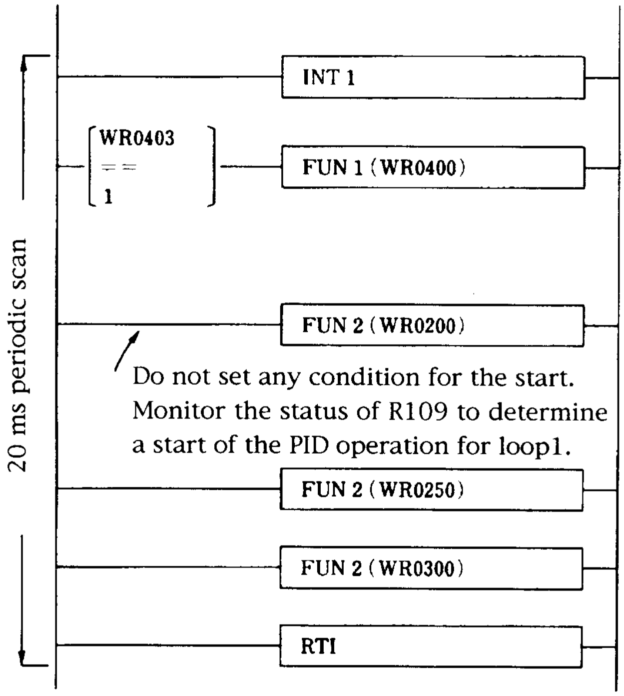
At the PID Constant Change flag (R102) ON, FUN1 will set the result of the change in the PID constants.
1 : PID constants are valid.
0 : PID constants are invalid.

Will be set by FUN2

"1" will be set when FUN2 detects an error.

Will be set when the output value calculated by the PID function is not between the output upper limit and the output lower limit.

When R10D turns ON, an error code will be set in {2} the Error code 2.



Start of the 20 ms periodic scan.

Design the program to execute FUN1 (WR400) after FUN0 has been normally terminated (WR403 = 1). Through this design an error will be identified in the Error code 0 area if FUN0 produces the error.

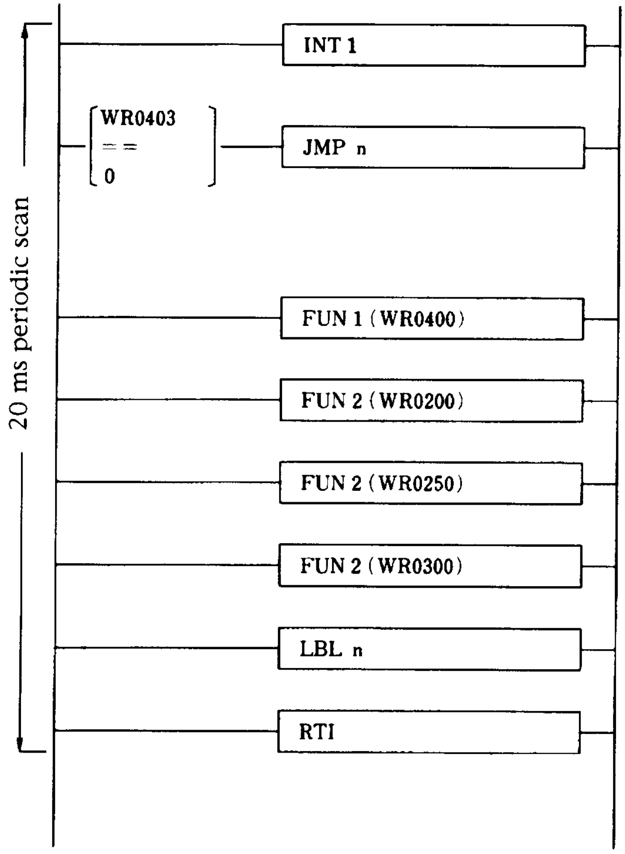
PID operation is made for the loop 1.

Do not set any condition for the start. Monitor the status of R109 to determine a start of the PID operation for loop1.

PID operation is made for the loop 2.

PID operation is made for the loop 3.

The above program can be replaced by the following.



Start of the 20 ms periodic scan.

The program will jump to LBLn if FUN0 does not perform normally. When FUN0 performs normally, the following commands will be executed. Through this design an error will be identified in the Error code 0 area if FUN0 produces the error.

The loop No. will be determined to execute in this 20 ms scan.

PID operation is made for the loop 1.

PID operation is made for the loop 2.

PID operation is made for the loop 3.

The program will jump to this step if FUN0 does not perform normally, skipping the instructions.

Chapter 4 Communication with the host computer

4.1 Outline

The communication protocol for H-250/252CPU is based on the communication procedure for H series (H-300/700/2000). The following description is made in accordance with the procedure in H series.

The communication protocol of H-series specifies the protocol that is effective between the communication module of H-series and a host computer (a personal computer or micro-computer is assumed).

The modules of H-series, applicable to this protocol are, (1)CPU, (2)COMM module, (3)remote I/O module in a local station, (4)GPIB module and(5)CPU link module.

The communication protocol of H-series specifies the particular task codes (command codes) and the procedure for the transmission control.

The outlined structure of the protocol is shown in Fig.4.1.1

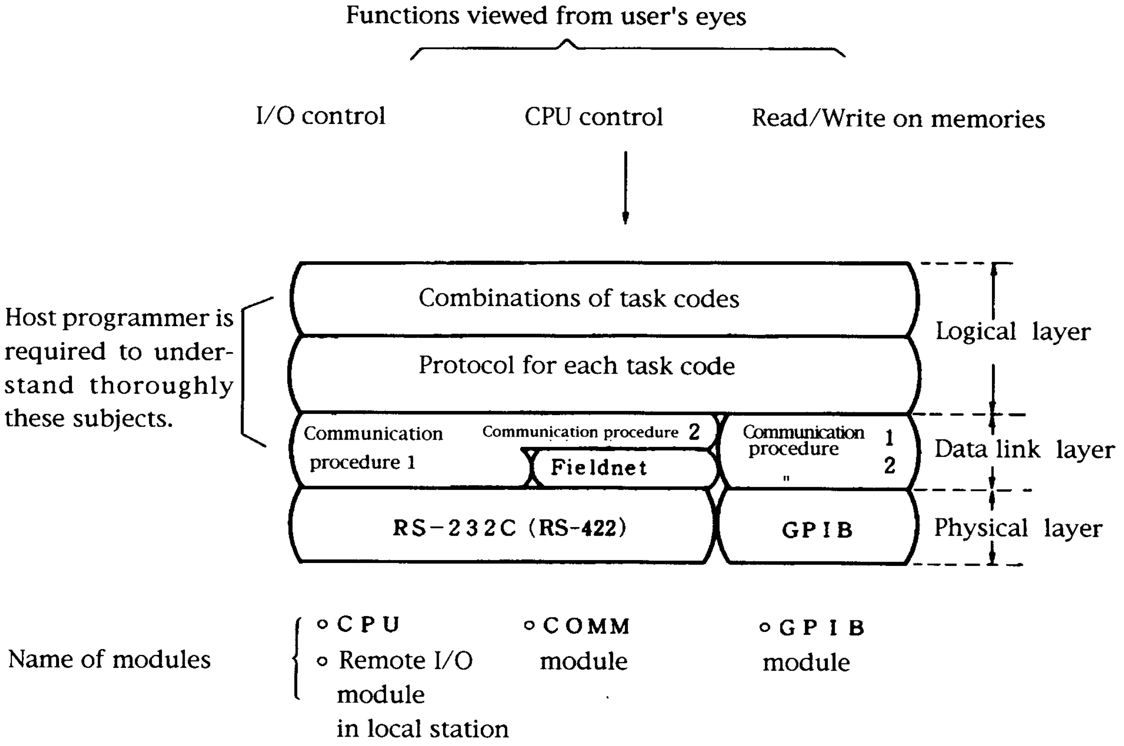


Figure 4.1.1 Outline of the communication protocols for H-Series

CPU, the remote I/O module in a local station and the COMM module utilize RS-232C(RS-422 is also effective for COMM) at the physical level. The simplified procedure is used at the data link level. The simplified procedure is the uni-directional and is common to COMM, CPU and the remote I/O module in a local station. The COMM and GPIB are, in addition, provided with the bi-directional procedure that enables the transmission of start up signals to the host computer. The COMM, in addition to the above, has the procedure corresponding to Fieldnet 600(2-port LAN).

Nevertherless, in H-250/252, no support is provided for the communications by COMM module, remote I/O local module and GPIB module, nor for Field net 600.

As explained above, for H-series, the protocol is defined so that the host computer (controller) can access all CPU in the network of H-series.

The following six functions of H-series are available to users.

(1) CPU control (RUN/STOP control of CPU, Set/Reset of occupancy, Read CPU status etc.)

(2) I/O control (Various monitors)

(3) Write memory (Clear all, transfer all etc.)

(4) Read memory (Read programs etc.)

(5) Response (Various responses from CPU)

To utilize these functions, various task codes are to be assembled in the host computer program.

4.2 Procedure for transmission control

(1) Characters for transmission control

H-series programmable controller (hereinafter abbreviated as PC) uses control characters as shown in Table 4.2.1.

Table 4.2.1 Characters for transmission control

STX (Start of Text)	Top signal of Text	H02
CR (Carriage Return)	End signal of Text	H0D
ACK (Acknowledge)	Signal of normal transmission	H06
NAK (Negative Acknowledge)	Signal of abnormal transmission	H15
ENQ1 (Enquiry 1)	Ordinary enquiry	H05
ENQ2 (Enquiry 2)	Enquiry to a destination	H05
EOT (End of Transmission)	Signal of End of transmission	H04

(2) Message format

Maximum message length: 503 bytes/message (including transmission control character).

STX

STX	Response TM	L	U	M	P	TEXT	SUM	CR

LUMP.....In the transmission system of H-series, address should be assigned on the network, in order to designate required PC from the host side.

L: Loop No. (Loop No. of CPU link)

Item	Loop No.
CPU link Loop 1	H01
CPU link Loop 2	H02
No CPU link No access to a terminal linked	*HFF

U: Unit No. (Station No. of CPU link)

Item	Unit No.
CPU link ST No. 0 │ CPU link ST No. 63	H00 │ H3F
No CPU link No access to a terminal linked	*HFF

* When Loop No. is HFF, Unit No. should also be HFF, and vice versa.

M: Module No."H00" specified.

P: Port No. "H00" specified.

In the case of troubles in the transmission system, the module No. (M) and the port No. (P) that detected the trouble are set at M and P respectively, as shown in Table 4.2.2 and Table 4.2.3.

Table 4.2.2 Module No. (M)

Name of module	Module No.				Remarks
	Nearest slot to CPU				
CPU	H00	-	-	-	H01 ~ H03 Not in use
BASIC	H04	-	-	-	H05 ~ H07 Not in use
CPU LINK	H08	H09	-	-	H0A ~ H0F Not in use
REMOTE (Host)	H10	H11	H12	H13	H14 ~ H17 Not in use
COMM	H18	-	-	-	H19 ~ H1F Not in use
GPIB	H20	-	-	-	H21 ~ H27 Not in use

Table 4.2.3 Port No. (P)

Item		Port No.
CPU	CPU	H00
	Peripheral ports	H01
BASIC	——	H00
REMOTE	REMOTE Local station No.0 | REMOTE Local station No.9	H00 | H09
COMM	*RS - 232C	H00
	RS - 422	H01
GPIB	——	H00

* When RS-232C port of COMM module is connected to FNT600-T65(Product of Hitachi Engineering Co.),H00 ~ H7D will be set.

- ○ Text.............Task code
- ○ Response TM......To be assigned when the signal sent to PC from a host.
- ○ Su...............Sum value.

ACK

ACK	CR

NAK

NAK	Response TM	RTC	CR

ENQ1

ENQ	Response TM	"1"	CR

EOT

EOT	CR

- ○ RTC Return code
- ○ Response TM

Allocated only when the communication is made from a host to a PC at one way start. This is not allocated when the communication is made from a PC to a host.

This value specifies the waiting time for sending response tasks, ACK, EOT, ENQ, NAK from PC. This value, once set from the host, will remain valid till a next receipt of the response TM. Range of set is "H00" ~ "H0F", and its value is 0 ~ 150ms. (Response TM x 10ms)

(3) Negate command and error details

(a) Command to request a re-try of communication

In the case of a failure in receiving a message from a terminal (a host or PC), the negative acknowledge command is used to the terminal.

Sending format

NAK	TM	RTC	CR

(TM: To be assigned when the signal sent to PC from a host.)

RTC H00...............Dummy (To be sent only from a host to PC. Refer to 4.2(8)⑤)
H01 ....Parity error
H02...............Sum error
H03...............Framing error
H04...............Over run error
H05...............Protocol error
H06...............ASCII error
H07...............Communication buffer error
H08...............Communication time over

(b) Abort command

The command to execute a forced abort of communication initiated from a host side.

Communication format: RTC

HFF........Abort

NAK	TM	RTC	CR

(TM: To be assigned when the signal sent to PC from a host.)

Details of errors

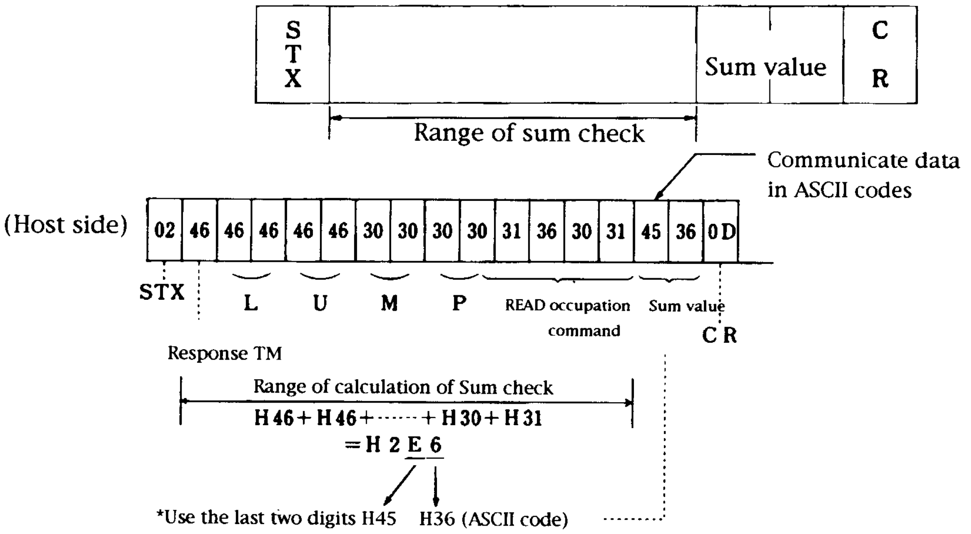
(a) Parity error (RTC is H01)

In each transfer of a character, when the even parity check is specified a check is made to see whether the number of bits for "1" added in the communicated data including parity bits are even or not. Likewise, when the odd parity check is specified a check is made to see whether the number of bits for "1" in the communicated data including parity bits are odd or not. This type of check is called as the vertical parity check. If these conditions are not fulfilled, NAK signal with RTC as 01 will be sent to the host that is dispatching data. In the CPU module, only the even parity check is provided, therefore the odd parity check can not be specified.

(b) Sum error (RTC is H02)

If the sum value in the transferred data from the sender differs from the sum value in the receiver, NAK signal with RTC as H02 will be transferred to the sender.

The definition of the sum value: The value given by two-digit ASCII expression of hexadecimal numbers of lowest digit byte of binary sum value resultant of ASCII data in the scope of the sum check.



The following is the expression of this data by characters.

$\boxed{STX}$ + "F" + "F F F F0000" + "1601" + "E 6" + $\boxed{CR}$
Sum value

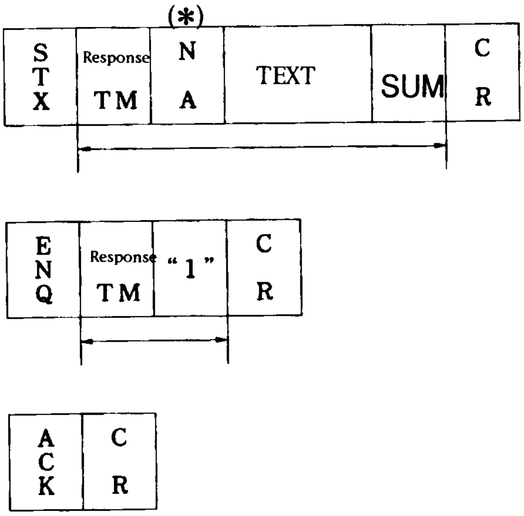
(c) Framing error (RTC is H03)

When an abnormality is occurred in the frame accommodating received data starting from the start bit and ending by the stop bit, NAK signal with RTC as H03 will be transferred to the sender.

(d) Over run error(RTC is H05)

When a failure is occurred in acquiring transferred data, NAK signal with RTC as H04 will be transferred to the sender.

(e) Protocol error (RTC is H05)



(*)NA: Means the net work address which is named as the total of L. U. M. P.

(f) ASCII error (RTC is H06)

NAK signal with RTC as H06 will be transferred to the sender, when a received data is found to be other than 0 to 9, nor A to F in ASCII code in the scope of the data format as shown in the following figures.

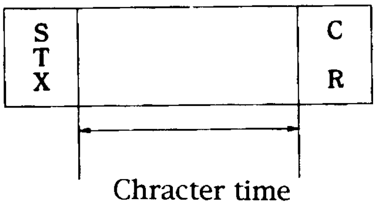
(g) Receiving buffer error(RTC is H07)

NAK signal with RTC as H07 will be transfered to the sender, when the receiving buffer(503 bytes) is overflowed.

(h) Receiving time over(RTC is H08)

STX
ENQ
ACK
NAK
EOT

When a time over is occurred after receiving the start character of (STX, ENQ, ACK, NAK, EOT) and a timer is started and before receiving a CR chracter, the error signal will be transfered.



Duration of check time is 16 seconds in 2400 BPS, 8 seconds in 4800 BPS, 4 seconds in 9600 BPS and 2 seconds in 19200 BPS.

(i) Abort(RTC is HFF)

To be used when the host forces to abort communication.

(4) Definition of incorrect commands

(1) When a command was received in a form different from the correct expression as specified.

The length of the required task code is less than 14 characters.

The command length of ENQ is less than 4 characters.

The command length of NAK is less than 5 characters.

Number of characters from STX to CR is even number.

(2) When a command is received in a form other than correct task codes, ENQ, NAK.

(5) Definition of protocol error

When the communication system of H series received data that is not supposed to, or when received an abnormal commands, the system defines these status as protocol error. Examples of protocol errors are shown below.

(a) PC side

When PC is ready after a reset or power on, PC received from a host a signal that does not express any of requested task codes nor NAK.

When PC is ready in normal condition, PC received from a host a signal that does not express any of requested task codes nor NAK.

When PC is ready to receive a response code after PC sent ACK to a host, PC received from the host a command other than ENQ and NAK.

(b) Host side (in general)

When a host is ready after a reset or power on, the host received from PC a command other than NAK, even though PC did not receive any data from the host.

When a host is ready to receive ACK from PC after the host sent a requested task code, the host received from PC a command other than NAK.

When a host is waiting for dispatching ENQ, the host received EOT from PC, even though the host did not send any enquiry command to PC.

When a host is ready to receive a response task code, the host received a signal that does not express any of EOT, the response code nor NAK.

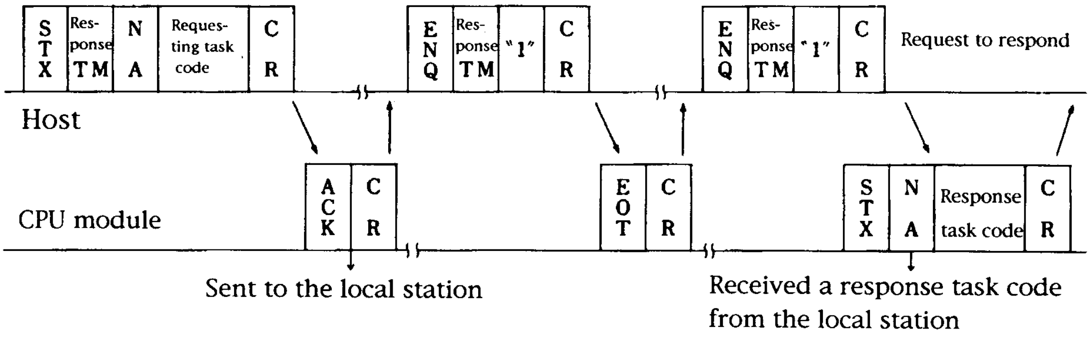
(6) Enquiry command from a host (ENQ1)

ENQ1.....Send this command when requesting PC to send a response task code after the host sent a request task code to PC.

(7) Basic control procedure

(For (7) ~ (11), refer also to Table 4.2.4 in (12) PC communication matrix)

① When a host sends a requesting task code to NA (address of a local station).



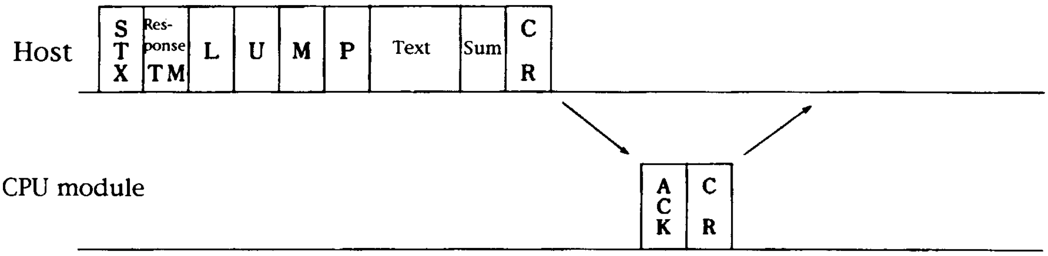
NA : Net work address of the local station (L. U. M,.P).

Response TM : Waiting time for sending the response from PC .

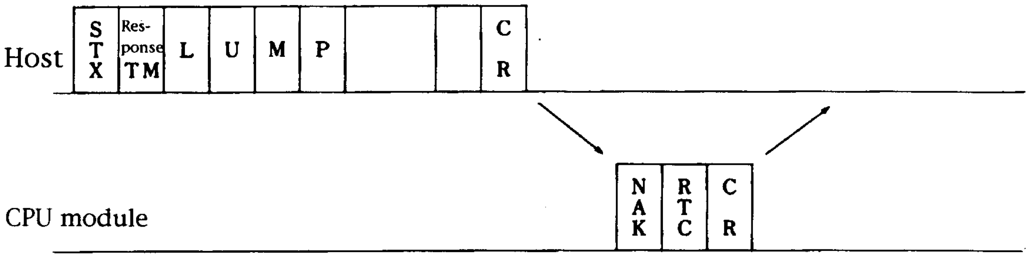
ENQ "1" : Request to send, between the host and PC.

(8) Details of the control procedure

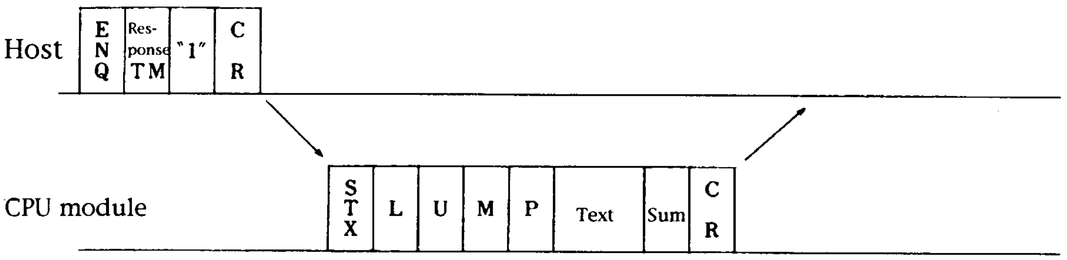
① When PC is received a normal message from the host.



② Negative acknowledgement in the case of a failure in receiving a message from the host.

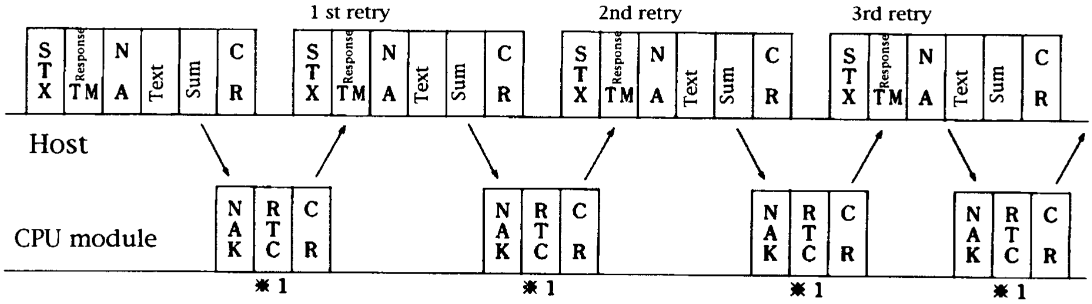


③ Response to enquire from the host when a response text is prepared in PC

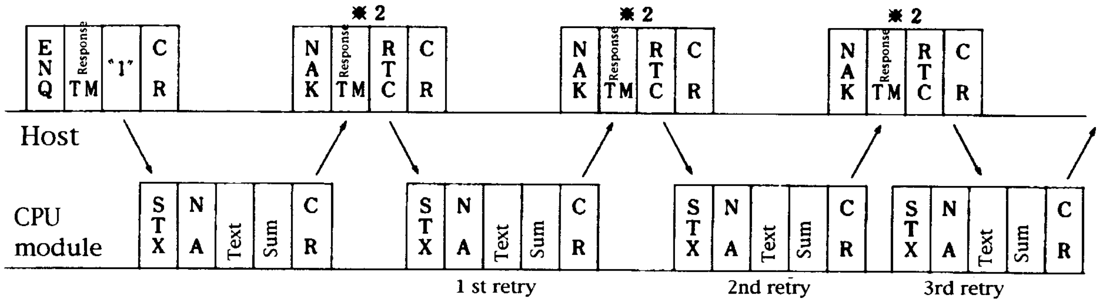


L, U, M, P : Address of the local stAtion

④ Negative acknowledgement of a massage from the host



⑤ Negative acknowledgement from the host



✱ 1 Negative acknowledgement for an error in receiving or an abnormal command.

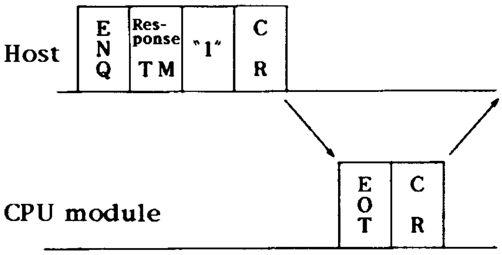
Error in receiving : Framing error
Parity error
Over run error
ASCII error
Receiving buffer error
Sum error
Time out error in receiving
Abnormal command: Protocol error

N A : Netwok address (L, U, M, P)

✱ 2 Negative acknowledgement by error in receiving.
RTC = H00 (Dummy)

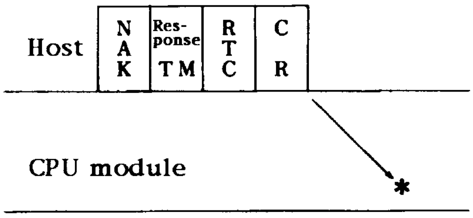
(9) Details of the communication matrix from the host to PC

① When ENQ 1 is received



When the response text is not prepared in PC, EOT will be returned.

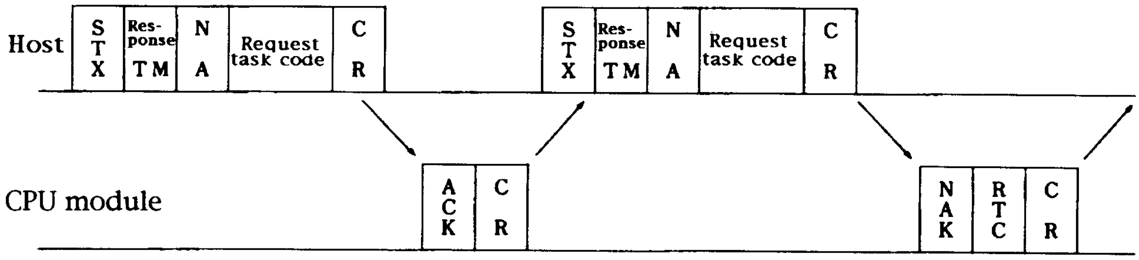
② When NAK (Abort) is received from the host



*PC sends nothing to the host, returning to a ready (initial) status.
The host, after sent an abort, should wait 50ms or more till the next text sending.

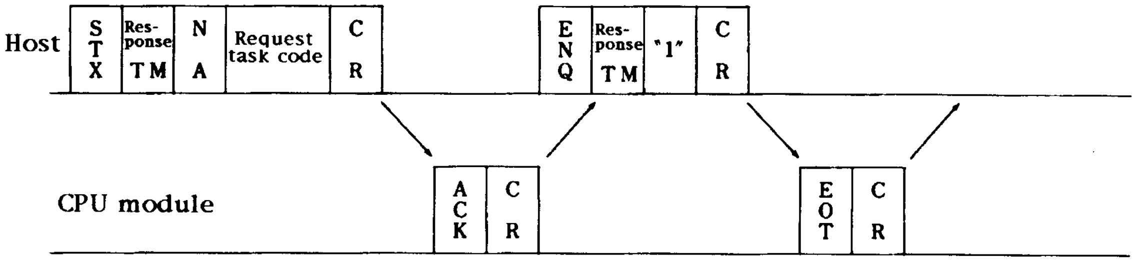
(10) Response to a command from the host, during the waiting time in PC to generate the response task code.

① A requesting task code is received from the host

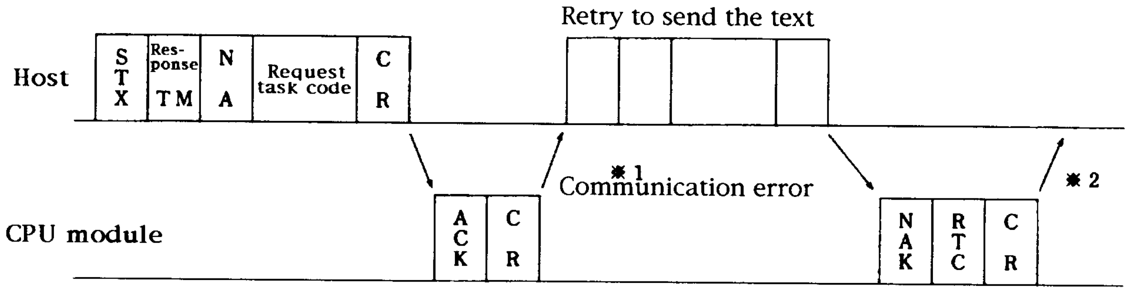


H05 (Protocol error) as RTC

② ENQ1 is received from the host



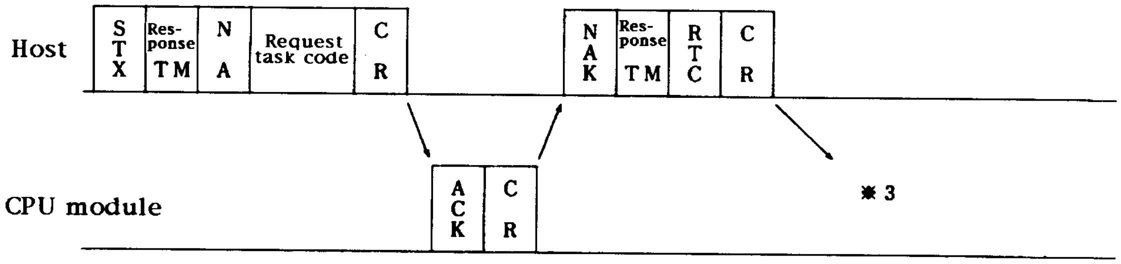
③ Failure in the host in receiving a response from the PC



※1 When the host failed to receive ACK from PC, the host assumes a receipt of NAK and will re-try to send the text.

※2 PC, after sent ACK, when received the text will send NAK (RTS is H05 : Protocol error) to the host and will return to a ready (initial) status.

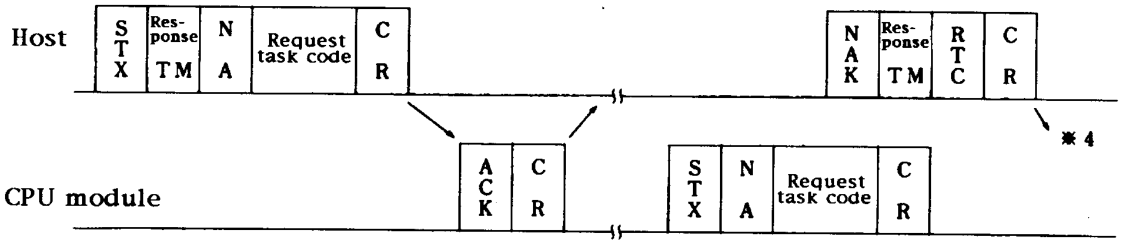
④ NAK (Abort) is received from the host.



※3 When NAK (Abort) is received from the host, PC will return to a ready (initial) status.

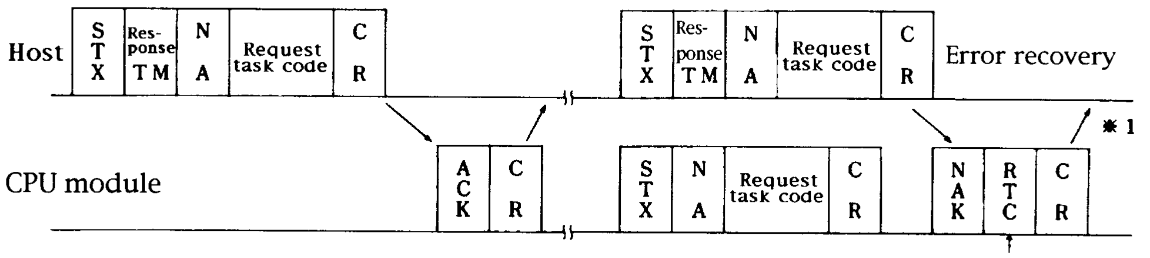
(11) When PC, with the response task code sending generated, is waiting for ENQ 1 from the host, PC received a signal other than ENQ 1.

① NAK (Abort) is received from the host.



※4 Recovering operations.

② Requesting task code is received from the host



*1 Returned to a ready (Initial status)

RTC is H05 (Protocol error)

(12) The matrix of PC communication

This matrix shows operations of CPU module in response to various commands from the host. To prepare the host's programs refer to Table 4.2.4.

Table 4.2.4 Communication matrix effective between the host and PC

Item	Status symbol	A	B		C		D	
		Receive the requesting text	Receive ENQ 1		Receive ENQ 2		Receive NAK (other than abort)	
1	READY (Initial)	Send ACK →2	Send NAK (note 3) →1		Send NAK (note 3) →1		→1	
2	Waiting for ENQ1	Send NAK (note 3) →1	Send EOT (note 1) →2	Send response text (note2) →3	Send EOT (note 1) 2	Send response text (note2) →3	→1	
3	READY	Send ACK →2	Send NAK (note 3) →1		Send NAK (note 3) →1		Send response text till 2nd retry →3	Send respons text, 3rd retry →1

Item	Status symbol	E	F		G	
		Receive abnormal command	Receiving error (note 4)		Receive NAK (abort)	
1	READY (Initial)	Send NAK (note 3) →1	Send NAK →1		→1	
2	Waiting for ENQ1	Send NAK (note 3) →1	Send NAK, till 2nd retry →2	Send NAK, 3rd retry →1	→1	
3	READY	Send NAK (note 3) →1	Send NAK, till 2nd retry →3	Send NAK, 3rd retry →1	→1	

(note 1) In case of no response text.
(note 2) In case the response text is available.
(note 3) Protocol error.
(note 4) Parity error, Sum error, Framing error, Over run error, ASCII error, Receiving buffer error, Receiving time out error.
(note 5) In case of Sum error, Protocol error is assumed and returned to 1.
(note 6) Remains 3 when Sum error or ASCII error occurs at F - 3.

4.3 "Occupation" of CPU and its setting and resetting

In H series, each of hosts (or peripherals) can access at the same time to a single CPU belonging to the communication system.

In this situation, when some hosts actually accessed to the single CPU, troubles will be occurred as shown in the following example.

For example, while a host is writing a program on the CPU, another host is trying to write another program, thereby the hosts are creating an access competition.

In order to avoid such trouble, the provision is made at CPU side to prevent such undue competition, by configurating the status of the subject CPU.

For the above purpose, when a host is occupating to the CPU the status of CPU is configurated to exclude an access from other hosts.

Status of access to CPU are:

(1) READ occupation

(2) WRITE occupation

(3) NO occupation

These three status are explained below.

(1) Occupation of memory

Occupation of CPU is required so that CPU is able to respond properly to each of all peripherals in accordance with their access request (read/write user's program, data memory monitoring). If the occupation is not made, alarm response monitoring nor read/write of a program will be disable.

READ occupation: This type of occupation permits simultaneous co-occupation of CPU by maximum of four peripherals, in which CPU permits reading user's program, read/write for monitoring data memory, on the condition that WRITE occupation is not made by any of the peripherals. During READ occupation, user's program can not be written.

WRITE occupation: This type of occupation permits writing user's program, on the condition that READ occupation is not made by any of the peripherals. Unlike READ occupation, simultaneous occupation by peripherals is not possible. After necessary processing in CPU is finished accompanied by dispatch of READ and WRITE occupation, a resetting task code should be dispatched to reset the occupation. Failure to reset this occupation will cause a possible subsequent task code to be disable because CPU is deemed occupied. During WRITE occupation, a STOP to RUN operation of CPU is not possible by the key switch.

NO occupation: This status means CPU is not in the READ occupation nor WRITE occupation.

(2) Indication of status of occupation

To know CPU is under occupation or not is possible by monitoring special internal outputs.

R7E0=ON: CPU is occupied by master station (local station)

WRF040 : Addresses and modes of occupation are set for peripherals that are occupying
~ CPU, as shown in detailed charts below.
WRF04B

WRF040~42	Occupation table	A
WRF043~45	〃	B
WRF046~48	〃	C
WRF049~4B	〃	D

① Occupation mode	Always H00	WRF040
②LOOP	③UNIT	WRF041
④MODULE	⑤PORT	WRF042

① Occupation mode = H00 : Not occupied
H01 : READ occupation
H02 : WRITE occupation

② LOOP : LOOP No. of the peripherals occupying CPU.
③ UNIT : UNIT No. "
④ MODULE : MODULE No. "
⑤ PORT : PORT No. "

○ An example of WRITE occupation.

WRF040	WRITE	A
WRF043	Not occupied	B
	Not occupied	C
	Not occupied	D

WRITE occupation is registered only on the area A. the areas B ~ D are always not occupied.

○ An example of READ occupation

WRF040	READ1	A
WRF043	READ2	B
WRF046	READ3	C
WRF049	Not occupied	D

Occupied by three peripherals

→

READ1	A
READ2	B
READ3	C
Not occupied	D

A peripheral quits registration of the area B

→

READ1	A
READ2	B
Not occupied	C
Not occupied	D

The area C is transferred to B, resulting in the area C unoccupied

As shown above, allocation of registrations for READ occupations are filled in series from the top. Therefore area C (D) would never be registered, with B being in the status of NO occupation.

4.4 Specifications of Task Code

H-series PC communicates with the host by dispatching task codes. These task codes are detailed in this section.

(1) Detailed function of the task codes

Detailed functions of each task code and each response command are explained the following.
Each task code is explained in accordance with the following general format.

Task code Response task code	Task code No.	Description	Classification	CPU control Read memory Write memory I/O control
Function				
conditions of operation	The conditions are shown for the execution of each code. Expressions of the conditions are explained in the next page.			
Format	The format of the requesting task code and The format of the response task code.			
Explanations				
Example				

When H-250/252 and a host are connected for performing the communication with an upper rank machine (H-300/702/2002 or H-300/700/2000), refer to Chapter 6 of the Software Edition manual for the upper rank machine.

When an upper rank machine and a host are connected for performing the communication with H-250/252,refer to this chapter.

How to read the tables of the conditions of execution.

The tables show the relationship between the status of CPU that enables the task code and the status of occupation of the memory. For details of the status of CPU, refer to the descriptions for the task code "H10 Read status of CPU".

[Example 1] for the condition of execution..

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
×	×	×	×	×	READ occupation	Status of occupation
○	×	×	×	○	Write occupation	

○ Means "Enable"
× Means "Disable"

In the example 1, the task code can be executed only when the CPU is in STOP or in ERROR and at the same time the CPU is occupied by WRITE.

[Example 2] for the condition of execution.

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	

In the example 2, The task code can be executed when the CPU is occupied.

Response task code	H 0 0	Normal execution	Classification	Response

Function

To show the task code requested was normally executed.

Format

H 0 0	(a)	(b)

(a) The task code actually executed.

(b) Data of the execution result.
For details, refer to each task code descriptions.

Response task code	H 0 1	Abnormai task code	Classification	Response

Function

To show that abnormality exists in the requested task code or the execution. (The task not defined, incorrect parameters, etc.)

Format

H 0 1	(a)	(b)

(a) The task code requested to execute.
(b) The return code.

Explanations

Details of the return codes are shown below.

Return code	Description
H01	The task code is not defined.
H02	The code to select a function is not defined.
H04	The address is abnormal.
H05	Number of steps/number of word is abnormal.
H06	I/O code is abnormal.
H07	I/O No. is abnormal.
H09	Tried to write out of the memory size.
H0A	Memory size overflowed

Response task code	H 0 2	Alarm	Classification	Response

Function

To show the master station is in monitor but not occupying CPU.

Format

H 0 2	(a)	(b)

(a) The task code requested to execute.
(b) Data of the execution result.
For details, refer to each task code descriptions.

Response task code	H 0 3	Nonexecutable	Classification	Response

Function

To show the requested task code is not executable.

Format

H 0 3	(a)	(b)

(a) The task code requested to execute.
(b) The return code.

Explanations

Details of the return codes are shown below.

Return code	Description
H01	ROM memory
H02	Mismatching with the parameter area.
H03	Mismatching occupation code (being occupied as READ) .
H04	Mismatching occupation code (being occupied as WRITE) .
H05	Under debugging other stations.
H06	Four stations already occupied as READ.
H07	Own station occupies no CPU.
H08	Other station is occupying CPU.
H0A	RAM memory error
H0B	CPU is in RUN.
H0C	Operational error.
H0D	No program.
H0E	Combination error of task codes.
H0F	Discrepancies in the program.

The descriptions differ depending upon the task code argued. For details, refer to the table showing the responses to each task code in Section 4.5.

Response task code	H 0 5	B U S Y	Classification	Response

Function

To show that the requested task code is not executable because the other task code is already in the course of execution.
(Note) Assemble the program so that the retry of the communication can be made from the task code while the BUSY response is being made.

Format

H 0 5	(a)

(a) The task code requested to execute.

Response task code	H 0 8	Network Failure	Classification	Response

Function

To show that a failure occurred in the communication.

Format

H 0 8	(a)	(b)

(a) The task code requested to execute.
(b) The network address that the failure was detected at.

Explanations

To show that a failure was occurred in the communication or that the address directed does not exist.

Task code	H 10	Read Status of CPU	Classification	CPU control

Function

To read the status of CPU, the status of memories and the version of the software. This task code is executable when the CPU is not occupied

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	
○	○	○	○	○	No occupation	

Format

Request

H1 0	(a)

(a) Selection of the functions (Sub-command)

① H 0 0 : Read the status of CPU.

② H 0 1 : Read the status of memories

③ H 0 2 : Read the version identification of the software

④ H 0 3 : Read error codes

⑤ H 0 4 : Read the name of CPU

Response

① Read the status of the CPU (Subcommand H00).

(a)	H 1 0	(b)	(c)

(a) Response task code (when normally executed H00).
For the task codes other than the normal response task code, refer to the table showing the responses to each task code in Section 4.5.

(b) The status that is read out of CPU.

(c) User's program version (H00 ~ HFF)
This data will be counted only when it is written at the moment an occupation is removed (an accumulative numbers of WRITE occupation will be given). H00 is given on an energizing of the power source.

② Read memory status (Subcommand H01)

(a)	H 1 0	(b)	(c)	(d)

(a) Response task code (H00 in normal execution)
(b) Type of memory
(c) User's memory capacity (No. of steps)
(d) Data memory capacity (No. of words)

③ Read the system software version (Subcommand H02)

(a)	H 1 0	(b)

(a) Response task code (H00 in normal execution)
(b) Version (BCD data in 4 digits)
- Version of the system software (ROM)

④ Read error code (Subcommand H03)

(a)	H 1 0	(b)

(a) Response task code (H00 in normal execution)
(b) CPU error code (Hexadecimal in 2 digits)
- Same as the content of the special internal output WRF000.

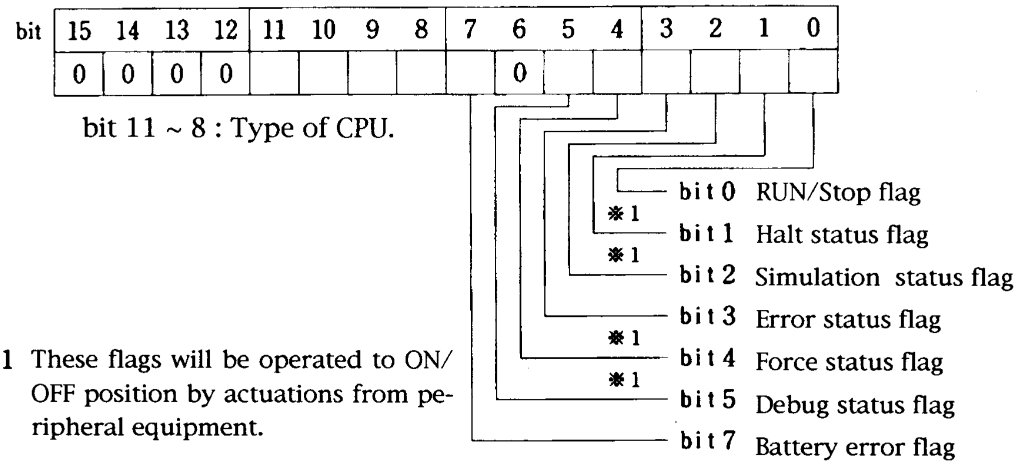
⑤ Read the name of CPU (Subcommand H04)

(a)	H 1 0	(b)

(a) Response task code (H00 in normal execution)
(b) Name of CPU (ASCII code in 16 characters)
CPU22-02HB H-252
CPU21-02H H-250
The portion that is short of 16 characters is added by spaces (H20) to make 16 characters in total.

Explanations

① The status (4 digits) b of CPU is detailed below (the response to subcommand"H00").



bit 11 ~ 8 : Type of CPU.

※1 These flags will be operated to ON/OFF position by actuations from peripheral equipment.

bit 11 ~ 8 Table of CPU type

b i t				Type of CPU	Remarks
11	10	9	8		
0	0	0	1	CPU2−20H CPU−20Ha CPUP−20H	
0	0	1	0	CPU2-07H HH-700 CPUP-07H CPU-07Ha H-252	
0	0	1	1	CPU2-03H CPU-03Ha CPUP-03H H-250 H-200	
0	1	0	0	H−100M	

bit 0 RUN/Stop flag
Indicates CPU is in RUN or in stop.
"1" : RUN/ "0" : Stop.

bit 1 Halt status flag
Indicates CPU is in halt or not.
"1" : Halt/ "0" : Not in Halt.

bit 2 Simulation status flag
Indicates CPU is in the simulation or not.
"1" : in Simulation/ "0" : Not in Simulation.
* As H-250/252 is not able to execute in a simulation mode, the flag will be "0".

bit 3 Error status flag
Indicates CPU is in error or not.
"1" : in Error/ "0" : Normal
When this flag = 1, the identification of error will be given by reading the error code (refer to ④) out.

bit 4 Force status flag

Indicates CPU specified Force or not.

"1" : Force / "0" : Not Force

* As H-250/252 is not able to execute in a Force mode, the flag will be "0".

bit 5 Debug status flag

Indicates CPU is debugging or not.

"1" : Debugging / "0" : Not in debug operation

bit 7 Battery error flag

Indicates the backup battery of CPU is normal or not

"1" : Battery not mounted or low voltage / "0" : Battery is normal

② Status of memories (response to the subcommand H 01)

Item	Description	Explanation	Remarks
Type of memory	H01	EPROM memory pack	MPH- **R
	H02	EEPROM memory pack or Memory pack not installed	MPH* - **E (Note)
Capacity of user's memory	H0010	4kS	MPH-4E, MPH2-4E
	H0020	8kS	MPH-8E, MPH-8R
	H0040	16kS	MPH-16E, MPH-16R
Capacity of data memory	H0040	2kW	MPH-4E, MPH2-4E MPH-8E, MPH-8R
	H0080	18kW	MPH-16E, MPH-16R

(Note) When the software protect using MPH2-4E, MRH-8E, MPH-16E (See Chapter 4 Memory pack, Hardware edition) is performed, "H01" will be made.

③ Version identification of the system software (response to subcommand H02). Version identification of the software of the system ROM installation H series CPU.

④ Error code (response to sub-command H03).
It is possible to read the error code that is same as that of the special internal output WRF000 (self diagnostic error code).

⑤ Name of CPU (response to sub-command H04).

CPU22-02HB - - H-252
CPU21-02H-----H-250

Example

Selection of a function (sub-command) : H00

Request

1	0	0	0

Response

0	0	1	0	0	2	0	1	0	1

Status of CPU, H-252 in operation.

Version name of the user's program = 01H

Selection of a function (sub-command) : H01

Request

1	0	0	1

Response

0	0	1	0	0	2	0	0	2	0	0	0	4	0

EEPROM memory pack or Memory pack not installed

User's memory, capacity 8ks

Data memory, capacity 2kw

Task code	H 11	RUN/Stop of CPU	Classification	CPU control

Function

To control RUN/Stop of CPU from the host.
The "RUN" task requested during an operation or "stop" task requested during a stop will not cause an error.

conditions of execution

The following conditions should be satisfied.

(1) The change over switch is in STOP position, and DIPSW No.4 of CPU is ON.
(2) The status of CPU is not DEBUG nor ERROR.
(3) In the case of a selection of functions to a request to stop, the internal special output R7C4 is ON.
(4) In the case of a selection of functions to a request to run, the internal special output R7C3 is ON and R7E9 is OFF.
When the parameters are set for the inputs for the control operation, their contacts are ON.

(Note) When another station occupies the CPU by WRITE the response will be " Not executable". (When own station occupies by WRITE the request task is executable.)

Format

Request

H 1 1	(a)

(a) Selection of a function (Sub-command)
① H 0 0 : Request to stop
② H 0 1 : Request to run

Response

(a)	H 1 1

(a) Response task code
(H00, when normally executed)
For the task codes other than the normal response task code, refer to the table showing the responses to each task code in Section 4.5.

Explanations

(1) STOP (Sub-command H00)
CPU operation will be stopped.
When STOP is requested during CPU is stopped, a normal response task code will be sent.
When STOP is requested during CPU is in ERROR, the error status will be reset. When the error code is H10 ~ H2F, the error status can not be reset.

(2) RUN (Sub-command H01)
CPU will be put into operation.
When RUN is requested during CPU is running, a normal response task code will be sent.

Example

	Request to STOP	Request to RUN
Request	1 1 0 0	1 1 0 1
Response	0 0 1 1	0 0 1 1

Task code	H 16	Set/Reset of Occupation of CPU	Classification	CPU control

Function

To declare to access to the user's memory. Unless CPU is occupied by this task code the host can not access to the user's memory. Some selection will cause the processing that is the same as that of a finish of alterations on parameters (Task code H27).

conditions of execution

		Status of CPU						
		STOP	RUN	HALT	DEBUG	ERROR		
Selection of a function Subcommand	H 0 1	○	○	○	○	○	READ Occupation	Status of Occupation
		×	×	×	×	×	WRITE Occupation	
	H 0 2	×	×	×	×	×	READ Occupation	
		○	○	○	× * 1	○	WRITE Occupation	
	H 0 5	○	○	○	○	○	READ Occupation	
		○	○	○	○	○	WRITE Occupation	
	H 0 6	○ * 2	○ * 2	○ * 2	× * 3	○ * 2	READ Occupation	
		○	○	○	○	○	WRITE Occupation	
	H 0 0	○	○	○	○	○	READ Occupation	
		○	○	○	○	○	WRITE Occupation	

* 1 : Not executable during a debugging operation by another station.

* 2 : Not executable during a READ occupation made also by another station.

* 3 : Not executable during a debugging operation by another station or during a READ occupation made also by another station.
For details of the execution condition in the occupation status, refer to the items of the corresponding explanation.

Format

Request

H 1 6	(a)

(a) Selection of a function (Subcommand)
①H 0 1 :READ occupation
②H 0 2 :WRITE occupation
③H 0 5 : Change the occupation mode by own station to READ from WRITE
④H 0 6 : Change the occupation mode by own station to WRITE from READ
⑤H 0 0 : Remove the occupation by own station

Response

(a)	H 1 6	(b)

(a) Response task code (H00 when normally executed)
For the task codes other than the normal response task code, refer to the table showing the responses to each task code in Section 4.5.
(b) User's program version (H00 ~ HFF)

Explanations

(1) READ occupation (Subcommand H01)
In case own station does not occupy CPU yet, use this code in order to read a user's program, to monitor data memories or I/O, to set data, etc.

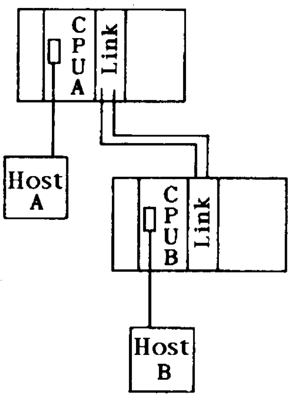
(2) WRITE occupation (Subcommand H02)
In case own station does not occupy CPU by WRITE yet, use this code in order to write a user's program. When other stations occupy the CPU this task will not be executable.

(3) Change of the mode of occupation (from WRITE to READ) (Subcommand H05)
Changes the mode of occupation to READ.
This task is not executable when own station does not occupies CPU beforehand.
Processing of a finish of alterations on parameters will be on the change of mode from WRITE to READ.

(4) Change of the mode of occupation (from READ to WRITE) (Subcommand H06)
Changes the mode of occupation to WRITE.
This task is not executable when own station does not occupies CPU beforehand.
When other stations occupy the CPU this task will not be executable.

(5) Remove the occupation (Subcommand H00)
Removes the occupation in own station.
Processing of a finish of alterations on parameters will be made on the removal of WRITE occupation.

(Link:
TLINK-02H
or LINK-02H)



Operational conditions in the status of occupations

Status of the occupation of CPU	Select functions (sub-command)				
	H01	H02	H05	H06	H00
Not occupied	○	○	×	×	○
Own station occupies WRITE	×	○	○	○	○
Other station occupies WRITE	×	×	×	×	○
Own station alone occupies READ	○	×	○	○	○
Both own and other stations occupy READ	○	×	○	×	○
Other station alone occupies READ	○	×	×	×	○
Four other stations occupy READ each	×	×	×	×	○

When the host A is own station the host B is other station and vice versa.

Task code	H 17	Forced Removal of Occupations	Classification	CPU control

Function

To force to remove a status of undue occupation that is left maintained by a failed programming device.(Selection of a function H00)
In case the first try to occupy CPU after a connection to the CPU by the host, use the selection of the function H01.
This task code is executable even when CPU is not occupied.

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	
○	○	○	○	○	No occupation	

Format

Request

H 1 7	(a)

(a) Selection of functions (Sub-command)
①H 0 0 : Force to remove all the occupations
②H 0 1 :Force to remove the occupation of own station

Response

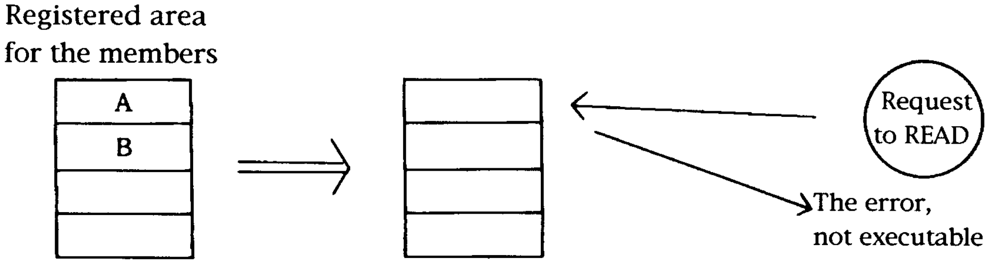
(a)	H 1 7

(a) Response task code (H00, when normally executed)
For the task codes other than the normal response task code, refer to the table showing the responses to each task code in Section 4.5.

Explanations

In this task code, when the forced removal is accompanied by a change in the parameter area, the processing will be made in the same way as that of a finish of alterations on parameters (refer to Task code H27).

① Force to remove all the occupations (Sub-command H00)
All the occupations will be removed unconditionally. Therefore, when this command is executed peripherals can not continue their READ operations (for example). Care should be taken before the execution of this task to search for the existence of peripherals under registration by monitoring the table of occupation (WRF040 ~ WRF04B) to prevent unexpected removals.

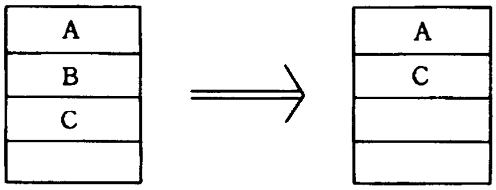


Under the occupations
(READ)
by the peripherals
A and B.

All the registrations
of the member peripherals
are removed

The error (unexecutable)
is occurred because
all the occupations are
already removed by force,
which rejects all access tries
from the peripherals
A and B.

② To force to remove the occupation of own station (Sub-command H01)
The occupation by own station is removed.
The occupations by other stations are maintained.



Under the occupations
(READ) by the peripherals
A, B and C.

Forced removal of
the occupation
of own station (B).

Example

	Selection of a function HOO	Selection of a function HO1
Request	1 7 0 0	1 7 0 1
Response	0 0 1 7	0 0 1 7

Task code	H 18	Set and Read Calendar Clock	Classification	CPU control

Function

To set and read the calendar clock built in CPU.

Format

Request

H 1 8	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)

From (b) to (h) are given only when the selection of function is H01.

(a) Selection of functions (Sub-command)

① H 0 0 To read the calendar clock

② H 0 1 To set the calendar clock

③ H 0 2 To adjust within 30 seconds (0 ~ 29 seconds → 00 second,
30 ~ 59 seconds → + 1 minute, 00 second)

(b) Year (BCD style in four digits, A. D.)
(c) Month (BCD style, H01 ~ H12)
(d) Day (BCD style, H01 ~ H31)
(e) Day of the week (H00.....Sunday, H01......Monday, H02......Tuesday,
H03.......Wednesday, H04.......Thursday,
H05......Friday, H06.....Saturday)
(f) Hour (BCD style, H00 ~ H23)
(g) Minute (BCD style, H00 ~ H59)
(h) Second (BCD style, H00 ~ H59)

Response

(a)	H 1 8	(b)	(c)	(d)	(e)	(f)	(g)	(h)

Given as the responsive data only when the selection of functions is H00.

(a) Response task code (H00, when normally executed)

*2 The contents of (b) ~ (h) are the same as those of the request task code.
For the task codes other than the normal response task code, refer to each task code in Section 4.5.

Explanations

① To read the calendar clock (Subcommand H00)
To read the data of the calendar clock built in CPU module.

② To set the calendar clock (Subcommand H01)
To set the data on the calendar clock built in CPU module.

③ To adjust within 30 seconds (Subcommand H02)
To adjust by a second the calendar clock built in CPU module.

0 ~ 29 seconds → 00 second,   30 ~ 59 seconds → + 1 minute, 00 second

Example

Request   Selection of functions (Subcommand) : H00

1 8	0 0

Response

0 0	1 8	1 9	9 1	0 3	2 1	0 4	0 8	0 5	3 0
Normal execution		1991		March (Month 03)	21st	Thursday	8 (hour)	5 (minutes)	30 (seconds)

Request

Selection of functions (Subcommand) : H01

1 8	0 1	1 9	9 1	0 4	2 0	0 6	1 6	5 0	3 0
		1991		April (Month 04)	20st	Saturday	8 (hour)	5 (minutes)	30 (seconds)

Response

0 0	1 8
Normally executed	

Task code	H 20	All Clear	Classification	Write memory

Function

To clear the specified area of the user's memory

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
×	×	×	×	×	READ occupation	Status of occupation
O	×	×	×	O	Write occupation	

Format

Request

H 2 0	(a)

(a) Selection of functions (Subcommand)

① H 0 0 : To initialize the entire area of the user's memory.

② H 0 1 : To initialize HI-FLOW area. (Note)

③ H 0 2 : To initialize HI-LADDER area.

④ H 0 3 : To clear to zero of entire area of the user's memory.

(Note) FLOW area is not supported by H-250/252.

Response

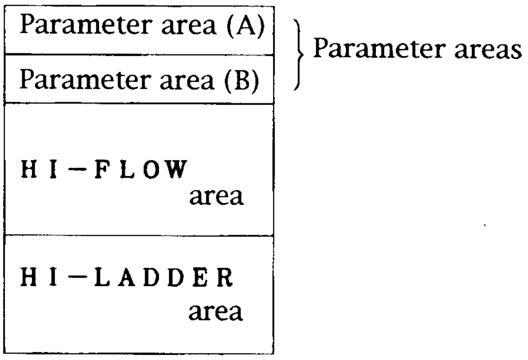
(a)	H 2 0

(a) Response task code (H00, when normally executed) For the task codes other than the normal response task code, refer to each task code in Section 4.5.

Explanations

(1) Structure of the user's memory. The structure of the user's memory is shown right. The parameter area (A) contains the assignments for I/O, memories, etc. and the area (B) contains the timer informations, etc.

User's memory



Explanations

(2) Explanations on each function

① To initialize the entire area of the user's memory. (Subcommand H00)
To initialize total area of the parameter areas (A), (B), HI-FLOW area and HI-LADDER area.
When this command is executed the memory will be assigned as follows.
The I/O assignment will also be erased.

		Assigned capacity
Parameter area ((A) + (B))	:	H0280
HI-FLOW area	:	H0000
HI-LADDER area	:	*1 (Installed capacity) - H0280

*1) Installed capacity will be known by the user's memory capacity information given by reading (task code H10) the status of CPU.

② To initialize HI-FLOW area. (Subcommand H01).
To initialize only HI-FLOW area.
(Note) FLOW area is not supported by H-250/252.

③ To initialize HI-LADDER area. (Subcommand H02).
To initialize only HI-LADDER area.

④ To clear to zero of entire area of the user's memory. (Subcommand H03).
To write zeros on the entire area of the memory actually installed in the memory pack.

When the subcommand H00 or H03 is executed in the task code H20, never fail to execute the task code H27 (To end parameter change) after completion of the writing in each area of the user's memory.

Task code	H 23	Transfer Program to Address Directed	Classification	Write memory

Function

To write the specified numbers of steps on the user's memory starting from the specified address.

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
×	×	×	×	×	READ occupation	Status of occupation
O	×	×	×	O	Write occupation	

Format

Request

H23	(a)	(b)	(c)	(d)	

(a) To specify the memory area (Subcommand)

① H 0 0 : Parameter area (A)
② H 0 1 : HI-FLOW area (Note)
③ H 0 2 : HI-LADDER area
④ H 0 3 : Parameter area (B)

(b) Address of the user's memory (Absolute address).
(c) Number of steps (H01 ~ H3C maximum 60 steps)
(d) Data written (Four bytes for one step)
(Note) FLOW area is not supported by H-250/252.

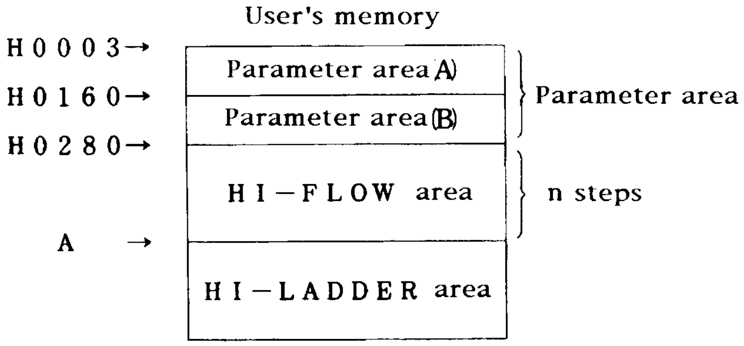
Response

(a)	H 2 3

(a) Response task code (H00, when normally executed). For the task codes other than the normal response task code, refer to each task code in Section 4.5.

Explanations

(1) The structure of user's memory.
The structure and address are shown right.



A=H0280+n steps

* The top address of HI-LADDER area is H0280, because HI-FLOW area is not used in H-250/252.

(2) Explanations for each function.

① Parameter area (A) (Subcommand H00)
To write a specified[*1] data on the parameter area (A).
* When this command is executed , never fail to execute the task code H27 (end parameter change) after completion of the writing on the memory.

② HI-FLOW area (Subcommand H01)
To write a specified program[*1] on HI-FLOW area.

③ HI-LADDER area (Subcommand H02)
To write a specified program[*1] on HI-LADDER area.

④ Parameter area (B) (Subcommand H03)
To write a specified[*1] data on the parameter area (B).
Maximum of 60 steps can be written at each access.

[*1] : The specified data or program means the data or the program that is read by the task code H31 (Read a program at a specified address). If an invalid data or a invalid program is written CPU will have a chance to stop by the error.

Task code	H 26	Write Assignment on Memory	Classification	Write memory

Function

To write informations for the assignment on the memory.

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
×	×	×	×	×	READ occupation	Status of occupation
O	×	×	×	O	Write occupation	

Format

Request

H26	H00 (a)	H 0 0 0 0 0 2 8 0 (b)	H 0 0 0 0 0 0 0 0 (c)	(d)

(a) Subcommand (H00, fixed).

(b) Capacity of the memory in the parameter area. (H00000280, fixed).

(c) Capacity of the memory in HI-FLOW area. (Hexadecimal, 8 digits).

(d) Capacity of the memory in HI-LADDER area. (Hexadecimal, 8 digits).

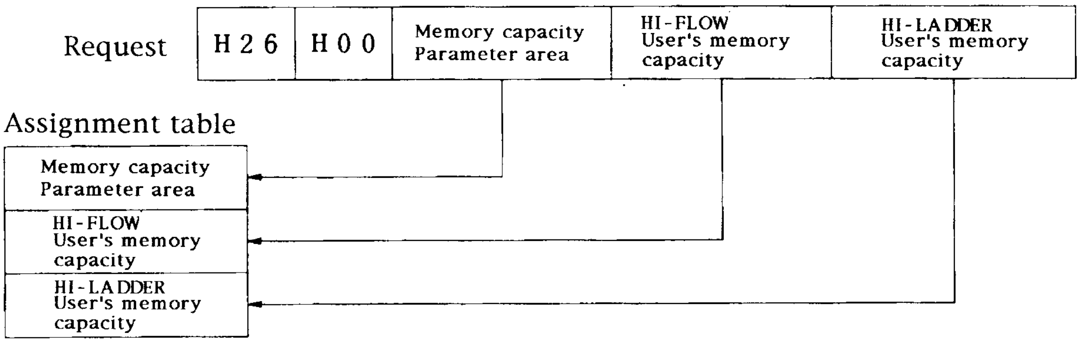
*HI-FLOW area is not supported by H-250/252. (H00000000 is fixed)

Response

(a)	H 2 6

(a) Response task code (H00, when normally executed). For the task codes other than the normal response task code, refer to each task code in Section 4.5.

Explanations



When this task code is executed , never fail to execute the task code H27 (end parameter change) after completion of the writing on the memory.

Task code	H 27	End Change of Parameters	Classification	Write memory

Function

To inform CPU that the data have been changed in the parameter area.

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
×	×	×	×	×	READ occupation	Status of occupation
O	×	×	×	O	Write occupation	

Format

Request

H 2 7

Response

(a)	H 2 7

(a) Response task code (H00, when normally executed). For the task codes other than the normal response task code, refer to each task code in Section 4.5.

Explanations

(1) To assign I/O and to enable operations of outputs/inputs and communications in accordance with the new assignment based on the informations stored in the parameter area (A) of the user's memory.

(2) In case of an initialization of the entire area of the user's memory, a zero clear, a change in the parameter area (A) or writing a memory allocation, never fail to execute this task code after the writing was completed.

(3) In case I/O assignment of communication modules are erased or slots are changed after an execution of this task code, the communications will not be possible between the host and CPU. In such case connect a programming equipment to CPU and restore the intended status by a retry of I/O assignments and other necessary operations.

Task code	H 28	Change Settings of Timer/Counter	Classification	Write memory

Function

To change settings of the timer/counter for HI-LADDER programs.

conditions of execution

① The status of CPU

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
×	×	×	×	×	READ occupation	Status of occupation
○	○	○	○ *1	○	Write occupation	

*1 Possible for DEBUG HALT.
Impossible for DEBUG RUN.

② The special internal output R7C7 is ON for the permission of the change under a running of CPU.

③ The normal scan time is within 3 seconds under a running of CPU.

④ CPU is not in a serious failure when an error is occurred.
The special internal output R7EA (Modification during RUN) is ON in cases other than shutdowns and errors.

Format

Request

H28	H02 (a)	(b)	(c)	(d)	(e)	(f1)	(f2)	(g1)	(g2)

(a) Subcommand : H02 (should be fixed).
(b) No. of Timer counter: H0000 ~ H01FF (0 ~ 511).
(c) Code of changes : H00 Not to be changed.
H01 To change the time base only.
H02 To change the 1st setting value only.
H03 To change the time base and the1st setting value.
H04 To change the 2nd setting value only (WDT command).
H05 To change the time base and the 2nd setting value (WDT command).
H06 To change the 1st setting value and the 2nd setting value (WDT Command).
H07 To change the time base, the 1st setting value and 2nd setting value (WDT command).

(Note) When no changes are made, all the data should be zero.

(Example) When the code "H05" (to change the time base and the 2nd setting value) is specified set I/O code (f1) and I/O No. (f2) of the 1st setting to zero.

	(a)	(b)	(c)	(d)	(e)	(f 1)	(f 2)	(g 1)	(g 2)
2 8	0 2	W D T No	0 5	0 2	0 0 0 0	0 0	0 0 0 0 0 0	0 7	0 0 0 0 1 A

H 0 2, Fixed

H 0 0 0 0, Fixed

"0" for no change

(d) Time base.

H00	Counter
H01	Timer 0.01 second
H02	Timer 0.1 second
H03	Timer 1 second

The counter can be set when the timer/counter No. is H0000 ~ H01FF (0 ~ 511).
The timer (0.01 seconds) can be set when the timer/counter No. is H0000 ~ H003F (0 ~ 63).
The timer (0.1 seconds) and the timer (1 second) can be set when the timer/counter No. is H0000 ~ H00FF (0 ~ 255).

(e) Address H0000 (should be fixed at H0000 in any case)

(f1) I/O code, 1st setting value
(f2) I/O No., 1st setting value
(g1) I/O code, 2nd setting value
(g2) I/O No., 2nd setting value

Type	I／Oコード	I／ONo
Constant	H 0 7	H000000~H00FFFF
WX	H 0 8	H000000~H004FF9
WY	H 0 9	H000000~H004FF9
WR	H 0 A	H000000~H0043FF
WL	H 0 B	H000000~H0003FF H001000~H0013FF
WM	H 0 C	H000000~H0003FF

Response

(a)	H 28

(a) Response task code (H00, when normally executed)

Explanations

To change the setting values of the timer/counter in LADDER program. When CPU is in an operation the change of settings will be made on an execution of END command of a normal scan without stopping this scan.

(Note) When an address is specified at a value other than H0000 the error return code H04 (abnormal address) or H0F (abnormal program) will be occurred, or a timer setting written previously in a specified address (in the user's program area) will be changed. These should be well taken into considerations.

Task code	H 31	Read Program at Specified Adress	Classification	Read memory

Function

To read a program at a specified address with a specified number of steps.

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	

Format

Request

H 3 1	H 0 0 (a)	(b)	(c)

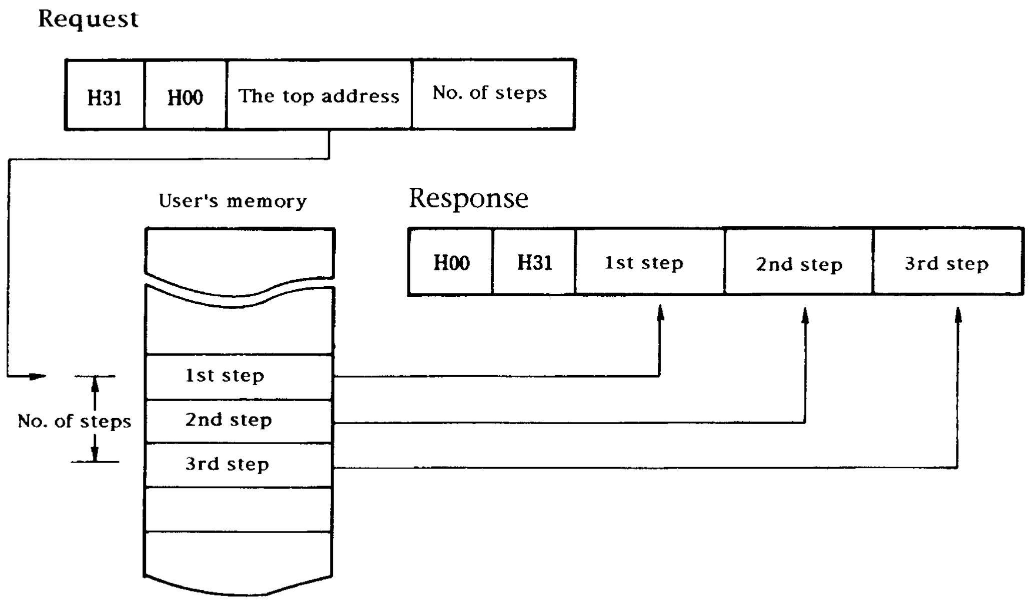
(a) Dummy (Set to H00).
(b) The top address (The absolute address in CPU).
(c) Number of steps (H01 ~ H3C Maximum 60 steps).

Response

(a)	H 3 1	(b)		(b)
		1st step		Nth step

(a) Response task code (H00, when normally executed)
For the task codes other than the normal response task code, refer to each task code in Section 4.5.
(b) Contents of the program.

Explanations



Example

To read three steps out of the address H1AB0.

Request

3 1	0 0	1 A B 0	0 3
		The top address	No. of steps

Response

0 0	3 1	4 4 1 0 0 8 0 0	4 0 8 0 4 0 0 0	4 2 D 0 8 0 0 0
		1st step	2nd step	3rd step

Task code	H 33	Search the Last Circuit	Classification	Read memory

Function

To read circuit number of the last circuit and its top address, and the number of steps in the circuit. (Only for HI-LADDER)

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	

Format

Request

H 3 3

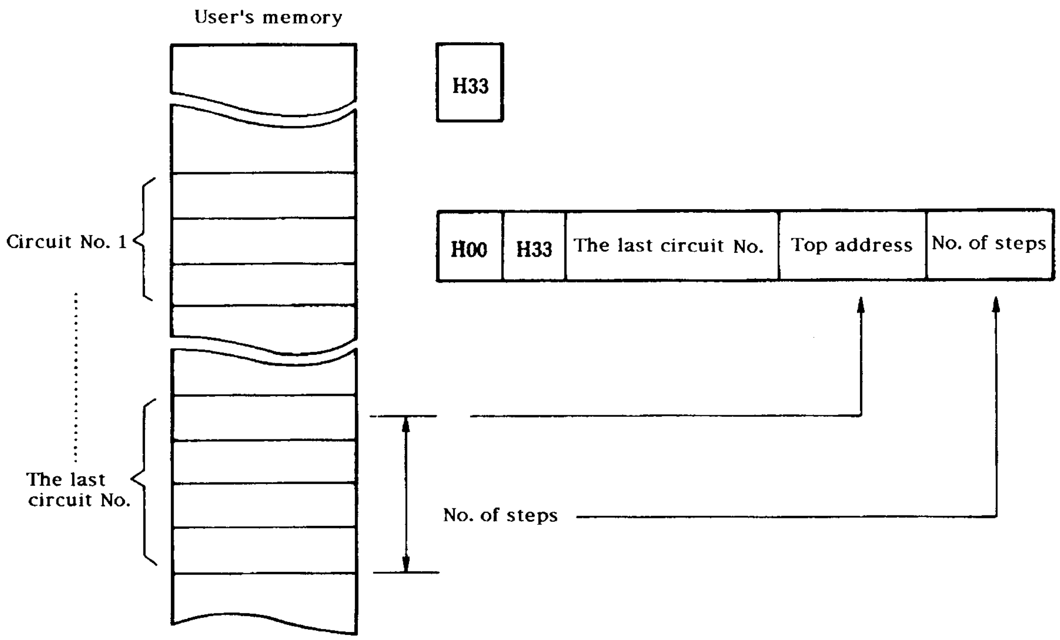
Response

(a)	H 3 3	(b)	(c)	(d)

(a) Response task code (H00, when normally executed)
For the task codes other than the normal response task code, refer to each task code in Section 4.5.
(b) Circuit No. (4 digits, hexadecimal)
(c) Top address (4 digits, hexadecimal)
(d) Number of steps (4 digits, hexadecimal)

(Note) When no program is provided in the memory.
(b) Circuit No. = H0000
(c) Top address = H0000
(d) Number of steps = H0000

Explanations



Example

To be assumed that the last circuit No. is 100 (H64), the top address is H1C80, number of steps is 10 (H0A).

Request

3	3

Response

0	0	3	3	0	0	6	4	1	C	8	0	0	0	0	A
				Circuit No.				Top address				No. of steps			

Task code	H 35	Read Memory Allocation	Classification	Read memory

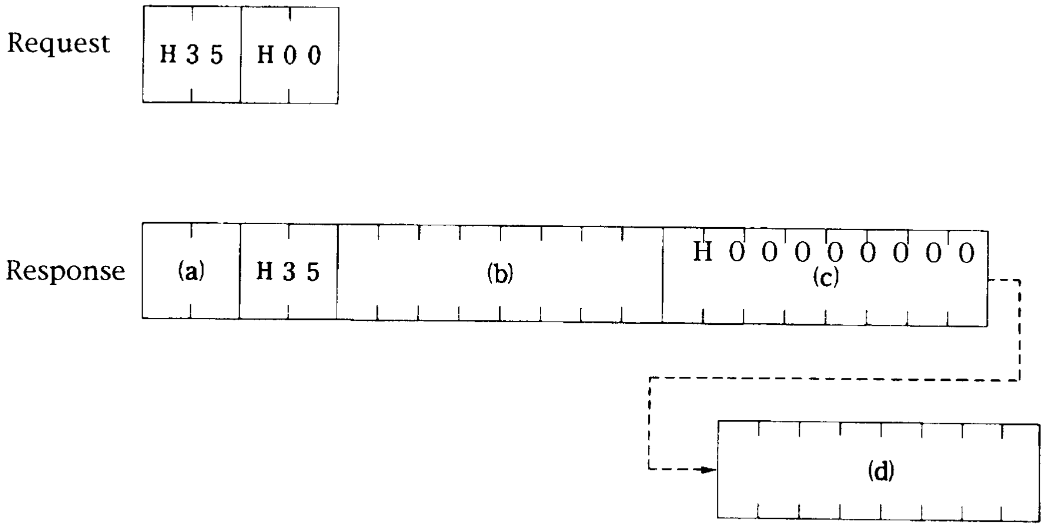
Function

To read the allocation data of memories.

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	

Format



(a) Response task code (H00, when normally executed)
For the task codes other than the normal response task code, refer to each task code in Section 4.5.

(b) Capacity of the memory in the parameter area (Hexadecimal, 8 digits).
Refer to the explanations for the task code "H20".

(c) Capacity of the memory in HI-FLOW area. (Hexadecimal, 8 digits).

(d) Capacity of the memory in HI-LADDER area. (Hexadecimal, 8 digits).

*HI-FLOW area is not supported by H-250/252.
(H00000000 is fixed)

Explanation

Request		
	H35	H00

Response					
	H00	H35	Memory capacity of parameter area	User's memory capacity of HI-FLOW	User's memory capacity of HI-LADDER

Memory assignment table

Memory capacity of parameter area
HI-FLOW User's memory capacity
HI-LADDER User's memory capacity

Example

640 (H280) steps are assigned to the parameter area, and 6 k (H1800) steps are assigned to HI-LADDER memory capacity.

Request

3	5	0	0

Response

0	0	3	5	0	0	0	0	0	2	8	0	0	0	0	0	0	0	0	0

Parameter area (0 0 0 0 0 2 8 0) | HI-FLOW (0 0 0 0 0 0 0 0)

0	0	0	0	1	8	0	0

HI-LADDER

Task code	H 40	Monitor Data on Specified I/O No. (Continuous N points)	Classification	I/O control

Function

To read a group of monitored data comprising continuous N points (word) started from a specified I/O No.

* This task code is executable even when CPU is not occupied. The return code in this case is "H02" (Own station does not occupy the CPU).

* All I/O data outside of H-250/252 will be returned as OFF (0).

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	
○	○	○	○	○	No occupation	

Format

Request

H 4 0	(a)	(b)	(c)

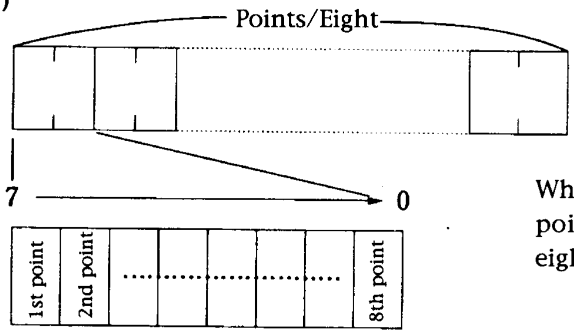
(a) I/O code (Refer to I/O code table)
(b) I/O No. (Refer to the reference table I/O No., Decimal/Hexadecimal expression).
(c) No. of bits H01 ~ HFO (1 ~ 240)
No. of words H01 ~ H78 (1 ~ 120)

Response

(a)	H 4 0	(b)

(a) Response task code (H00, when normally executed)
For the task codes other than the normal response task code, refer to each task code in Section 4.5.

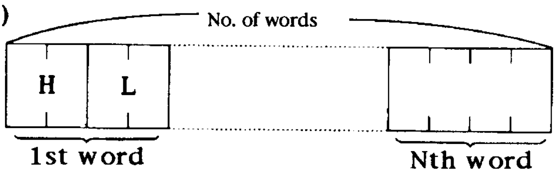
(b) Monitor data
(Bit data)



When the monitored points are less than eight empty bit = 0.

Binary image (H00 ~ HFF).

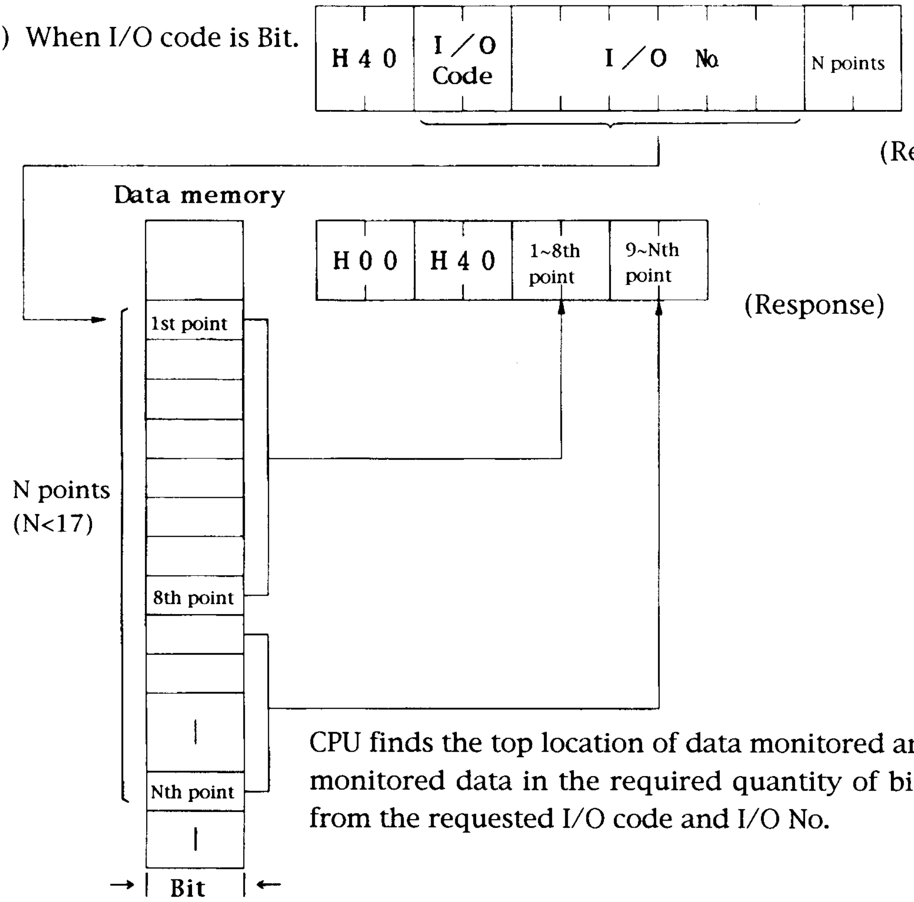
(Word data)



Binary image (H 0 0 0 0~H F F F F)

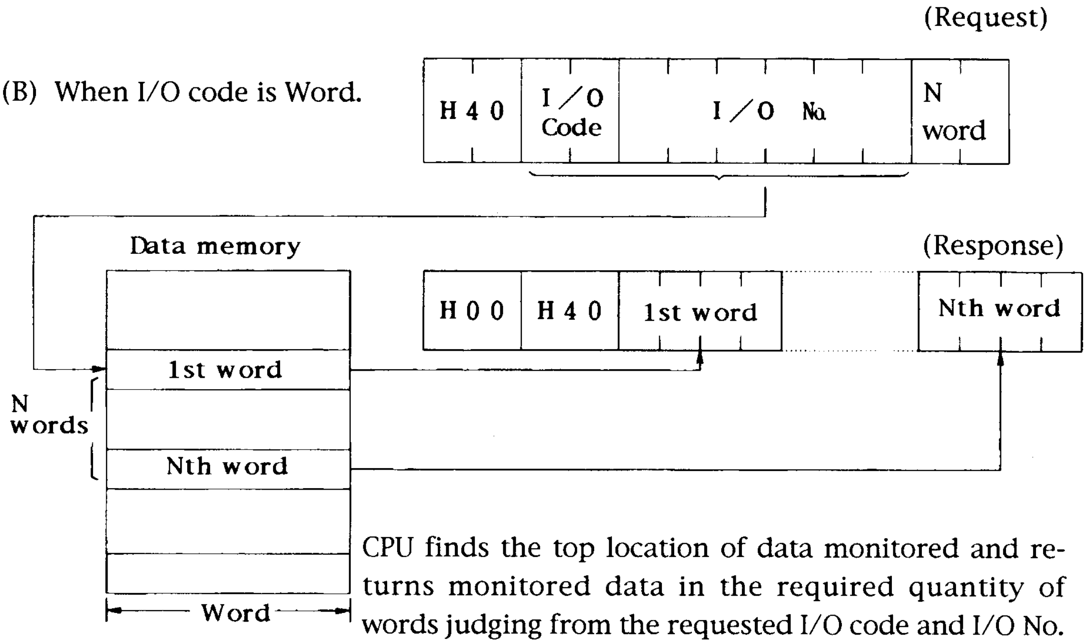
Explanations

(A) When I/O code is Bit.



CPU finds the top location of data monitored and returns monitored data in the required quantity of bits judging from the requested I/O code and I/O No.

(B) When I/O code is Word.



CPU finds the top location of data monitored and returns monitored data in the required quantity of words judging from the requested I/O code and I/O No.

Explanations

In case of monitoring external I/O given with I/O mixed assignment (WORD 4W/ 4W, BIT B1/1, BIT B2/1, BIT B1/2, BIT B2/2, FUN 0~6) in using continuous N points of Task code H40, when input I/O number of assigned I/O is specified as output I/O number, the value of output (Y) can not be correctly monitored.

Please specify the I/O number of corresponding output (Y) and monitor when you monitor the output.

* The internal output area of WR is discontinuous in WR0-3FF and WR400-43FF in H-252.

Therefore, please note the following points when you monitor N points of two areas continuously.

Only when First I/O No. is specified as DR3FE, all N points specified including DR3FE are returned to "0". Besides, the continuous N points can be normally monitored

Table 4.2.5 I/O codes

I/O codes	Symbols
H00	X
H01	Y
H02	R
H03	L
H04	M
H05	Timer/Counter
H06	CL
H07	(Not used)
H08	WX
H09	WY
H0A	WR
H0B	WL
H0C	WM
H0D	TC
H0E	DIF
H0F	DFN

Table 4.2.6 I/O No. Decimal ←→ Hexadecimal

Symbol	I／Oコード	I ／ O №	
		Decimal (Partly hexadecimal)	Hexadecimal
X	H00	00000~4FF95	H000000~ H4FF5F
Y	H01	00000~4FF95	H000000~ H4FF5F
Timer counter	H05	0 ~ 511	H000000~H0001FF
CL	H06	0 ~ 511	H000000~H0001FF
WX	H08	0000 ~ 4FF9	H000000~H004FF9
WY	H09	0000 ~ 4FF9	H000000~H004FF9
TC	H0D	0 ~ 511	H000000~H0001FF
DIF	H0E	0 ~ 511	H000000~H0001FF
DFN	H0F	0 ~ 511	H000000~H0001FF

(Note)

1. Decimal expressions for X and Y the last two digits are decimal and the first three digits are hexadecimal.
   Conversion from decimal to hexadecimal be made in the last two digits.
   (Example) 4FF90 to 4FF5A
2. Decimal expressions for WX and WY only the last one digit is decimal and the first three digits are hexadecimal.

Task code	H 42	Forced Set/Reset on Specified I/O No. (Continuous N points)	Classification	I/O control

Function

To force to Set/Reset and reset specified data on the data area of continuous N points (words) starting from a specified I/O No.

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	

Format

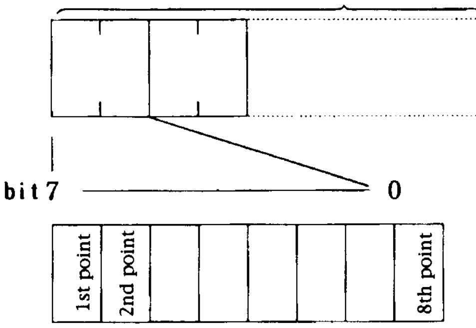
Request

H 4 2	(a)	(b)	(c)	(d)

(a) I/O code *1
(b) I/O No. *1
(c) No. of bits
H01~HC8 (1~200)
No. of words
H01~H64 (1~100)
*1 : Refer the task code H40

(d) Set/Reset of data
(Bit data)

Points/Eight

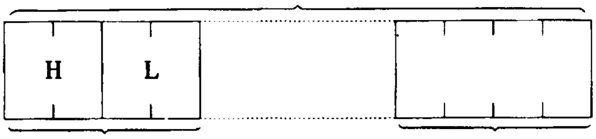


Binary image (H00 ~ HFF)

When the monitored points are less than eight, empty bit should be 0.

(Word data)

Word data values



1st word

Nth word
Binary image (H0000 ~ HFFFF)

Response

(a)	H 4 2

(a) Response task code (H00, when normally executed)
For the task codes other than the normal response task code, refer to each task code in Section 4.5.

Format

* The internal output area of WR is discontinuous in WR0-3FF and WR400-43FF in H-252.

Therefore, please note the following points when you force to set/reset N points of two areas continuously.

Only when First I/O No. is specified as DR3FE, the forced set/reset of all N points specified including DR3FF are not done and normal execution "H00" is returned. Besides, the forced set/reset of the continuous N points is possible.

Task code	H 44	Monitor Data on Specified I/O No. (Random N points)	Classification	I/O control

Function

To read a group of monitored data comprising random N points (word) at specified I/O Nos.

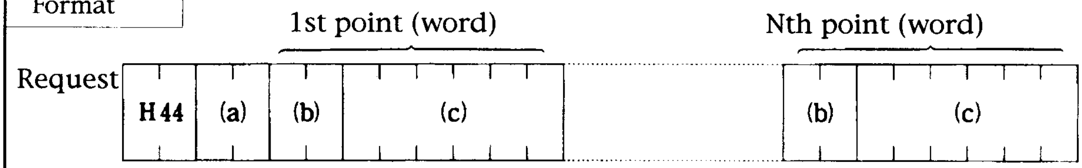
* This task code is executable even when CPU is not occupied. The return code in this case is "H02" (Own station does not occupy the CPU).

* All I/O data outside of H-250/252 will be returned as OFF (0).

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	
○	○	○	○	○	No occupation	

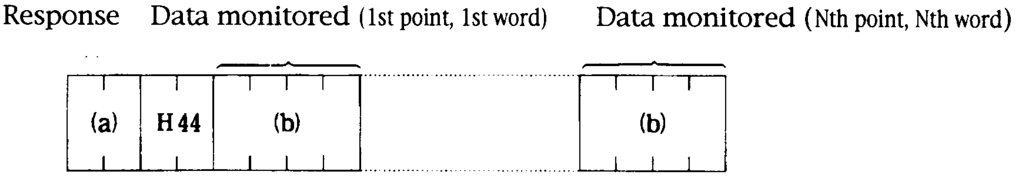
Format



(a) No. of bits/No. of words H01 ~ H3F (1 ~ 60)

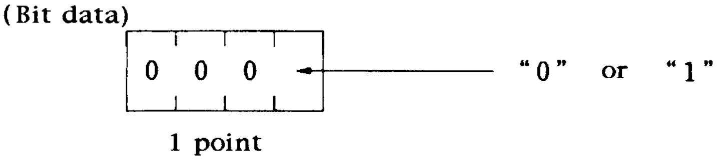
(b) I/O code } Refer to the task code H40

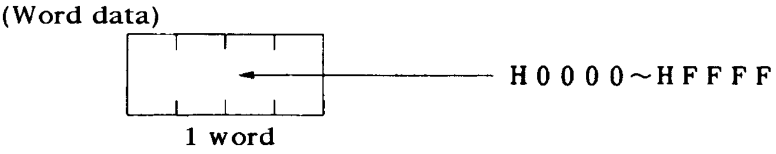
(c) I/O No.



(a) Response task code (H00, when normally executed)
For the task codes other than the normal response task code, refer to each task code in Section 4.5.

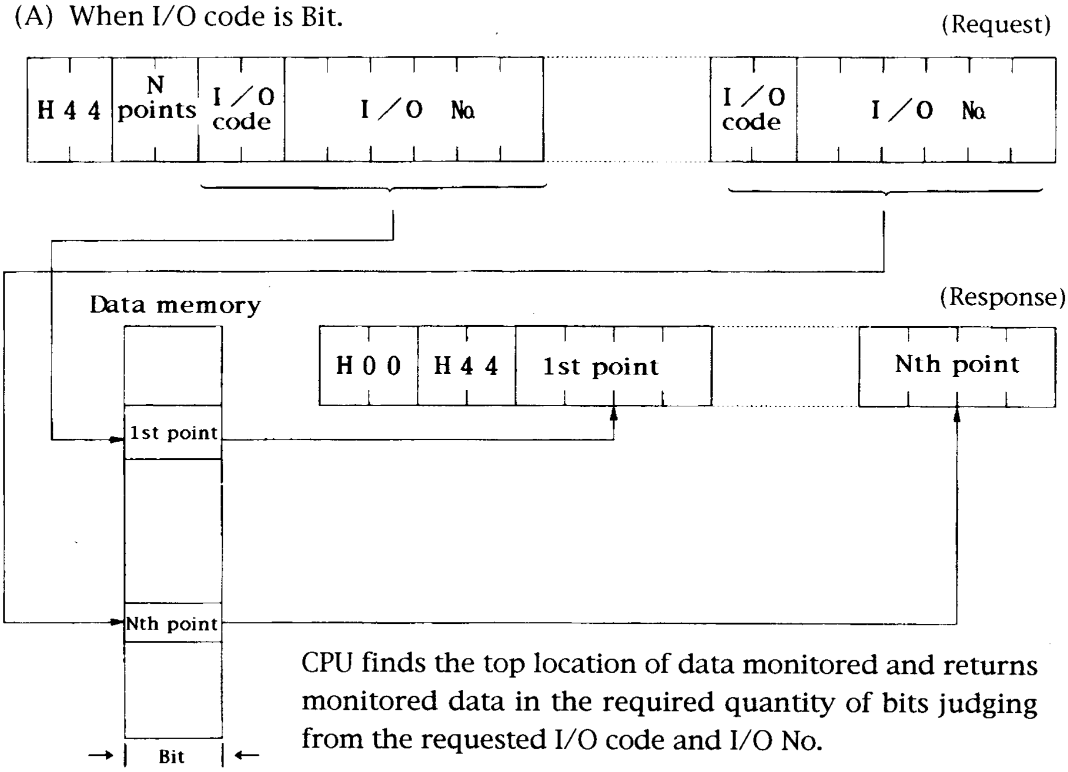
(b) Data monitored (Refer to the following illustrations)





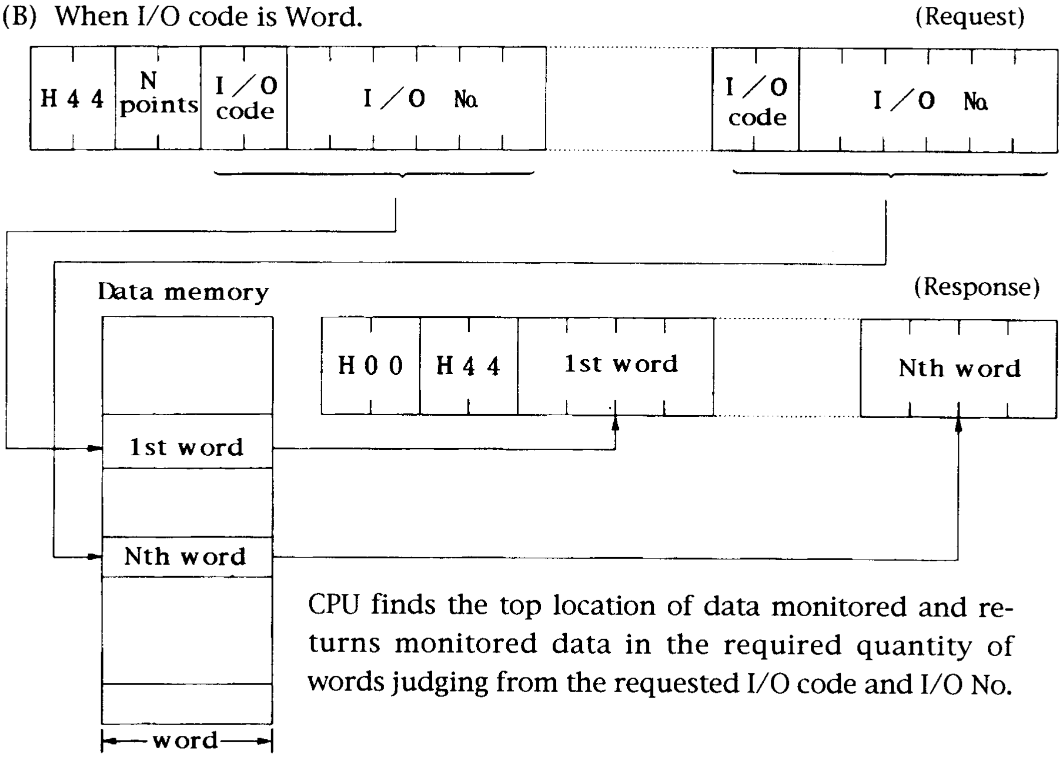
Explanations

(A) When I/O code is Bit.



CPU finds the top location of data monitored and returns monitored data in the required quantity of bits judging from the requested I/O code and I/O No.

(B) When I/O code is Word.



CPU finds the top location of data monitored and returns monitored data in the required quantity of words judging from the requested I/O code and I/O No.

Task code	H 45	Forced Set/Reset on Specified I/O Nos. (Random N points)	Classification	I/O control

Function

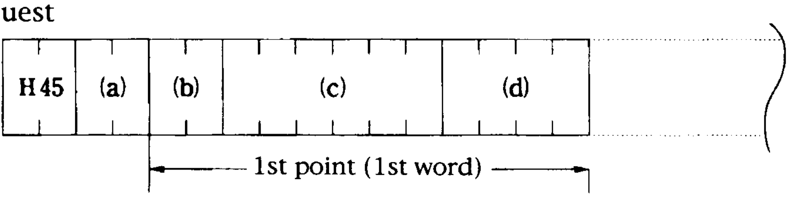
To force to set and reset specified data on the data area of random N points (words) specified by I/O Nos.

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	

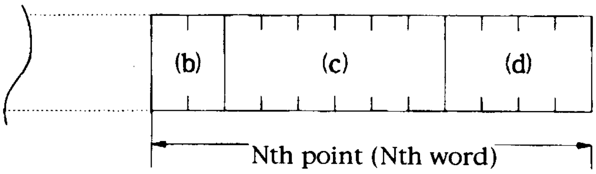
Format

Request



(a) No. of bits/No. of words H01 ~ H28 (1 ~ 40)

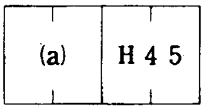
(b) I/O code } Refer to
(c) I/O No. } the task code H40
(d) Data



(Bit data)

(Word data)

Response



(a) Response task code.
For the task codes other than the normal response task code, refer to each task code in Section 4.5.
In H-250/252, the normal execution "H00" will be returned even for the I/O that is outside of the range.

Task code	HA0	Monitor Data on Specified I/O No. (Continuous N points)	Classification	I/O control

Function

To read a group of monitored data comprising continuous N points (word) started from a specified I/O No.

* This task code is executable even when CPU is not occupied.

* All I/O data outside the H-250/252 will be returned as OFF (0).

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	
○	○	○	○	○	No occupation	

Format

Request

H A 0	(a)	(b)	(c)

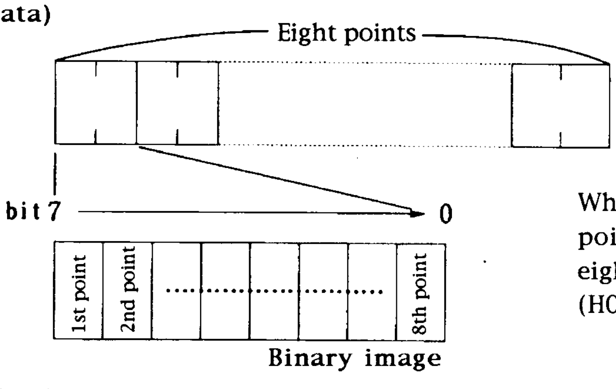
(a) I/O code } Refer to the task code H40
(b) I/O No. }
(c) No. of bits H01 ~ HFO (1 ~ 240)
No. of words H01 ~ H78 (1 ~ 120)

Response

(a)	H A 0	(b)

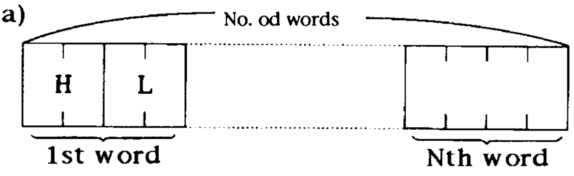
(a) Response task code (H00, when normally executed)
For the task codes other than the normal response task code, refer to each task code in Section 4.5.

(b) Monitor data
(Bit data)



When the monitored points are less than eight, empty bit = 0. (H00 ~ HFF).

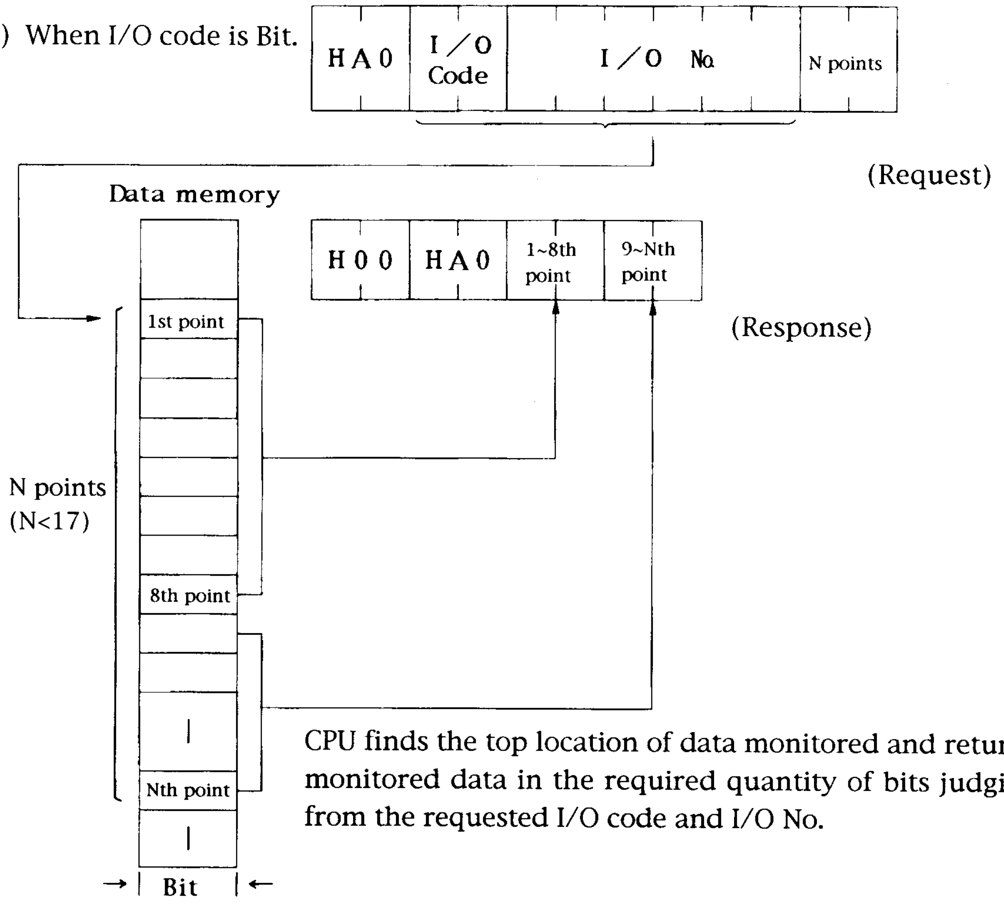
(Word data)



Binary image (H 0 0 0 0 ~ H F F F F)

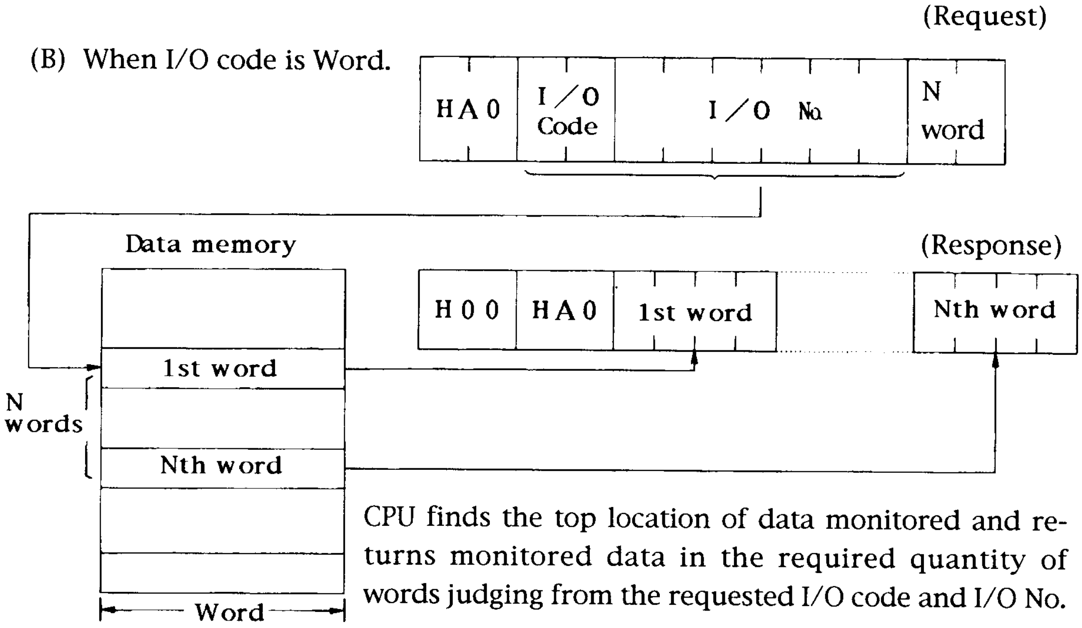
Explanations

(A) When I/O code is Bit.



CPU finds the top location of data monitored and returns monitored data in the required quantity of bits judging from the requested I/O code and I/O No.

(B) When I/O code is Word.



CPU finds the top location of data monitored and returns monitored data in the required quantity of words judging from the requested I/O code and I/O No.

Explanations

In case of monitoring external I/O given with I/O mixed assignment (WORD 4W/ 4W, BIT B1/1, BIT B2/1, BIT B1/2, BIT B2/2, FUN 0~6) in using continuous N points of Task code HA0, when input I/O number of assigned I/O is specified as output I/O number, the value of output (Y) can not be correctly monitored.

Please specify the I/O number of corresponding output (Y) and monitor when you monitor the output.

* The internal output area of WR is discontinuous in WR0-3FF and WR400-43FF in H-252.

Therefore, please note the following points when you monitor N points of two area continuously.

Only when First I/O No. is specified as DR3FE, all N points specified including DR3FE are returned to "0". Besides, the continuous N points can be normally monitored.

Task code	HA2	Forced Set/Reset on Specified I/O Nos. (Continuous N points)	Classification	I/O control

Function

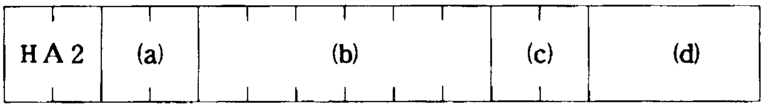
To force to Set/Reset and reset specified data on the data area of continuous N points (words) starting from a specified I/O No.

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	
○	○	○	○	○	No occupation	

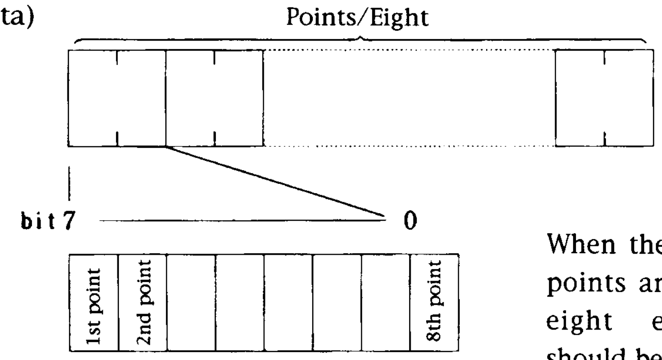
Format

Request



(a) I/O code *1
(b) I/O No. *1
(c) No. of bits
H01~HC8 (1~200)
No. of words
H01~H64 (1~100)
*1 : Refer the task code H40

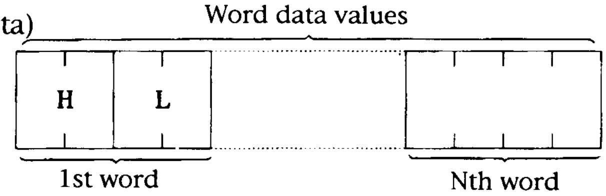
(d) Set/Reset of data
(Bit data)



Binary image (H00 ~ HFF)

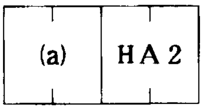
When the monitored points are less than eight empty bit should be 0.

(Word data)



Binary image (H0000 ~ HFFFF)

Response



(a) Response task code (H00, when normally executed)
For the task codes other than the normal response task code, refer to each task code in Section 4.5.
In H-250/252,the normal execution "HOO" will be returned even for the I/O that is outside of the range.

Format

* The internal output area of WR is discontinuous in WR0-3FF and WR400-43FF in H-252.

Therefore, please note the following points when you force to set/reset N points of two areas continuously.

Only when First I/O No. is specified as DR3FE, the forced set/reset of all N points specified including DR3FF are not done and normal execution "H00" is returned. Besides, the forced set/reset of the continuous N points is possible.

Task code	HA4	Monitor Data on Specified I/O No. (Random N points)	Classification	I/O control

Function

To read a group of monitored data comprising random N points (word) at specified I/O Nos.

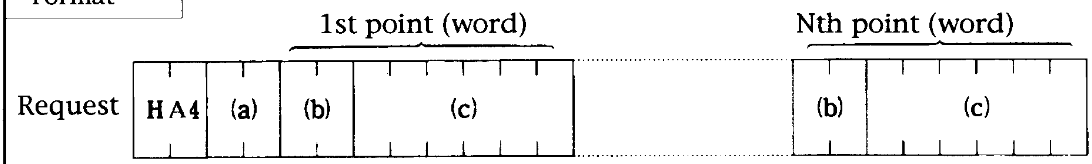
* This task code is executable even when CPU is not occupied.

* All I/O data outside the H-250/252 will be returned as OFF (0).

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	
○	○	○	○	○	No occupation	

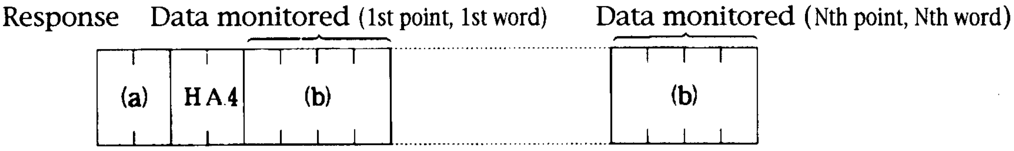
Format



(a) No. of bits/No. of words H01 ~ H3F (1 ~ 60)

(b) I/O code } Refer to the task code H40

(c) I/O No.

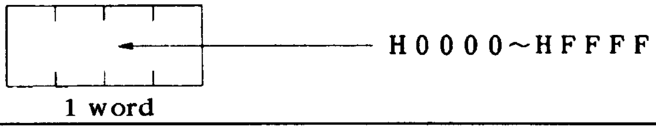


(a) Response task code (H00, when normally executed)
For the task codes other than the normal response task code, refer to each task code in Section 4.5.

(b) Data monitored (Refer to the following illustrations)

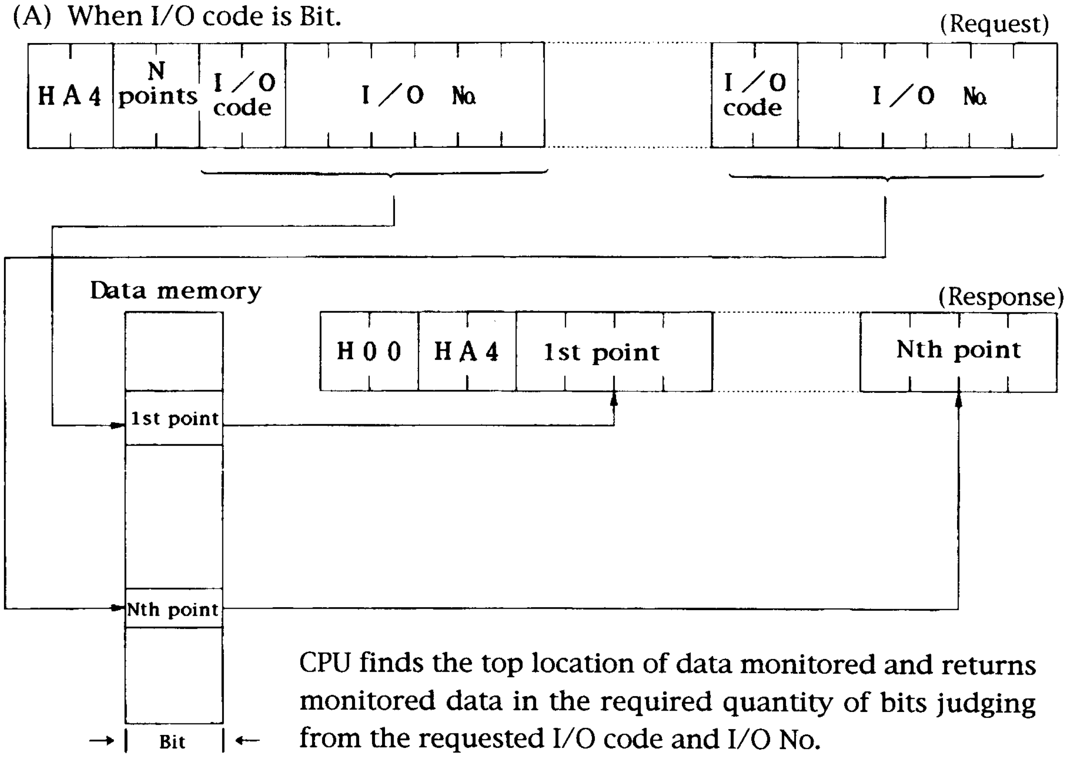
(Bit data)

(Word data)



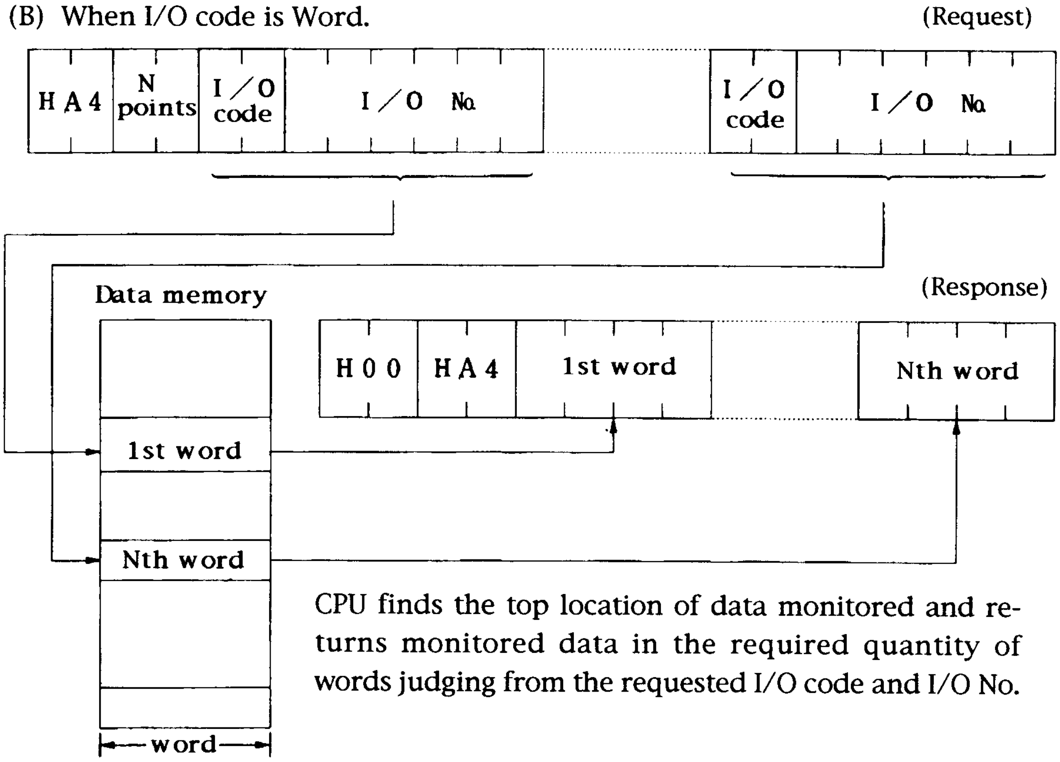
Explanations

(A) When I/O code is Bit.



CPU finds the top location of data monitored and returns monitored data in the required quantity of bits judging from the requested I/O code and I/O No.

(B) When I/O code is Word.



CPU finds the top location of data monitored and returns monitored data in the required quantity of words judging from the requested I/O code and I/O No.

Task code	HA5	Forced Set/Reset on Specified I/O Nos. (Random N points)	Classification	I/O control

Function

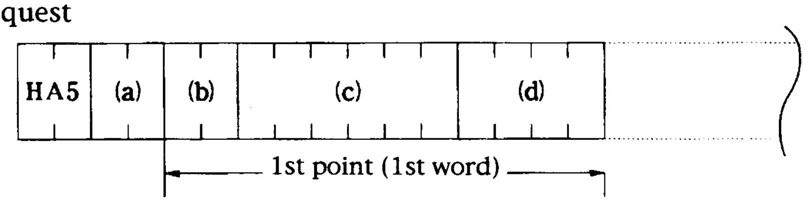
To force to set and reset specified data on the data area of random N points (words) specified by I/O Nos.
* This task code is executable even when CPU is not occupied.

conditions of execution

Status of CPU						
STOP	RUN	HALT	DEBUG	ERROR		
○	○	○	○	○	READ occupation	Status of occupation
○	○	○	○	○	Write occupation	
○	○	○	○	○	No occupation	

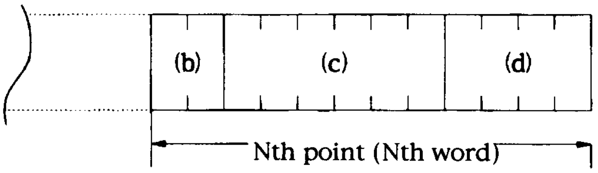
Format

Request

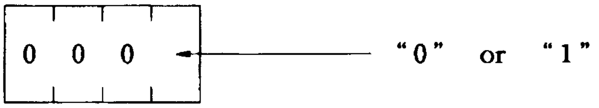


(a) No. of bits/No. of words H01 ~ H28 (1 ~ 40)

(b) I/O code } Refer to
(c) I/O No. } the task code H40
(d) Data

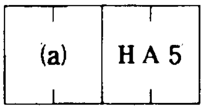


(Bit data)



(Word data)

Response



(a) Response task code.
For the task codes other than the normal response task code, refer to each task code in Section 4.5.
In H-250/252,the normal execution "HOO" will be returned even for the I/O that is outside of the range.

Table 4.5 Table of Response Task codes

Task code	Subcommand		Response task code		Return code		Error description
		Code		Code		Code	
H10	Status	H00	Normal execution	H00			
	Memory status	H01	Normal execution	H00			
	Software version	H02	Normal execution	H00			
	CPU error code	H03	Normal execution	H00			
	Read CPU name	H04	Normal execution	H00			
	Reserved sub-command	H05	Do not use				
	Not defined	H06 ~ HFF	Abnormal task code	H01	Sub-command not defined	H02	Requested subcommand is not defined.
H11	STOP	H00	Normal execution	H00			
			Unexecutable	H03	Incorrect operation	H0C	Key switch is not in STOP position. DIPSW No.4 is not in ON position. REMOTE STOP is not permitted.
	RUN	H01	Normal execution	H00			
			Unexecutable	H03	Incorrect operation		Key switch is not in STOP position. DIPSW No.4 is not in ON position. CPU status is in an error.. CPU status is in debugging. Other station occupies WRITE. Remote RUN is not permitted. User's operational conditions are not fulfilled
					Combined errors	H0E	CPU is not permitted to RUN caused by the task code to prohibit RUN.
	Not defined	H02 ~ HFF	Abnormal task code	H01	Sub-command not defined	H02	Requested subcommand is not defined.

Task code	Subcommand		Response task code		Return code		Error description
		Code		Code		Code	
H16	Status	H00	Normal execution	H00			
	READ occupation	H01	Normal execution	H00			
			Unexecutable	H03	Different occupation code	H04	Own station occupies CPU by WRITE.
					Overflow of occupations	H06	Other four stations occupy READ already.
					Occupation by other station	H08	Other station occupies CPU by WRITE
	Write occupation	H02	Normal execution	H00			
			Unexecutable	H03	Different occupation code	H03	Own station occupies CPU by WRITE.
					Other station is debugging.	H05	Other station is debugging
					Other stations occupy CPU.	H08	Other station occupies CPU.
	Change the mode of occupation WRITE to READ	H05	Normal execution	H00			
			Unexecutable	H03	No occupation	H07	Own station has no occupation.
	Change the mode of occupation READ to WRITE	H06	Normal execution	H00			
			Unexecutable	H03	Other station is debugging	H05	Other station is debugging
					No occupation	H07	Own station has no occupation.
					Other station occupies CPU.	H08	Other station occupies CPU. (Able when own station alone has an occupation).
	Not defined	H03 H04 H07 and more	Abnormal task code	H01	Sub-command not defined	H02	Requested subcommand is not defined.

Task code	Subcommand		Response task code		Return code		Error description
		Code		Code		Code	
H17	Forced to remove the occupation of own station	H00	Normal execution	H00			
	Forced to remove all the occupations	H01	Normal execution	H00			
	Not defined	H02 ~ HFF	Abnormal task code	H01	Sub-command not defined	H02	Requested subcommand is not defined.
H18	Read the calendar clock	H00	Normal execution	H00			
	Set the calendar clock	H01	Normal execution	H00			
			Abnormal task code	H01	Abnormal number of words	H05	Data length is too long.
			Unexecutable	H03	Data error	H0C	Incorrect data is set.
	Adjust in 30 seconds	H02	Normal execution	H00			
	Not defined	H03 ~ HFF	Abnormal task code	H01	Sub-command not defined	H02	Requested subcommand is not defined.
H20	Initialize the total area of user's memory.	H00	Normal execution	H00			
			Unexecutable	H03	Different occupation code	H03	Own station occupies CPU by READ.
					No occupation	H07	Own station has no occupation.
					RAM error	H0A	Mismatch is found by READ/WRITE check.
					CPU in operation	H0B	CPU is in operation
	Initialize HI-FLOW area *	H01	Normal execution	H00			
			Unexecutable	H03	ROM memory	H01	Memory pack is EPROM
					Different occupation code	H03	Own station occupies CPU by READ.
					No occupation	H07	Own station has no occupation.

* H-250/252 does not support HI-FLOW (Normal execution "H00" will be returned).

Task code	Subcommand		Response task code		Return code		Error description
		Code		Code		Code	
H20					CPU in operation	H0B	CPU is in operation and not in mode changes during RUN.
	Initialize HI-LADDER area	H02	Normal execution	H00			
			Unexecutable	H03	ROM memory	H01	Memory pack is EPROM.
					Different occupation code	H03	Own station occupies CPU by READ.
					No occupation	H07	Own station has no occupation.
					CPU in operation	H0B	CPU is in operation and does not accept mode changes.
	Initialize the total area of user's memory.	H03	Normal execution	H00			
			Unexecutable	H03	ROM memory	H01	Memory pack is EPROM
					Different occupation code	H03	Own station occupies CPU by READ.
					No occupation	H07	Own station has no occupation.
					CPU in operation	H0B	CPU is in operation .
	Not defined	H04 ~ HFF	Abnormal task code	H01	Sub-command not defined	H02	Requested sub-command is not defined.
H23	Write on the parameter area (A).	H00	Normal execution	H00			
			Abnormal task code	H01	Abnormal address	H04	Address to transfer is out of the specified.
					Abnormal number in steps or words	H05	Number of the steps to transfer is out of the specified area of the parameter (A).
					Memory overflow	H09	Number of addresses + steps exceed the parameter area.
			Unexecutable	H03	ROM memory	H01	Memory pack is EPROM.
					Different occupation code	H03	Own station occupies CPU by READ.

Task code	Subcommand	Code	Response task code	Code	Return code	Code	Error description
H23					No occupation	H07	Own station has no occupation.
					CPU in operation.	H0B	CPU is in operation.
	Write on HI-FLOW area *	H01	Normal execution	H00			
			Abnormal task code	H01	Abnormal address	H04	Address to transfer is out of the specified.
					Abnormal number in steps and words	H05	Number of the steps to transfer is out of the specified.
					Memory overflow	H09	Number of addresses + steps exceed the parameter area.
			Unexecutable	H03	ROM memory	H01	Memory pack is EPROM.
					Different occupation code	H03	Own station occupies CPU by READ.
					No occupation	H07	Own station has no occupation.
					CPU in operation.	H0B	CPU is in operation.
	Write on HI-LADDER area	H02	Normal execution	H00			
			Abnormal task code	H01	Abnormal address	H04	Address to transfer is out of the specified.
					Abnormal number in steps and words	H05	Number of the steps to transfer is out of the specified.
					Memory overflow	H09	Number of addresses + steps exceed the LADDER area.
			Unexecutable	H03	ROM memory	H01	Memory pack is EPROM.
					Different occupation code	H03	Own station occupies CPU by READ.
					No occupation	H07	Own station has no occupation.
					CPU in operation.	H0B	CPU is in operation and not in a mode change during RUN.

* H-250/252 does not support HI-FLOW (Normal execution "H00" will be returned).

Task code	Subcommand		Response task code		Return code		Error description
		Code		Code		Code	
H23	Write on the parameter area (B)	H03	Normal execution	H00			
			Abnormal task code	H01	Abnormal address	H04	Address to transfer is out of the specified.
					Abnormal number in steps and words	H05	Number of the steps to transfer is out of the specified.
					Memory overflow	H09	Number of addresses + steps exceed the parameter area.
			Unexecutable	H03	ROM memory	H01	Memory pack is EPROM.
					Different occupation code	H03	Own station occupies CPU by READ.
					No occupation	H07	Own station has no occupation.
					CPU in operation.	H0B	CPU is in operation and not in mode changes during RUN
		H04 ~ HFF	Abnormal task code	H01	Sub-command not defined	H02	Requested subcommand is not defined.
H26	Allocation of data on memories	H00	Normal execution	H00			
			Abnormal task code	H01	Memory size over		Total memory assignment exceeded the physical capacity.
			Unexecutable	H03	ROM memory	H01	Memory pack is EPROM.
					Different occupation code	H03	Own station occupies CPU by READ.
					No occupation	H07	Own station has no occupation.
					CPU in RUN operation.	H0B	CPU is in operation and not in mode changes during RUN.
	Not defined	H01 ~ HFF	Abnormal task code	H01	Sub-command not defined	H02	Requested sub-command is not defined.

Task code	Subcommand		Response task code		Return code		Error description
		Code		Code		Code	
H27			Normal execution	H00			
			Unexecutable	H03	ROM memory	H01	Memory pack is EPROM.
					Different occupation code	H03	Own station occupies CPU by READ.
					No occupation	H07	Own station has no occupation.
					CPU in operation.	H0B	CPU is in operation.
H28		H02	Normal execution	H00			
			Abnormal task code	H01	Sub-command not defined	H02	1. Memory area is not specified at H02. 2. Code changed to H08 or more.
					Abnormal address.	H04	Specified address is not on H0000, nor on the user's program area.
					Abnormal number of steps or words.	H05	The number of steps to transfer is out of the specified.
					Abnormal I/O code.	H06	1. I/O code is abnormal. 2. Time base is H04 or more.
					Abnormal I/O No.	H07	1. I/O No. is abnormal. 2. Timer/Counter number is 512 or more. 3. In the watch dog timer 1st set value (constant) >= 2nd set value (constant). 4. Time base is specified at 0.01, 0.1 or 1.0 second when the timer counter number is 256 or more. 5. Time base is specified at 0.01 second when the timer counter number is 64 or more.
			Unexecutable	H03	ROM memory	H01	Memory pack is EPROM.

Task code	Subcommand		Response task code		Return code		Error description
		Code		Code		Code	
H28					Improper parameters	H02	The specified timer counter number is used in HI-FLOW program.
					Different occupation code	H03	Own station occupies CPU by READ.
					No occupation	H07	Own station has no occupation on CPU.
					CPU in debugging operation	H0B	CPU is in debug running.
					Incorrect operation	H0C	1. R7C7 is not in ON. 2. CPU is in a serious failure. 3. Scan time is over 3 seconds.
					No program	H0D	No HI-LADDER program provided. *
					Improper program	H0F	1. Time base is specified at 0.01, 0.1 or 1.0 second for the counter. 2. Counter is specified at time base for the timer. 3. A change in the second set value is specified in the code for change, in cases not in WDT. 4. Timer is not programmed in the user's address so specified.
H31		H00	Normal execution	H00			
			Abnormal task code	H01	Abnormal address	H04	Specified address exceeds the actual memory capacity.
					Abnormal number of steps	H05	Number of steps to be read is out of the range 1 ~ 60.
			Unexecutable	H03	No occupation	H07	Own station has no occupation

* When there is no program in H-250/252, the Return code of H28 of task code is not "no program"(H0D) but "a program irrationality"(H0F).

Task code	Subcommand		Response task code		Return code		Error description
		Code		Code		Code	
H33			Normal execution	H00			
			Unexecutable	H03	No occupation	H07	Own station has no occupation
H35	Read memory allocations	H00	Normal execution	H00			
			Unexecutable	H03	No occupation	H07	Own station has no occupation
	Reserved sub-command	H02 H04	Do not use.				
	Not defined	H01 H03 H05 ~ HFF	Abnormal task code	H01	Sub-command not defined.	H02	Requested command is not defined.
H40			Normal execution	H00			
			Abnormal task code	H01	Abnormal No. of steps or words.	H05	Number of the required points is out of the specified.
					Abnormal I/O code	H06	Requested type code of I/O is not defined, or is I/O of a type that is impossible to be monitored.
					Abnormal No. of I/O	H07	Requested I/O No. is out of the specified.
			Alarm	H02			Own station has no occupation.
H42			Normal execution *	H00			
			Abnormal task code	H01	Abnormal No. of steps or words.	H05	Number of the required points is out of the specified.
					Abnormal I/O code	H06	Requested type code of I/O is not defined, or is I/O of a type that is not to be set by force.
					Abnormal No. of I/O	H07	Requested I/O No. is out of the specified.
			Unexecutable	H03	No occupation	H07	Own station has no occupation.

* In H-250/252, the normal end "H00" will be returned even for the range that is outside the specified I/O range.

Task code	Subcommand		Response task code		Return code		Error description
		Code		Code		Code	
H44			Normal execution	H00			
			Abnormal task code	H01	Abnormal No. of steps or words.	H05	Number of the required points is out of the specified.
					Abnormal I/O code	H06	Requested type code of I/O is not defined, or is I/O of a type that is not to be set by force.
					Abnormal No. of I/O	H07	Requested I/O No. is out of the specified.
			Alarm	H02			Own station has no occupation
H45			Normal execution *	H00			
			Abnormal task code	H01	Abnormal No. of steps or words.	H05	Number of the required points is out of the specified.
					Abnormal I/O code	H06	Requested type code of I/O is not defined, or is I/O of a type that is not to be set by force.
					Abnormal No. of I/O	H07	Requested I/O No. is out of the specified.
			Unexecutable	H03	No occupation	H07	Own station has no occupation
HA0			Normal execution	H00			
			Abnormal task code	H01	Abnormal No. of steps or words.	H05	Number of the required points is out of the specified.
					Abnormal I/O code	H06	Requested type code of I/O is not defined, or is I/O of a type that is impossible to be monitored.
					Abnormal No. of I/O	H07	Requested I/O No. is out of the specified.

* In H-250/252, the normal end "H00" will be returned even for the range that is outside the specified I/O range.

Task code	Subcommand		Response task code		Return code		Error description
		Code		Code		Code	
HA2			Normal execution *	H00			
			Abnormal task code	H01	Abnormal No. of steps or words.	H05	Number of the required points is out of the specified.
					Abnormal I/O code	H06	Requested type code of I/O is not defined, or is I/O of a type that is not to be set by force.
					Abnormal No. of I/O	H07	Requested I/O No. is out of the specified.
HA4			Normal execution	H00			
			Abnormal task code	H01	Abnormal No. of steps or words.	H05	Number of the required points is out of the specified.
					Abnormal I/O code	H06	Requested type code of I/O is not defined, or is I/O of a type that is not to be set by force.
					Abnormal No. of I/O	H07	Requested I/O No. is out of the specified.
HA5			Normal execution *	H00			
			Abnormal task code	H01	Abnormal No. of steps or words.	H05	Number of the required points is out of the specified.
					Abnormal I/O code	H06	Requested type code of I/O is not defined, or is I/O of a type that is not to be set by force.
					Abnormal No. of I/O	H07	Requested I/O No. is out of the specified.

* In H-250/252, the normal end "H00" will be returned even for the range that is outside the specified I/O range.

(Note 1) For details of the return command code to each task code, refer to the explanation tables in Item (1) Section 4.4 for codes H00, H01, H02, H03, H05, H08. In some of these tables explanations for the response command codes H02, H05 and H08 are omitted for simplicity because these commands will possibly be common to most of the task code cases.

(Note 2) The return codes will be occurred in the case of the response task code H01 and H03.

4.6 Example of a program

This example shows a communication by a personal computer with H- series CPU. The system comprises the components shown in the following illustrations.

The program in the host side is written by BASIC language.

This program will help you understand how the task code will function.

In actual applications where CPU module is controlled from a host, the system structure, timings, remedies for failures and other necessary items should well be taken into considerations.

(1) Structure of the program

Main program

Main processings
1000 ~ 1410

Inputs of NA,
and Task codes
1420 ~ 1850

Subroutine

Receive response data from CPU 2000 ~ 2110

Sum up the received data 3000 ~ 3170

Display the received data 4000 ~ 4270

Initialize GPIB processing 5000 ~ 5070

Display error messages 6000 ~ 6170

(2) Program list

```
100 ' SAVE "PC98",A
200 '
300 '
1000 '*******************************************
1010 '*                                         *
1020 '*       COMMUNICATION  SAMPLE PROGRAM     *
1030 '*                                         *
1040 '*******************************************
1050 '
1060 CLS:DIM CODE$(2):NAOLD$="":CMAOLD$="":TM$="F"
1070 '
1080 '
                        OPEN  COMMUNICATION PORT
1090 '
1100 INPUT "HOST TYPE ? B16=0,PC98=1";A              ' Select a host
1110 IF A >=1 THEN 1140 ELSE 1120
1120 OPEN "COM1:4800,E,7,1,CS0,DS0,CD0" AS #1        ' Open the port
1130 GOTO 1450
1140 OPEN "COM:E71NN" AS#1
1150 GOTO 1450
1160 '
1170 NAKCTR=0
1175 TSK$=CHR$(2)+CODE$(1)+SM$(1)+CHR$(13)         '  Generate  the task code
1180 PRINT #1,TSK$;:                                '  Send the task code
1190 GOSUB 2030
1200 IF LEFT$(R$,1)=CHR$(6) THEN 1240:'              In case ACK is received
1210 GOSUB 5030:'                                    In case NAK or failures is received
1220 IF NAKCTR<=3 THEN 1180:'                        Retry up to three times
1230 GOSUB 6030:GOTO 1360:'                          Communication error
1240 PRINT #1,CHR$(5)+TM$+"1"+CHR$(13);:'            Send ENQ1
1250 GOSUB 2030
1260 IF LEFT$(R$,1)=CHR$(2) THEN 1310:'              In case STX is received
1270 IF LEFT$(R$,1)=CHR$(4) THEN 1240:'              In case EOT is received
1280 GOSUB 5030:'                                    In case NAK or failures is received
1290 IF NAKCTR<=3 THEN 1180:'                        Retry up to three times
1300 GOSUB 6030:GOTO 1360:'                          Communication error
1310 GOSUB 3030
1320 IF SERR<>0 THEN 1180:'
1330 GOSUB 4030                                      Sum error ?
1340 PRINT
1350 IF MID$(R$,10,2)<>"00" THEN GOSUB 6130:GOTO 1360:' In the case of an abnormal return
1360 PRINT "CONTINUE? (Y/N)"                           Press any key to terminate
1370 CNT$=INKEY$                '
1380 IF CNT$="Y" OR CNT$=CHR$(13) THEN 1450
1390 IF CNT$="N" THEN CLOSE 1:GOTO 1410
1400 GOTO 1370
1410 STOP
1420 '
1430 ' INPUT ADDRESS (L U M P) AND TASK CODE INPUT
1440 '
1450 FOR J=1 TO 2
1460    K=0:C$=""
1470    RESTORE 1850
1480    FOR I=1 TO J
1490       READ MSG$,LMT
1500    NEXT I
1510    PRINT MSG$;
1520 '  LOCATE ,,1:A$=INPUT$(1):LOCATE ,,0
1530              A$=INPUT$(1)
1540    IF K=0 AND A$=CHR$(13) THEN
         IF J=1 THEN NA$=NAOLD$:PRINT NA$:GOTO 1660
```

```
           ELSE CMA$=CMAOLD$:PRINT CMA$:GOTO 1660
1550     IF J=2 AND A$=CHR$(13) THEN PRINT:GOTO 1650
1560     RESTORE 1840
1570     FOR N=1 TO 16
1580        READ B$
1590        IF A$=B$ THEN 1620
1600     NEXT N
1610     BEEP:GOTO 1520
1620     PRINT A$;
1630     C$=C$+A$:K=K+1
1640     IF K<LMT THEN 1520 ELSE PRINT
1650     IF J=1 THEN NA$=C$:NAOLD$=C$ ELSE CMA$=C$:CMAOLD$=C$
1660     PRINT
1670 NEXT J
1680 CODE$(1)=TM$+NA$+CMA$
1690 SM$(0)="":SM$(1)="":SM$(2)=""
1700 FOR Q=0 TO 2
1710     SUM=0
1720     FOR P=1 TO LEN(CODE$(Q))
1730        SUM=SUM+ASC(MID$(CODE$(Q),P,1))
1740     NEXT P
1750     SM$(Q)=RIGHT$(HEX$(SUM),2)
1760 NEXT Q
1770 PRINT  "EXECUTE ?"
1780 LOCATE ,,1:A$=INPUT$(1):LOCATE ,,0
1790 IF A$="" THEN 1780
1800 PRINT:PRINT:IF A$<>CHR$(13) AND A$<>"Y" AND A$<>"y" THEN 1450
1810 IF LEN(NA$)=0 OR LEN(CMA$)=0 THEN BEEP:GOTO 1450
1820 GOTO 1170
1830 '
1840 DATA 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F
1850 DATA "NETWORK ADDRESS ? ",8,"COMMAND & DATA  ? ",484
2000 '
2010 '  RECEIVED, RESPONSE DATA
2020 '
2030 EFLG=0:RXLENG=0:R$="":R1$=""
2040 WHILE EFLG=0
2050     WANS$=""
2060     WANS$=INPUT$(1,1)
2070     IF WANS$=CHR$(&HD) THEN EFLG=1:GOTO 2100
2080     IF RXLENG<255 THEN R$=R$+WANS$ ELSE R1$=R1$+WANS$
2090     RXLENG=RXLENG+1
2100 WEND
2110 RETURN
3000 '
3010 '  SUM UP RECEIVED DATA
3020 '
3030 SUM1=0:SERR=0:SD$=R$
3040 IF RXLENG>255 THEN SD$=R1$:GOTO 3090
3050 FOR I=2 TO LEN(R$)-2
3060     SUM1=SUM1+ASC(MID$(R$,I,1))
3070 NEXT I
3080 GOTO 3150
3090 FOR I=2 TO LEN(R$)
3100     SUM1=SUM1+ASC(MID$(R$,I,1))
3110 NEXT I
3120 FOR I=1 TO LEN(R1$)-2
3130     SUM1=SUM1+ASC(MID$(R1$,I,1))
3140 NEXT I
3150 RSUM$=RIGHT$(SD$,2)
```

```
3160 IF RIGHT$("0"+HEX$(SUM1),2)<>RSUM$ THEN SERR=1
3170 RETURN
4000 '
4010 ' DISPLAY RECEIVED DATA
4020 '
4030 DP=16:R2$=R$:IF LEN(R1$)<>0 THEN R2$=R1$
4040 PRINT "L U M P        = ";MID$(R$,2,8)
4050 PRINT " RESPONSE CODE = ";MID$(R$,10,4)
4060 IF (LEN(R$)-2)=13 THEN 4260
4070 PRINT " RESPONSE DATA= ";
4080 FOR I=14 TO LEN(R$)-2
4090    DP=DP+1
4100    IF DP>80 THEN DP=17:PRINT:PRINT SPACE$(16);
4110    DISP$=MID$(R$,I,1):PRINT DISP$;
4120 NEXT I
4130 IF LEN(R1$)=0 THEN 4250
4140    DP=DP+1
4150    IF DP>80 THEN DP=17:PRINT:PRINT SPACE$(16);
4160    DISP$=MID$(R$,I,1):PRINT DISP$;
4170    DP=DP+1
4180    IF DP>80 THEN DP=17:PRINT:PRINT SPACE$(16);
4190    DISP$=RIGHT$(R$,1):PRINT DISP$;
4200 FOR I=1 TO LEN(R1$)-2
4210    DP=DP+1
4220    IF DP>80 THEN DP=17:PRINT:PRINT SPACE$(16);
4230    DISP$=MID$(R1$,I,1):PRINT DISP$;
4240 NEXT I
4250 PRINT
4260 PRINT "  SUM VALUE     = ";RIGHT$(R2$,2)
4270 RETURN
5000 '
5010 'INITIALIZE COMMUNICATION TO GPIB
5020 '
5030 PRINT #1,CHR$(15)+"FF"+CHR$(13);'    Command to abort
5040 FOR K=0 TO 100
5050 NEXT K
5060 NAKCTR=NAKCTR+1
5070 RETURN
6000 '
6010 'ERROR MESSAGE
6020 '
6030 BEEP
6040 PRINT "COMMUNICATION ERROR (CHECK SENT TEXT)   "
6050 PRINT
6060 RETURN
6070 '
6130 BEEP
6140 PRINT "RETURN ERROR (CHECK SENT TEXT)  "
6150 PRINT
6160 RETURN
6170 '
```

Appendix 1 Self-Diagnosis Error Codes

Listed below are self diagnosis errors and measures to be taken. Each error code will be output in WRF000 of special internal output in hexadecimal.

Error code	Name of errors [Timing for checks]	Class	Description and causes	Measures, etc.	Error lamp	Stop / Run	Special internal output to be set	
							Bit	Word
11	System ROM error-1 [At power source ON]	Fatal failure	Abnormality was detected in sum check of the system ROM by the microcomputer. (System ROM could not be read correctly)	Hardware failure in CPU. When the error is not recovered by turning on power supply again, replace CPU. [Other causes and check points] •Is there any equipment generating electronic noises nearby ?	On	Stopped	—	—
12	System RAM error-1 [At power source ON]	Fatal failure	Abnormality was detected in read/write check of the system RAM by the microcomputer. (System RAM could not be read/write correctly)		On	Stopped	—	
13	Microcomputer error [Continuously]	Fatal failure	Abnormality was detected because the microcomputer attempted execute an undefined command during execution of the system program. [System ROM could not be read correctly)		On	Stopped	R7C8	—
23	Undefined command [Continuously]	Medium failure	Abnormality was detected because the microcomputer attempted execute an undefined command which could not be read by the processor for user program execution. (User program or system ROM could not be read correctly)		On	Stopped	R7C9	—
27	Data memory failure [At power source ON. On an initialization through a programming equipment]	Medium failure	Abnormality was detected in read/write check of data memory. (Data memory could not be read/write correctly)		On	Stopped	—	—
2C	Sequence processor failure [Continuously]	Medium failure	Abnormality was detected because the processor for user program execution dose not function in accordance with the program by the microcomputer. (The processor for user program execution could not be read correctly)	Hardware failure in CPU. When the error is not recovered by turning on power supply again, replace CPU. [Other causes and check points] •Is the cable connected independently ? • Is there any equipment generating ?	On	Stopped	R7C9	—

Error code	Name of errors [Timing for checks]	Class	Description and causes	Measures, etc.	Error lamp	Stop / Run	Special internal output to be set	
							Bit	Word
31	User's memory failure [At power source ON On RUN from STOP During RUN(for RAM only) On a change in parameters On an execution of the memory pack transfer function]	Medium failure	In the case of RAM memory . Abnormality was detected in sum check of the user memory. (The contents of user memory have not been correct.) In the case of ROM memory. Abnormality was detected in sum check of the user memory in ROM. (The contents of ROM memory have not been correct.)	• The contents of user program are destroyed. Perform initialization and transfer program again. • This error is indicated if the battery is completely discharged or the battery is remove for a long time. • When the ROM is used, this error may be indicated in the cases below. 1) ROM chip is mounted reversely. 2) ROM memory pack is not plugged in correctly. 3) Program is not written correctly in ROM.	On	Stopped	R7CA	—
33	User's memory size error [On RUN from STOP On an execution of the memory pack transfer function]	Medium failure	A capacity of user's program specified by the parameters exceeds the user's memory that is actually provided.	This error might be displayed when the contents of memory in the CPU module are not correct. When the error is not recovered by turning on power supply again, replace CPU.	On	Stopped	R7CC	—
34 *1	Syntax / Assemble error [On RUN from STOP]	Medium failure	A syntax eiror exists in the user's program (Ladder diagram). Details of the error will be set at the special internal output WRFOO1.	A syntax error or an assemble error exists in the user's program. Check the program and the assignment table again for correctness.	On	Stopped	R7D4	WRF 001
35	Mismatching in a comparison [On an execution of the memory pack transfer function]	Medium failure	In the memory pack transfer function, as the result of program comparison made between EEPROM and CPU: (a) Sum error in the memory pack. (b) Sum error in the CPU. (c) Mismatching exists in LADDER program.	—	Flick-ering	Stopped	—	—

*1 Includes the cases where the program tries to use the commands not supported by H-250/252 or unusable I/O area.

(Note) When MPH-1GE or MPH-16R assigned in 16k steps mount to H-250, the below errors occur.

- MPH-1GE in transfer mode → "33"
- MPH-16R or MPH-16E in soft - protcct
  - Program capacity < 8k steps → "5A" (program can be transferred)
  - Program capacity ≥ 8k steps → "33" (program can not be transferred)
- MPH-1GE in matching mode
  - Program capacity ≥ 8k steps → "35"

Error code	Name of errors [Timing for checks]	Class	Description and causes	Measures, etc.	Error lamp	Stop / Run	Special internal output to be set	
							Bit	Word
36	Mismatching in memory assignment [On an execution of the memory pack transfer function]	Medium failure	In the memory pack transfer function, as the result of program comparison made between EEPROM and CPU: Mismatching exists in memory assignment	—	Off	Stopped	—	—
44	Congestion error (Normal scan) [On an execution of END process]	Minor failure	Execution time (scan time) of normal scan was checked of each END command. In consequence, the parameter-set congestion check time (standard 100 msec) was exceeded. So it was detected as an abnormality.	Change the program so as shorten the scan time of the user's program, or change the congestion check time setting. (Note): parameter settings for CPU operation can be changed by peripherals.	On	(Stop) *1	R7D1	—
45	Congestion error (Periodic scan) [During the periodic process]	Minor failure	Execution time of periodic scan program was checked in periodic processing. As a result, the determined time was exceeded. So it was detected as an abnormality.	Change the program so as shorten the execution time of the periodic inteiruption scan.	On	(Stop) *1	R7D2	—
47 *2	Excessive number of points for I/O assignment [At power source ON On RUN from STOP During RUN(for RAM only) On a change in parameters]	Minor failure	When the sophisticated base is used, number of assigned slots is more than 29. When the standard base is used, number of assigned slots is more than 16.	Make I/O assignment so as to avoid the number of inputs over the maximum permissible value in the CPU unit.	On	Stopped	R7D6	—
54 *2	Communication module failure [Continuously]	Alarm	Hardware failure in the communication module. The unit No. and the slot No. of the modules under failure will be set on WRF004.	Provide appropriate recovery measures on the communication module in the error in refereace to error codes. Replace the failed communication module.	Off	Running	R7D7	WRF004
59 *2	Failure in the link module [Continuously]	Alarm	Hardware failure in the link module. Abnormal link parameters. The slot No. where the abnormality exists will be set on WRF007.	Provide appropriate recovery measures on the communication module in the error in reference to error codes. (For remedies from the errors, refer to the manual for the link module.)	Off	Running	R7DE	WRF007 (slot No.) WRF0E0 to F19F (details)

*1 Running can be continued even in the failure by setting the parameters by a programming equipment.

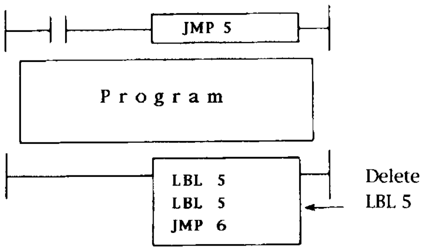
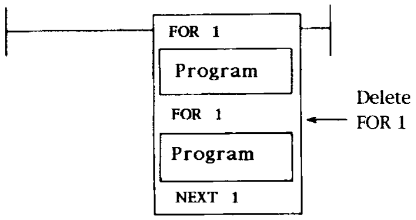
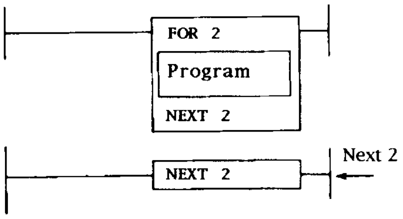
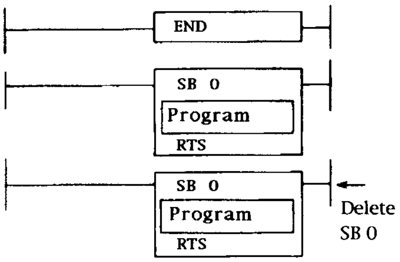
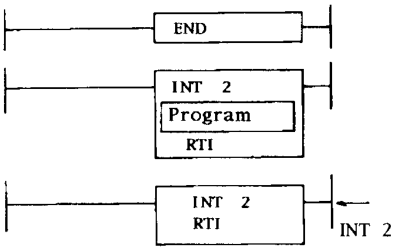
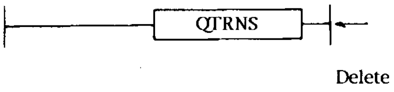
*2 Effective only H-252.

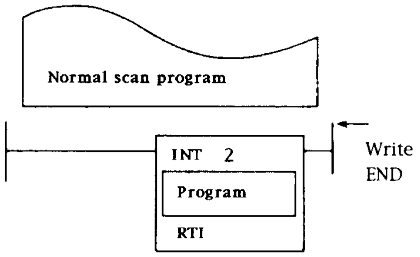
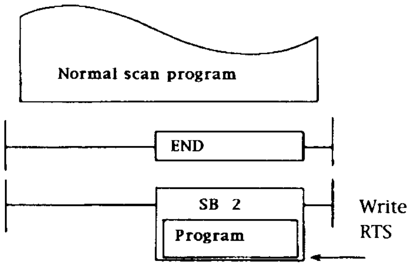
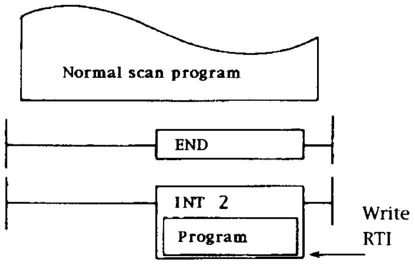
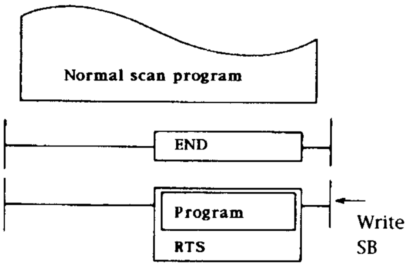
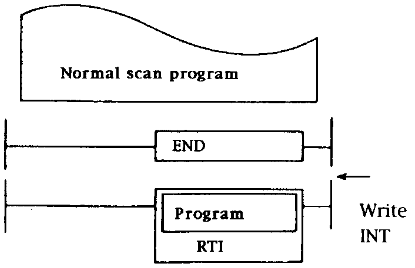
Error code	Name of errors [Timing for checks]	Class	Description and causes	Measures, etc.	Error lamp	Stop / Run	Special internal output to be set	
							Bit	Word
61	Communication port error (Parity) *1 [During communications]	Alarm	In communication with programming device or personal computer, a parity error of received data was detected. Cable was disconnected in the on-line mode. (Noise is superimposed on cable. Connector cable is not connected properly. Mismatch in transmission speed, etc.)	• Plug in the cable connector again. • Re-confirm wiring and signal allocation inside the connector. • Review setting of transmission speed, etc. • Eliminate noise source from the vicinity of cable • Other proper measure. • Error code 64 might be set in WRF000 by generating "Port transmission error (protocol error)" while monitoring with LADDER EDITOR (edition of PC98). However, the error lamp does not light and the monitor continues. Please disregard or clear this error by turning on special internal output R7EC of an error code clear.	Off	Running	—	—
62	Communication port error (Framing / Overrun) *1 [During communications]	Alarm	In communication with programming device or personal computer, framing error or overrun error was detected. (Noise is superimposed on cable. Connector cable is not connected properly. Mismatch in transmission speed, etc.		Off	Running	—	—
63	Communication port error (Time-out) *1 [During communications]	Alarm	In communication with programming device or personal computer, time-out error was detected. (Noise is superimposed on cable. Connector cable is not connected properly. Processing program on the personal computer side does not agree with CPU specifications.)		Off	Running	—	—
64	Communication port error (Protocol error) *1 [During communications]	Alarm	In communication with programming device or personal computer, protocol (transmission procedure) error was detected. (Noise is superimposed on cable. Connector cable is not connected properly. Processing program on the personal computer side does not comply with protocol.)		Off	Running	—	—
65	Communication port error (BCC error) *1 [During communications]	Alarm	In communication with programming device or personal computer, data could not be received correctly. (Noise is superimposed on cable. Connector cable is not connected properly. Processing program on the personal computer side does not comply with protocol.		Off	Running	—	—
71	Battery error [Continuously]	Alarm	Battery charge has become lower than the specified level. Or the battery is not mounted.	Renew the battery	Flickering	Running	R7D9	—

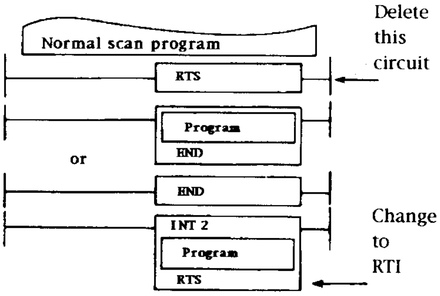
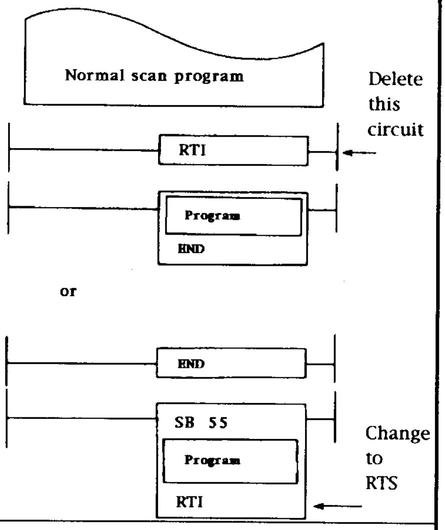
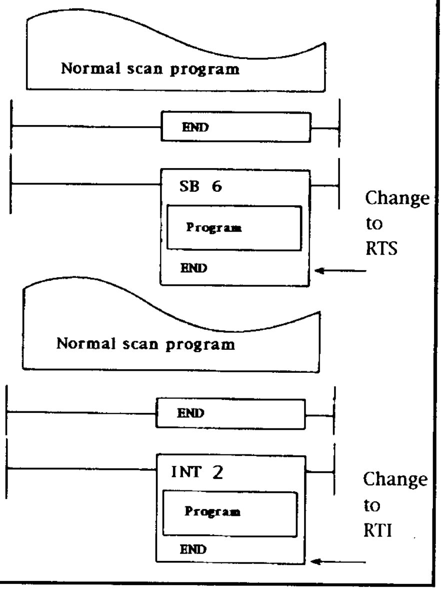
* Errors of code 61 to 65 may also occur the moment the cable is disconnected or reconnected in the live status. This is due to communication in the unstable condition peculiar to disconnection / reconnection time. Since recovery will automatically be made soon, CPU operation remains unaffected. For clearing the error code in CPU module, set the special internal output R7EB to 1.

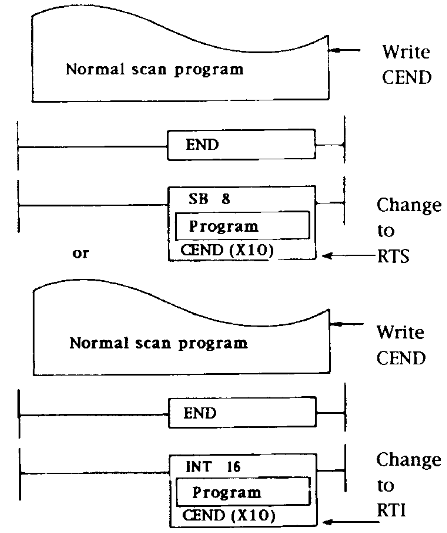
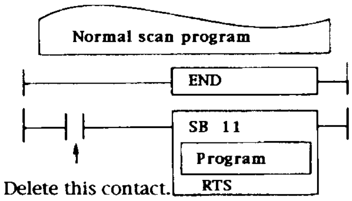
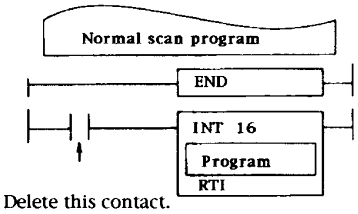
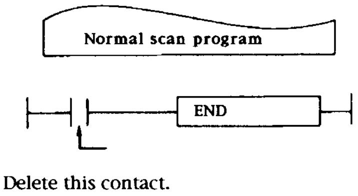
Appendix 2 Details of Syntax • Assembling Errors

(1) The followings are syntax • assembling error codes and corresponding circuit status. Error codes will be set in hexadecimal to WRF001 of the internal output.

Code (Hexa-decimal)	Item	Error description	Remedy	Example program
H0001	Double definition of LBL	Two or more LBL commands exist with the same number.	Avoid the double numbering. Give single number to single LBL command.	
H0002	Double definition of FOR	Two or more FOR commands exist with the same number.	Avoid the double numbering. Give single number to single FOR command.	
H0003	Double definition of NEXT	Two or more NEXT commands exist with the same number.	Avoid the double numbering. Give single number to single NEXT command.	
H0004	Double definition of SB	Two or more SB commands exist with the same number.	Avoid the double numbering. Give single number to single SB command.	
H0005	Double definition of INT	Two or more INT commands exist with the same number.	Avoid the double numbering. Give single number to single INT command.	
H000F	Undefined command error	Tried to use a command that is unusable in H-250/252, or found an invalid command.	Delete the command identified as undefined.	

Code (Hexa-decimal)	Item	Error description	Remedy	Example program
H0010	No definition for END	No END command exists before the commands INT and SB.	Define the command END before the command INT or SB.	
H0011	NO definition for RTS	No RTS command to be in response to the command SB.	Define the command RTS after the command SB.	
H0012	No definition for RTI	No RTI command to be in response to the command INT.	Define the command RTI after the command INT.	
H0013	No definition for SB	No SB command to be in response to the command RTS.	Define the command SB before the command RTS.	
H0014	No definition for INT	No INT command to be in response to the command RTI.	Define the command INT before the command RTI.	
H0016	I/O No. error	Exists a circuit containing I/O No. that is out of the applicable range of I/O of H-250/252.	Change or delete No. that is out of the applicable range	

Code (Hexa-decimal)	Item	Error description	Remedy	Example program
H0020	RTS in the error area	The command RTS exists in a normal scan area or in an interruption scan area.	Define RTS in a subroutine area.	
H0021	RTI in the error area	The command RTI exists in a normal scan area or in a subroutine area.	Define RTI in an interruption area.	
H0022	END in the error area	The command END exists in an interruption area or in a subroutine area.	Define END at the end of a normal scan area.	

Code (Hexa-decimal)	Item	Error description	Remedy	Example program
H0023	CEND in the error area.	The command CEND exists in an interruption scan area or in a subroutine area.	Define the command CEND in a normal scan area.	
H0030	A start condition error of RTS.	A condition for the start exists in an operational box that contains the command RTS.	Delete the start condition in the operational box.	
H0031	A start condition error of RTI.	A condition for the start exists in an operational box that contains the command RTS.	Delete the start condition in the operational box.	
H0032	A start condition error of END.	A condition for the start exists in an operational box that contains the command RTS.	Delete the start condition in the operational box.	

Appendix 3 Computational error codes

(1) When an error occurs in an execution of a control instraction, "1" will be set to the internal special output (R7F3) for the computation error (ERR), and an error code that indicates the error description will be set to WRF015.

(2) To clear to zero for the computation error, execute "R7F3 = 0" which realizes a forced set through a peripheral, or write program preliminary so that R7F3 is reset by external input. To clear the error code, execute "WRF015 = 0" which realizes a forced set through a peripheral, or write program preliminary so that R7F3 is reset by external input.

Error code (Hexa-decimal)	Name of the error	Description	Command to call error
H0013	SB not defined	The command SBn is not given in the program for the same code number "n" of the command CALn.	CAL
H0015	LBL not defined	The command LBLn is not given in the program for the same code number "n".	JMP CJMP
H0016	FOR not defined	The command FORn is not given in the program for the same code number "n" of the command NEXTn.	NEXT
H0017	NEXT not defined	The command NEXTn is not given in the program for the same code number "n" of the command FORn.	FOR
H0040	LBL in the error area	The command LBL is not given in the program for the same code number "n" of the command JMP (CJMP) in the same area.	JMP CJMP
H0041	CAL nesting overflow	Number of nesting layers exceeds six in a subroutine.	CAL
H0042	CAL not defined	RTS is executed even when the command CAL is not executed.	RTS
H0043	NEXT - FOR error	The command NEXTn exists before the command FORn that has the same code number "n".	FOR
H0044	NEXT in the error area	The command NEXTn is not given for the same code number "n" of the command FORn in the same area.	FOR
H0045	FOR - NEXT nesting overflow	FOR n and NEXT n are not properly nested.	FOR
H0046	FOR nesting overflow	The nesting FOR - NEXT are in six layer or more.	FOR NEXT

Appendix 4 Special Output

(1) List of Special Internal Output

Special Internal Bit Output (64 points)

I/O No.	Contents	ON	OFF
R7C0	Continue processing during cycle time over (normal scan)	U	U
R7C1	Continue processing during cycle time over (periodic scan)		
R7C2	(Undefined)	-	-
R7C3	Remote Run enabled	U	U
R7C4	Remote Stop enabled		
R7C5	Debug enabled		
R7C6	(Undefined)	-	-
R7C7	Enabled modification during RUN	U	U
R7C8	Serious failure	S	
* R7C9	Sequence processor error		
R7CA	User memory error		
R7CB	(Undefined)	-	-
R7CC	Memory size over	S	U
R7CD	(Undefined)	-	-
R7CE	(Undefined)		
R7CF	(Undefined)		
R7D0	(Undefined)		
R7D1	Congestion error (normal scan)	S	U
R7D2	Congestion error (periodic scan)		
R7D3	(Undefined)	-	-
R7D4	Syntax or assemble error	S	U
R7D5	(Undefined)	-	-
* R7D6	Excessive number of assignment points in I/O module	S	U
* R7D7	Communication module error		
R7D8	(Undefined)	-	-
R7D9	Battery error	S	S
R7DA	(Undefined)	-	-
R7DB	Self-diagnostic error	S	U
R7DC	(Undefined)	-	-
R7DD	(Undefined)		
* R7DE	Link module failure	S	U
R7DF	(Undefined)	-	-
R7E0	Mode key switch in STOP position	S	U
R7E1	Dip switch in the position "4"		
R7E2	Mode key switch in RUN position		
R7E3	Single scan ON after RUN		
R7E4	Always ON		X
R7E5	0.02 - s clock		S
R7E6	0.1 - s clock		
R7E7	1 - s clock		
R7E8	Occupancy flag		
R7E9	Disabled RUN		
R7EA	Modification during RUN		
R7EB	Error LED display clear	U	
R7EC	Internal special erroneous output clear		
R7ED	(Undefined)	-	-
R7EE	(Undefined)		
R7EF	(Undefined)		
R7F0	Carry flag (CY)	S	S
R7F1	Overflow (V)		
R7F2	Shift data (SD)	U	U
R7F3	Calculation error (ERR)	S	S
R7F4	Data error (DER)		
R7F5	(Undefined)	-	-
R7F6	(Undefined)		
R7F7	(Undefined)		
R7F8	Request to read calendar clock	U	S
R7F9	Request to set calendar clock		
R7FA	Request to adjust calendar clock		
R7FB	Incorrect calendar clock setting		
* R7FC	Trigger condition matching flag	S	
* R7FD	Trace monitor flag		
R7FE	(Undefined)	-	-
* R7FF	Converted I/O number in the case of bit I/O No. error		

The user must operate U.
S is controlled by systems. The user must not control S and X.
* : Effective only for H-252.

Special Internal Word Output (512 words)

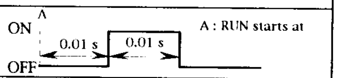
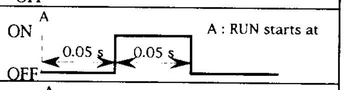
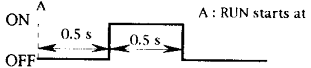
I/O No.	Contents	ON	OFF
WRF000	Self-diagnostic error code	S	U
WRF001	Syntax and assembler error details		
WRF002	(Undefined)	-	-
WRF003	(Undefined)		
WRF004	Invalid communication module slot number	S	U
WRF005	(Undefined)	-	-
WRF006	(Undefined)		
WRF007	Invalid link module slot number	S	U
WRF008	Error circuit number		
WRF009	(Undefined)	-	-
WRF00A	(Undefined)		
WRF00B	Calendar cock reading data (year)	S	S
WRF00C	Calendar clock reading data (month, day)		
WRF00D	Calendar clock reading data (a day of the week)		
WRF00E	Calendar clock reading data (hour, minute)		
WRF00F	Calendar clock reading data (second)		
WRF010	Maximum scan time		
WRF011	Current scan time		
WRF012	Minimum scan time		
WRF013	CPU status		
WRF014	Internal word output capacity		
WRF015	Calculation error code		U
WRF016	Residue register (lower order)		S
WRF017	Residue register (upper order)		
* WRF018	Converted I/O number in the case of double word I/O No. error		
* WRF019	Converted I/O number in the case of word I/O No. error		
WRF01A	(Undefined)	-	-
WRF01B	Calendar clock reading or setting data (year)	U/S	S
WRF01C	Calendar clock reading or setting data (month, day)		
WRF01D	Calendar clock reading or setting data (a day of the week)		
WRF01E	Calendar clock reading or setting data (hour, minute)		
WRF01F	Calendar clock reading or setting data (second)		
WRF020 WRF021	Status, communication module slot 0 status	S	
WRF022 to WRF031	Status, communication module slot 1 to 8 status		
WRF032 WRF033	Status, communication module slot 9 status		
WRF034 to WRF03F	(Undefined)	-	-
WRF040 WRF041 WRF042	Member registration area 1	S	S
WRF043 WRF044 WRF045	Member registration area 2		
WRF046 WRF047 WRF048	Member registration area 3		
WRF049 WRF04A WRF04B	Member registration area 4		
WRF04C WRF04D WRF04E	Debug registration area		
WRF04F to WRF0DF	(Undefined)	-	-
WRF0E0 to WRF13F	Link - 1 error flag	S	S
WRF140 to WRF19F	Link - 2 error flag		
WRF1A0 to WRF1FF	(Undefined)	-	-

* and WRF020 to WRF1FF are effective only for H-252.

(2) Details of Special Internal Bit Output

Number	Name	Interpretation	Detailed descriptions	Conditions of setting	Conditions of resetting
R7C0	Continue processing during cycle time over (normal scan)	0 : Discontinue RUN 1 : Continue RUN	Specifies to continue or discontinue RUN in the case of a cycle time over error in a normal scan.	The user must operate "ON"	The user must operate "OFF". "OFF" can also be made by the set switch(R. CL) for the power failure memory area.
R7C1	Continue processing during cycle time over (periodic scan)	0 : Discontinue RUN 1 : Continue RUN	Specifies to continue or discontinue RUN in the case of a cycle time over error in a periodic scan.		
R7C2	(Undefined)	-	-		
R7C3	Remote Run enabled	0 : Prohibit REMOTE RUN 1 : Permit REMOTE RUN	Specifies to permit or prohibit of REMOTE RUN (RUN command from a host computer).		
R7C4	Remote Stop enabled	0 : Prohibit REMOTESTOP 1 : Permit REMOTE STOP	Specifies to permit or prohibit of REMOTE STOP (STOP command from a host computer).		
R7C5	Debug enabled	0 : Prohibit to debug 1 : Permit to debug	Specifies to permit or prohibit of a debug RUN.		
R7C6	(Undefined)	-	-		
R7C7	Enabled modification during RUN	0 : Prohibit to change 1 : Permit to change	Specifies to permit or prohibit of changes of user's program during RUN.		
R7C8	Serious failure	0 : No serious failure 1 : Exists serious failure	Indicates the manager microcomputer is in a failure or not.	The system will operate "ON"	
R7C9 *2	Sequence processor error	0 : Normal 1 : Abnormal	Indicates the sequence processor (the microcomputer that realize the sequence operations) is in a failure or not.		
R7CA	User memory error	0 : Normal 1 : Abnormal	Indicates the user's memory is in a failure or not.		
R7CB	(Undefined)	-	-		
R7CC	Memory size over	0 : Normal 1 : Abnormal	Indicates the user's program size exceeds or not the user's program memory size assigned by the parameters.		
R7CD	(Undefined)	-	-		
R7CE	(Undefined)	-	-		
R7CF	(Undefined)	-	-		
R7D0	(Undefined)	-	-		
R7D1	Congestion error (normal scan)	0 : Normal 1 : Congestion error	Execution time exceeds or not the time specified by the parameters.		
R7D2	Congestion error (periodic scan)	0 : Normal 1 : Congestion error	Periodic scan is completed or not within that period of time.		
R7D3	(Undefined)	-	-		
R7D4	Syntax or assemble error	0 : Normal 1 : Error	User's program contains or not a syntax error (Details will be outputted to WRF001)		
R7D5	(Undefined)	-	-		
R7D6 *2	Excessive number of assignment points in I/O module	0 : Normal 1 : Excessive	Assignment in I/O exceeds or not the maximum permissible value.		
R7D7	Communication module error	0 : Normal 1 : Failure	Communication module is in the normal or in a failure (Failed slot No. will be outputted to WRF004).		
R7D8	(Undefined)	-	-		
R7D9	Battery error *1	0 : Normal 1 : Failure	Battery voltage is normal or dropped in the CPU module.		The system will operate "OFF"
R7DA	(Undefined)	-	-		The user must operate "OFF". "OFF" can also be made by the set switch(R. CL) for the power failure memory area.
R7DB	Self-diagnostic error	0 : Normal 1 : Error	Self diagnostic system detected or not an error (Details will be outputted to WRF000).		
R7DC	(Undefined)	-	-		
R7DD	(Undefined)	-	-		
R7DE *2	Link module error	0 : Normal 1 : Error	Link module is in the normal or in a failure (Failed slot No. will be outputted to WRF007) (Details will be outputted to WRF0E0 to WRF19F).		
R7DF	(Undefined)	-	-		

*1 Battery error (R7D9) will be returned to "OFF" automatically on a replacement of the battery.
*2 Effective for H-252 only.

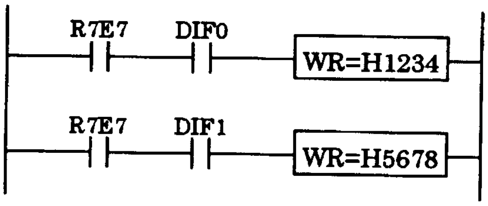
Number	Name	Interpretation	Detailed descriptions	Conditions of setting	Conditions of resetting
R7E0	Mode key switch in STOP position	0 : Key position, not in STOP 1 : Key position, in STOP	Any one of "1" will be ON. (R7E0 or R7E2)	The system will operate "ON"	The system will operate "OFF".
R7E1	Dip switch in the position "4"	0 : Switch position, not in REMOTE 1 : Key position, in REMOTE			
R7E2	Mode key switch in RUN position	0 : Key position, not in RUN 1: Key position, in RUN			
R7E3	Single scan ON after RUN	0 : 2nd scan or later after RUN 1: 1st scan after RUN			
R7E4	Always ON	0 : Not assigned 1 : Continuation			"ON" only Avoid "OFF"
R7E5	0.02 - s clock *2	0 : 0.01 seconds 1 : 0.01 seconds			The system will operate "OFF".
R7E6	0.1 - s clock *2	0 : 0.05 seconds 1 : 0.05 seconds			
R7E7	1 - s clock *2	0 : 0.5 second 1 : 0.5 second			
R7E8	Occupancy flag	0 : Not occupied 1 : Occupied	Indicates the status of occupation in the case of communications with a personal computer or a peripheral.		
R7E9	Disabled RUN	0 : Permit RUN 1 : Prohibit RUN	Prohibit or permit RUN.		
R7EA	Modification during RUN	0 : Modification unexecuted during RUN 1 : Modification under execution during RUN	RUN is halted (outputs in the hold) or not caused by modifications made during RUN.		
R7EB *1	Error LED clear	1: Clear	To clear the error indication in the LED.	The user must operate "ON"	
R7EC *1	Internal special erroneous output clear	1: Clear	To clear special internal outputs that are set for indications of errors in the error LED (To clear WRF000 ~ WRF008, R7C8 ~ R7DE).		

*1 If "1" is set contrary the system turns it to "OFF" automatically.

*2 Clock (R7E5 - R7E7) of a special internal output is changed synchronously with the execution of the user program (scan). Therefore, please replace the clock with an internal output at the head of the program and use the clock when you do the programming on the assumption that the clock does not change for one scan.

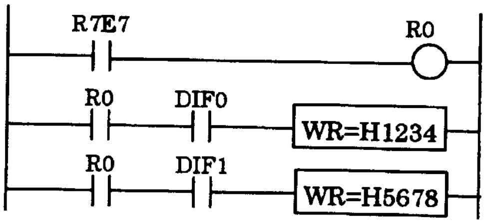
- **Program example when clock can change for one scan**

(Clock at one second)



- **The guarantee to which the clock does not change for one scan is a program example if it is necessary.**

(Clock at one second)



Number	Name	Interpretation	Detailed descriptions	Conditions of setting	Conditions of resetting
R7ED	(Undefined)	-	-	-	-
R7EE	(Undefined)	-	-		
R7EF	(Undefined)	-	-		
R7F0	Carry flag (CY)	0 : Carry OFF 1 : Carry ON	Carry flag used in the arithmetic commands.	The system will operate "ON"	The system will operate "OFF"
R7F1	Overflow (V)	0 : No overflow 1 : Overflow exists	Overflow flag in the arithmetic commands.		
R7F2	Shift data (SD)	0 : Shift data "0" 1 : Shift data "1"	Shift data as used in Shift command and other commands.	The user must operate "ON"	The user must operate "OFF"
R7F3	Calculation error (ERR)	0 : Normal 1 : Error	An erroneous command exists or not in the processing (Details will be outputted to WRF015).	The system will operate "ON"	The system will operate "OFF"
R7F4	Data error (DER)	0 : Normal 1 : Congestion Error	Data error is detected or not in the processing.		
R7F5	(Undefined)	-	-	The user must operate "ON"	The system will operate "OFF"
R7F6	(Undefined)	-	-		
R7F7	(Undefined)	-	-		
R7F8 *1	Request to read calendar clock	1 : Read	To read the calendar clock data, year, month, day, days of a week, hour, minute, second, and set them to WRF01B ~ WRF01F.		
R7F9 *1	Request to set calendar clock	1 : Set	To set the data outputted to WRF01B ~ WRF01F on the calendar clock.		
R7FA *1	Request to adjust calendar clock	1 : Adjust	Returned to "0" when second data is 0 ~ 29, and to "0 + 1 minute" when second data is 30 ~ 59.		
R7FB	Incorrect calendar clock setting	0 : Normal 1 : Error	An error in the calendar clock setting.	The system will operate "ON"	
R7FC	Trigger condition matching flag	0 : Trigger mismatches 1: Trigger matches	"1" will be set when a trigger condition is matched.		
R7FD	Trace monitor flag	0 : Stop 1: Under execution	"1" will be set when the trace monitor is executed.		
R7FE	(Undefined)	-	-	-	-
R7FF	Converted in bit I/O No. error		CPU will covert to R7FF when bit I/O No. in the program is out of usable range. *2	-	-

*1 If "1" is set contrary the system turns it to "OFF" automatically.

*2 Only for H-252 (In case of H-252, CPU will convert to L13FFF)

(3) Details of Special Internal Word Output

Number	Name	Stored data	Description
WRF000	Self-diagnostic error code	Self-diagnostic error codes	The error numbers detected by CPU are stored by the binary code.
WRF001 (R7D4)	Syntax and assembler error details	Syntax and assembler error codes	Syntax and assembler error codes of a user's program are stored by the binary code.
WRF002	(Undefined)	-	-
WRF003	(Undefined)	-	-
WRF004 (R7D7)	Invalid communication module slot number	Identification number of the slots in error *1	To store the slot identification number of the communication module that is in the error (The last 4 bits : 1 to 8).
WRF005	(Undefined)	-	-
WRF006	(Undefined)	-	-
WRF007 (R7DE)	Invalid link module slot number	Identification number of the slots in error *1	To store the slot identification number of the link module that is in the error 15 12 11 8 7 4 3 0 "0" a b "0" a : Unit No. (0 to 2) b : Slot No. (0 to 9)
WRF008	Error circuit number	No. of the circuit having an error	Stores circuit No. that are in undefined command, I/O No. error, or syntax assemble error
WRF009	(Undefined)	-	-
WRF00A	(Undefined)	-	-
WRF00B	Calendar clock reading data (BCD data)	Year	A.D. in 4 digits is always given. (Example) 1991 : Year 1991
WRF00C		Month, Day	Data for Month and Day are always given. (Example) 0921 : September 21
WRF00D		A day of the week	Data for a day of the week is always given. (Example) 0006 : Saturday *2
WRF00E		Hour, Minute	Hours and Minutes (24 hours system) are always given. (Example) 0805 : 8 hours past 5 minutes
WRF00F		Second	Data for Seconds is always given. (Example) 0015 : 15 seconds
WRF010	Maximum scan time	Maximum execution time of the normal scan	10ms units
WRF011	Current scan time	Current execution time of the normal scan	10ms units
WRF012	Minimum scan time	Minimum execution time of the normal scan	10ms units Minimum execution time is set to 65535 immediately after the initiation of RUN, and after to proper value.
WRF013	CPU status	Status of CPU operation 15 10 9 8 7 6 5 4 3 2 1 0 Not used a b c d e f g h i a : CPU type : 01 = H-2000, 10 = H-700/ 252 11 = H-300/ 250 b : Battery error : 1 = Error, 0 = Normal c : Not used d : Debug : 1 = Under execution, 0 = No execution e : Force : 1 = Under execution, 0 = No execution *3 f : Error : 1 = Error, 0 = Normal g : Simulation : 1 = Under execution, 0 = No execution *3 h : Halt : 1 = Under execution, 0 = No execution i : Operation : 1 = RUN, 0 = STOP	
WRF014	Internal word output capacity	Number of words for the internal word outputs	H-250 : 0400, H-252 : 4400
WRF015	Calculation error code	Calculation error codes	To store the error codes during the execution of an operation command.
WRF016	Residue register (lower order)	Division overflow (Remainder of the division operation)	32-bit operation : Upper order to F017, Lower order to F016
WRF017	Residue register (upper order)		16-bit operation : F016 is used
WRF018	Converted I/O number in the case of double word I/ O No. error	CPU converts to DRF018 when double word I/O number in the program is out of usable range. *4	
WRF019	Converted I/O number in the case of word I/O No. error	CPU converts to WRF019 when word I/O number in the program is out of usable range. *4	
WRF01A	(Undefined)	-	-

*1 To know which slot is in the error, make the corresponding special internal bit output (shown in parentheses in the Number column) OFF beforehand, or make R7EC ON beforehand.

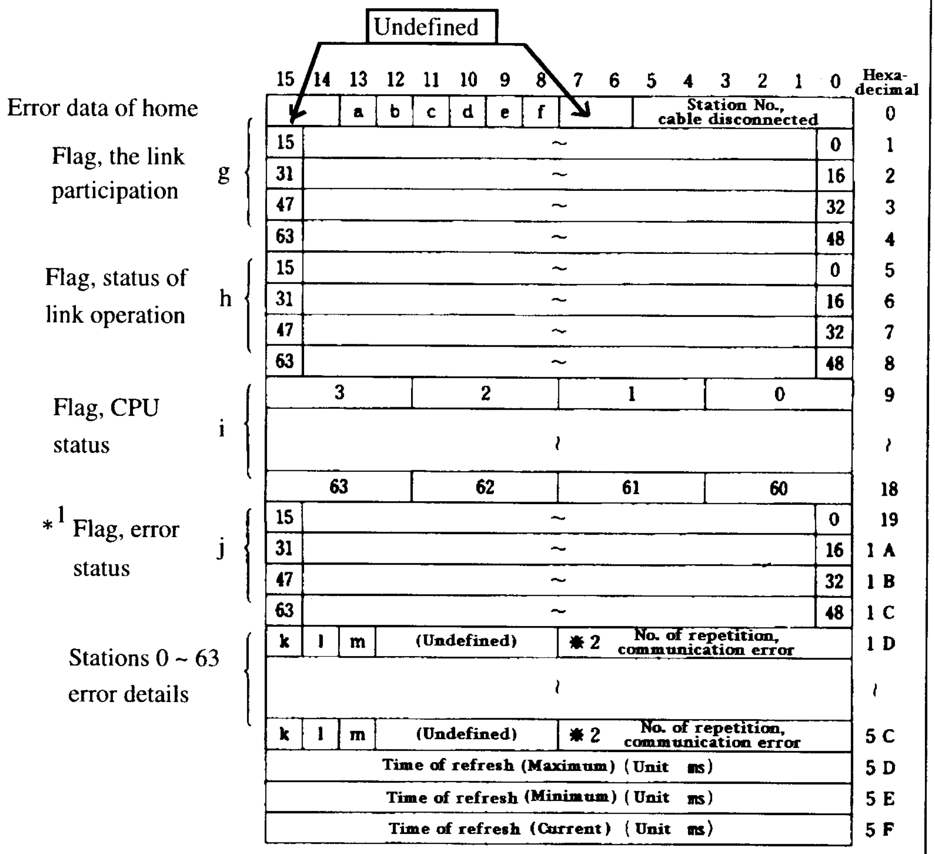
*2 0 : Sunday, 1 : Monday, 2 : Tuesday, 3 : Wednesday, 4 : Thursday, 5 : Friday, 6 : Saturday

*3 Not in use for H-250/252.

*4 Only for H-252 (In case of H-252, CPU will convert to DL13FE or WL13FF).

Number	Name	Stored data	Description
WRF01B	Calendar clock reading or setting	Year	A.D. reading or setting. (Example) $\boxed{1991}$: Year 1991
WRF01C		Month, Day	Month and Day, reading or setting. (Example) $\boxed{0921}$: September 21
WRF01D		A day of the week	A day of the week, reading or setting. (Example) $\boxed{0006}$: Saturday *2
WRF01E		Hour, minute	Hours and Minutes (24 hours system), reading or setting. (Example) $\boxed{0805}$: 8 hours past 5 minutes
WRF01F		Second	Seconds, reading or setting. (Example) $\boxed{0015}$: 15 seconds
WRF020 WRF021	Status, communication module slot 0 status	Status data (Slot 0)	15 8 7 0 Status 1 Status 2 Status 3 Status 4 Status description will not be given for the slot that the communication module is not mounted on. For details refer to the user's manual for the communication module.
WRF022 to WRF031	Status, communication module slot 1 to 8 status	Status data (Slot 1 to 8)	
WRF032 WRF033	Status, communication module slot 9 status	Status data (Slot 9)	
WRF034 to WRF03F	(Undefined)	-	-
WRF040 WRF041 WRF042	Member registration area 1	Port number occupied CPU	15 8 7 0 a "0" b c d e a : 0 = NOT occupied, 1 = READ occupation 2 = WRITE occupation b : Loop No. c : Unit No. d : Module No. E : Port No.
WRF043 WRF044 WRF045	Member registration area 2	Port number occupied CPU	
WRF046 WRF047 WRF048	Member registration area 3	Port number occupied CPU	
WRF049 WRF04A WRF04B	Member registration area 4	Port number occupied CPU	
WRF04C WRF04D WRF04E	Debug registration area	Port number under debug	15 8 7 0 Under debug "0" Loop No. Unit No. Module No. Port No.
WRF04F to WRF0DF	(Undefined)	-	-

* WRF020 to WRF1FF sre effective only for H-252.

Number	Name	Stored data	Description
WRF0E0 to WRF13F	Link 1, error flag		
WRF140 to WRF19F	Link 2, error flag	a : System bus error b : (Undefined) c : Area assigned to outside the specified d : Double assignment of the area e : Station No. assigned to outside the specified f : Communication route failure 1 = Error 0 = No error g : Numbers in the table are the station No., 1 = Participate, 0 = No participate h : Numbers in the table are the station No., 1 = Under link operation, 0 = Link quitted i : Numbers in the table are the station No., Details for 4 bits below; U V W X 4 bits U : 1 = CPU error, 0 = No error V : 1 = In simulation, 0 = Not in simulation W : 1 = In the halt, 0 = Not in the halt X : 1 = RUN, 0 = STOP j : Numbers in the table are the station No. 1 = Error, 0 = No error k : Time-out error, 1 = Error, 0 = No error l : Frame error, 1 = Error, 0 = No error m : Abnormal between CPU and link, 1 = Abnormal, 0 = Normal	
WRF1A0 to WRF1FF	(Undefined)		

*1 " 1 " will be set to "Flag, error status" in case one of the errors k, l, m as reserved in " Error details" is occurred. Error data will be given to k, l, m when the errors are occurred during the communication with other CPU by means of a peripheral.

*2 "Number of repetition, communication error" is the total sum of occurrence of K or l.

*3 WRF020 to WRF19F are effective only for H-252.