e-Front runners

MICREX-SX series SPH

USER'S MANUAL

Instructions

(For "SX-Programmer Standard Type: NP4H-SWN")

FEH588d

Preface

Thank you for purchasing Fuji Electric Programmable Controller MICREX-SX Series.
This User's Manual explains the system configuration, the memory and the language of SPH. Read this manual carefully to ensure correct operation.
When using modules or peripheral devices, be sure to read the corresponding user's manual listed below.

Title	Manual No.	Contents
User's Manual Hardware, MICREX-SX series SPH	FEH201	Explains the system configuration, the specifications and operations of modules in the MICREX-SX series.
User's Manual MICREX-SX series Expansion FBs for SX-Programmer Standard	FEH589	Explains the specifications and operations (installation procedures, etc.) of the expansion FBs for SX-Programmer Standard.
User's Manual MICREX-SX series SX-Programmer Standard	FEH590	Explains the menus, icons and operations of SX-Programmer Standard.

* In addition to the above manuals, the following Fuji Electric FA Components & Systems Co., Ltd. site offers various manuals and technical documents associated with MICREX-SX.
URL http://www.fujielectric.co.jp/fcs/eng/

Safety Precautions

Be sure to read the “Safety Precautions” thoroughly before using the module.
Here, the safety precaution items are classified into “Warning” and “Caution.”

Warning : Incorrect handling of the device may result in death or serious injury.

Caution : Incorrect handling of the device may result in minor injury or physical damage.

Even some items indicated by “Caution” may result in a serious accident.
Both safety instruction categories provide important information. Be sure to strictly observe these instructions.

Warning
❑ Place the emergency stop circuit, interlock circuit or the like for safety outside the PC. A failure of PC might break or cause problems to the machine.

Caution
❑ Sufficiently make sure of safety before program change, forced output, starting, stopping or anything else during a run. The wrong operation might break or cause machine problems.

Revision

*The manual No. is printed at the bottom right of the cover of this manual.

Printed on	*Manual No.	Revision contents
Mar. 2004	FEH588	First edition
Jun. 2004	FEH588a	The expression of addresses and instructions was changed according to the upgrading of the SX-Programmer standard.
Aug. 2005	FEH588b	The specifications for SPH2000 NP1PM-48R were added.
Jan. 2006	FEH588c	The specifications for SPH2000 NP1PM-48E/256E were added.
Dec. 2006	FEH588d	◆ Instruction processing speed chart was added. ◆ The contents were reviewed.

Contents

Contents

Section 1 Overview of Programming

Section 1 Overview of Programming

1-1 Programming Procedures

The general programming procedures are given below.

1) Preparing a new project
Select the model of the CPU module to be used and open the new project.

⇨ For the operation procedures, see "User's Manual SX-Programmer Standard (FEH590)."

⇩

2) Setting system definitions
- Register the modules such as input/output and communication modules, used for the SPH system (configuration) to be configured.
- Make necessary settings for system operation such as the CPU operation definitions and SX bus takt time.
- Change settings such as data area size, if necessary.

* System definitions can also be set after programming.

⇩

3) Creating program
- For MICREX-SX, create programs for each of control operation units.
- User functions and user function blocks may also be created.

⇨ For instruction specifications and procedures for creating user functions and user function blocks, see "Section 3 Instructions" and "Appendix 4 Procedures for Creating User FBs."

⇩

4) Allocating program to task
Allocate the program to a task (default, fixed or event), which executes it, prepared in the CPU.
These are set when programming, in the case of the "package displays" mode

⇨ For task specifications, see "Section 2-4 Tasks."

⇩

5) Transferring and monitoring project
Transfer the created project to the CPU for monitoring and debugging.

⇨ For the operation procedures, refer to "User's Manual SX-Programmer Standard (FEH590)."

Section 2 Specifications

Section 2 Specifications

2-1 Performance Specifications

2-1-1 SPH300

(1) NP1PS-32/32R/74/74R/117/117R

Item			Specification			Remarks
CPU models			NP1PS-32/ NP1PS-32R	NP1PS-74/ NP1PS-74R	NP1PS-117/ NP1PS-117R	
Program memory capacity			32K steps	74K steps	117K steps	
Program steps in a program			4096 steps (CPU with software version earlier than 62) 8192 steps (CPU with software version 62 or later)			
I/O memories	X, WX, DX		8192 points (including remote I/O points) * When the I/O expansion function is used: Max. 4096 words (117/117R) Max. 1856 words (74/74R)			Fixed value
	Y, WY, DY					
Standard memories	M, WM, DM	High speed	2K words			Fixed value
		Standard	6K words	30K words	126K words	Default value (Note 1)
Retain memories	L, WL, DL		4K words	16K words	32K words	Default value (Note 1)
Memories for user FBs	F, WF, DF		4K words	16K words	32K words	Default value (Note 1)
Memories for system FBs	Edge detection		1024 points (2K words)	4096 points (8K words)	4096 points (8K words)	Default value (Note 1)
	Timer	T	512 points (4K words)	2048 points (16K words)	2048 points (16K words)	
	Integrating timer	TR	128 points (1K words)	512 points (4K words)	512 points (4K words)	
	Counter	C	256 points (1K words)	1024 points (4K words)	1024 points (4K words)	
	Others		8K words	32K words	32K words	
System memories	SM, WSM, DSM		512 words			Fixed value
Parameters	V, WV, DV		-			(Note 2)
Temporary area			8K words (User specification disabled)			Fixed value
Tasks			Cyclic scan (default task): 1 task Fixed tasks + event tasks: 4 tasks in total			
Number of programs that can be registered			Max. 256 programs (max. 128 programs per task)			
Number of user FBs			512			
Nesting of user FBs			127 levels			
Number of user functions			512			
Nesting of user functions			127 levels			
Multi-CPU function			Max. 8 in one SPH system *A mixed system of the SPH300 series and SPH2000 series is possible.			
Redundant system			1:1 cold standby method, 1:1 warm standby method and N:1 cold standby method are supported.			

Notes: 1) The standard memories, retain memories, memories for user FBs and memories for system FBs not in the high-speed area can be increased or decreased by setting the CPU memory size definition of system definitions. This list gives the default values.
2) Parameters for user FBs (V, WV and DV) are allocated to instance memories for user FBs.
3) The I/O expansion function is supported by the following CPU versions: (NP1PS-74: V2364 or later, NP1PS-74R/117/117R: V2464 or later) NP1PS-32/32R does not support this function.

2-1 Performance Specifications

(2) NP1PS-245R

Item			Specification	Remarks
CPU models			NP1PS-245R	
Program memory capacity			245K steps	
Program steps in a program			8192 steps	
I/O memories	X, WX, DX		8192 points (including remote I/O points) * When the I/O expansion function is used: Max. 4096 wordss	Fixed value
	Y, WY, DY			
Standard memories	M, WM, DM	High speed	2K words	Fixed value
		Standard	254K words	Default value (Note 1)
Retain memories	L, WL, DL		127K words	Default value (Note 1)
Memories for user FBs	F, WF, DF		65K words	Default value (Note 1)
Memories for system FBs	Edge detection		4096 points (8K words)	Default value (Note 1)
	Timer	T	2048 points (16K words)	
	Integrating timer	TR	512 points (4K words)	
	Counter	C	1024 points (4K words)	
	Others		32K words	
System memories	SM, WSM, DSM		512 words	Fixed value
Parameters	V, WV, DV		-	(Note 2)
Temporary area			8K words (User specification disabled)	Fixed value
Tasks			Cyclic scan (default task): 1 task Fixed tasks + event tasks: 4 tasks in total	
Number of programs that can be registered			Max. 256 programs (max. 128 programs per task)	
Number of user FBs			512	
Nesting of user FBs			127 levels	
Number of user functions			512	
Nesting of user functions			127 levels	
Multi-CPU function			Max. 8 in one SPH system *A mixed system of the SPH300 series and SPH2000 series is possible.	
Redundant system			1:1 cold standby method, 1:1 warm standby method and N:1 cold standby method are supported.	

Notes: 1) The standard memories, retain memories, memories for user FBs and memories for system FBs not in the high-speed area can be increased or decreased by setting the CPU memory size definition of system definitions. This list gives the default values.

2) Parameters for user FBs (V, WV and DV) are allocated to instance memories for user FBs.

3) When the memory size is changed, set it so that the standard memory + retain memory <= 392192 words.

4) When a program of this CPU is stored in a memory card if module (type: NP1F-MM1), the maximum program memory is 119808 steps and the maximum program capacity in one POU is 4096 steps.

5) NP1PS-245R is supported by SX-Programmer Standard V2.2.1.0 or later.

2-1 Performance Specifications

2-1-2 SPH2000

(1) NP1PM-48R/48E

Item			Specification	Remarks
CPU models			NP1PM-48R/48E	
Program memory capacity			48K steps	
Program steps in a program			16384 steps	
I/O memories	X, WX, DX		8192 points (including remote I/O points)	Fixed value
	Y, WY, DY			
Standard memories	M, WM, DM		64K words	Default value (Note 1)
Retain memories	L, WL, DL		8K words	Default value (Note 1)
Memories for user FBs	F, WF, DF		8K words	Default value (Note 1)
Memories for system FBs	Edge detection		1024 points (2K words)	Default value (Note 1)
	Timer	T	512 points (4K words)	
	Integrating timer	TR	128 points (1K words)	
	Counter	C	256 points (1K words)	
	Others		8K words	
System memories	SM, WSM, DSM		512 words	Fixed value
Parameters	V, WV, DV		-	(Note 2)
Temporary area			32K words (For every task, user specification disabled)	Fixed value
Tasks			Cyclic scan (default task): 1 task Fixed tasks + event tasks: 4 tasks in total	
Number of programs that can be registered			Max. 256 programs (max. 128 programs per task)	
Number of user FBs			512	
Nesting of user FBs			124 levels	
Number of user functions			512	
Nesting of user functions			124 levels	
Ethernet interface			For NP1PM-48E only, 10BASE-T or 100BASE-TX	(Note 4)
Multi-CPU function			Max. 8 in one SPH system *A mixed system of the SPH300 series and SPH2000 series is possible.	
Redundant system			Not supported	

Notes: 1) The standard memories, retain memories, memories for user FBs and memories for system FBs not in the high-speed area can be increased or decreased by setting the CPU memory size definition of system definitions. This list gives the default values.
2) Parameters for user FBs (V, WV and DV) are allocated to instance memories for user FBs.
3) NP1PM-48R is supported by SX-Programmer Standard V2.2.0.0 or later.
4) For the specifications and operations of the Ethernet interface, refer to "User's Manual SPH2000 Ethernet Communications (FEH193)".

(2) NP1PM-256E

Item			Specification	Remarks
CPU models			NP1PM-256E	
Program memory capacity			256K steps	
Program steps in a program			16384 steps	
I/O memories	X, WX, DX		8192 points (including remote I/O points)	Fixed value
	Y, WY, DY			
Standard memories	M, WM, DM		1664K words	Default value (Note 1)
Retain memories	L, WL, DL		256K words	Default value (Note 1)
Memories for user FBs	F, WF, DF		64K words	Default value (Note 1)
Memories for system FBs	Edge detection		4096 points (8K words)	Default value (Note 1)
	Timer	T	2048 points (16K words)	
	Integrating timer	TR	512 points (4K words)	
	Counter	C	1024 points (4K words)	
	Others		32K words	
System memories	SM, WSM, DSM		512 words	Fixed value
Parameters	V, WV, DV		-	(Note 2)
Temporary area			32K words (For every task, user specification disabled)	Fixed value
Tasks			Cyclic scan (default task): 1 task Fixed tasks + event tasks: 4 tasks in total	
Number of programs that can be registered			Max. 256 programs (max. 128 programs per task)	
Number of user FBs			512	
Nesting of user FBs			124 levels	
Number of user functions			512	
Nesting of user functions			124 levels	
Ethernet interface			10BASE-T or 100BASE-TX	(Note 4)
Multi-CPU function			Max. 8 in one SPH system *A mixed system of the SPH300 series and SPH2000 series is possible.	
Redundant system			Not supported	

Notes: 1) The standard memories, retain memories, memories for user FBs and memories for system FBs not in the high-speed area can be increased or decreased by setting the CPU memory size definition of system definitions. This list gives the default values.
2) Parameters for user FBs (V, WV and DV) are allocated to instance memories for user FBs.
3) NP1PM-48R is supported by SX-Programmer Standard V2.2.0.0 or later.
4) For the specifications and operations of the Ethernet interface, refer to “User’s Manual SPH2000 Ethernet Communications (FEH193)”.

2-2 Memory

2-1-3 SPH200

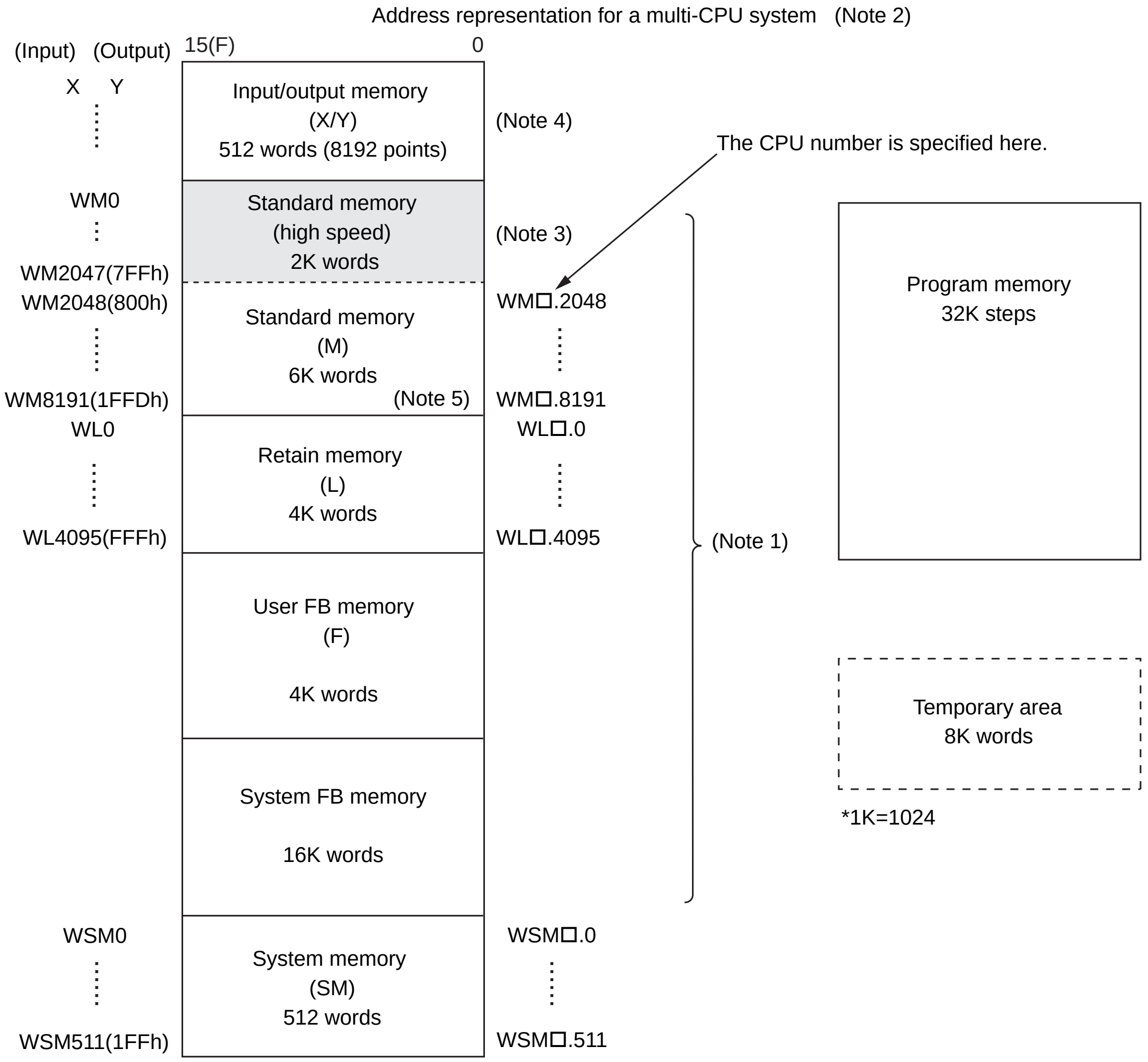
Item			Specification		Remarks
CPU models			NP1PH-08	NP1PH-16	
Program memory capacity			8K steps	16K steps	
Program steps in a program			4096 steps (CPU with software version earlier than 30) 2048 steps (CPU with software version 30 or later)		
I/O memories	X, WX, DX		8192 points (including remote I/O points)		Fixed value
	Y, WY, DY				
Standard memories	M, WM, DM		4K words	8K words	Default value (Note 1)
Retain memories	L, WL, DL		2K words	4K words	Default value (Note 1)
Mmories for user FBs	F, WF, DF		2K words	4K words	Default value (Note 1)
Memories for system FBs	Edge detection		256 points (512 words)	512 points (1K words)	Default value (Note 1)
	Timer	T	128 points (1K words)	256 points (2K words)	
	Integrating timer	TR	32 points (256 words)	64 points (512 words)	
	Counter	C	64 points (256 words)	128 points (512 words)	
	Others		2K words	4K words	
System memories	SM, WSM, DSM		512 words		Fixed value
Parameters	V, WV, DV		-		(Note 2)
Temporary area			8K words (User specification disabled)		Fixed value
Tasks			Cyclic scan (default task): 1 task Fixed tasks + event tasks: 4 tasks in total		
Number of programs that can be registered			Max. 64 programs (max. 64 programs per task)		
Number of user FBs			256		
Nesting of user FBs			64 levels		
Number of user functions			256		
Nesting of user functions			64 levels		
Multi-CPU function			Not supported		
Redundant system			Not supported		

Notes: 1) The standard memories, retain memories, memories for user FBs and memories for system FBs not in the high-speed area can be increased or decreased by setting the CPU memory size definition of system definitions. This list gives the default values.

2) Parameters for user FBs (V, WV and DV) are allocated to instance memories for user FBs.

2-2-1 Memory map

(1) NP1PS-32/NP1PS-32R (SPH300)



Notes: 1) The size of standard memory (excluding high-speed memory), retain memory, user FB memory, and system FB memory may be increased or decreased by the loader settings. In these memories, default values have been stored. For details refer to “4-4-2 Defining the CPU memory sizes.”

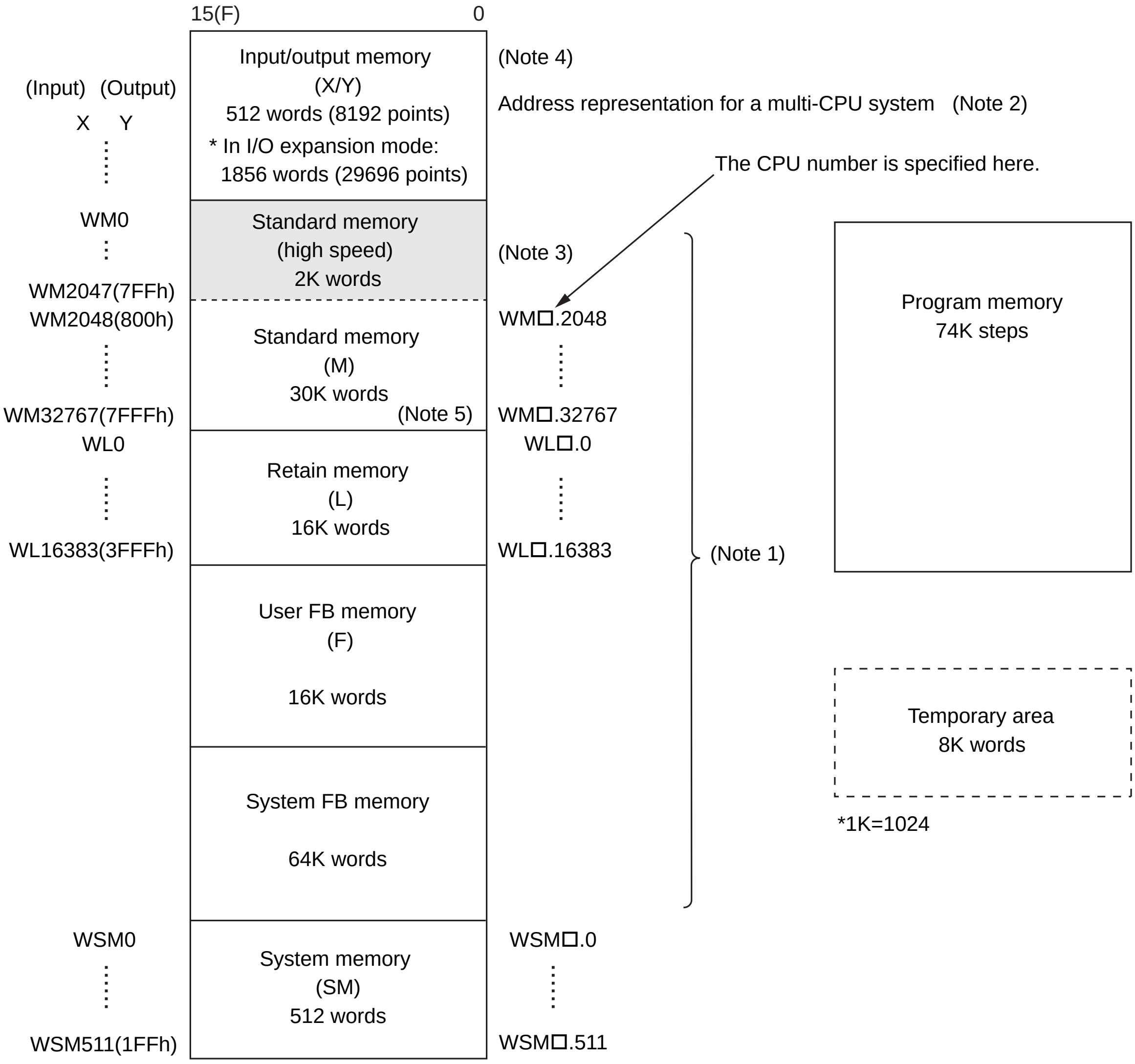
2) When another CPU memory is accessed in a multi-CPU system, the CPU number is specified in the □. This is not necessary to access own memory.

3) The first 2K words of the standard memory (M) are made up of high-speed memory which is accessed at a higher speed. Another CPU can not access this area as global memory. Its size cannot be altered.

4) For the input/output address assignments, refer to “2-3 Input/output Address Assignments.”

5) The last two words in the standard memory area are reserved by the system.

(2) NP1PS-74/NP1PS-74R (SPH300)



Notes: 1) The size of standard memory (excluding high-speed memory), retain memory, user FB memory, and system FB memory may be increased or decreased by the loader settings. In these memories, default values have been stored. For details refer to “4-4-2 Defining the CPU memory sizes.”

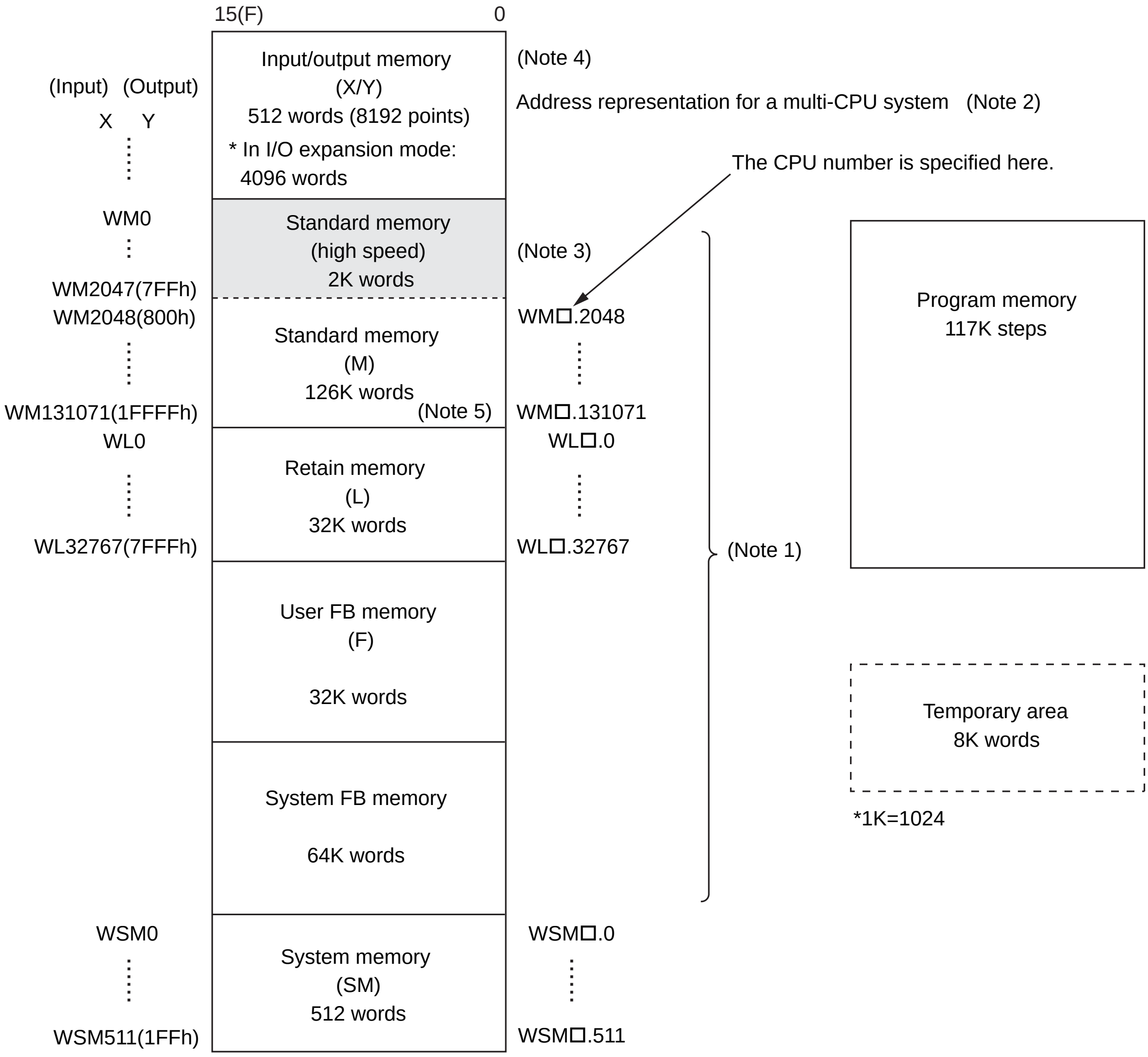
2) When another CPU memory is accessed in a multi-CPU system, the CPU number is specified in the □. This is not necessary to access own memory.

3) The first 2K words of the standard memory (M) are made up of high-speed memory which is accessed at a higher speed. Another CPU can not access this area as global memory. Its size cannot be altered.

4) For the input/output address assignments, refer to “2-3 Input/output Address Assignments.”

5) The last two words in the standard memory area are reserved by the system.

(3) NP1PS-117/NP1PS-117R (SPH300)



Notes: 1) The size of standard memory (excluding high-speed memory), retain memory, user FB memory, and system FB memory may be increased or decreased by the loader settings. In these memories, default values have been stored. For details refer to “4-4-2 Defining the CPU memory sizes.”

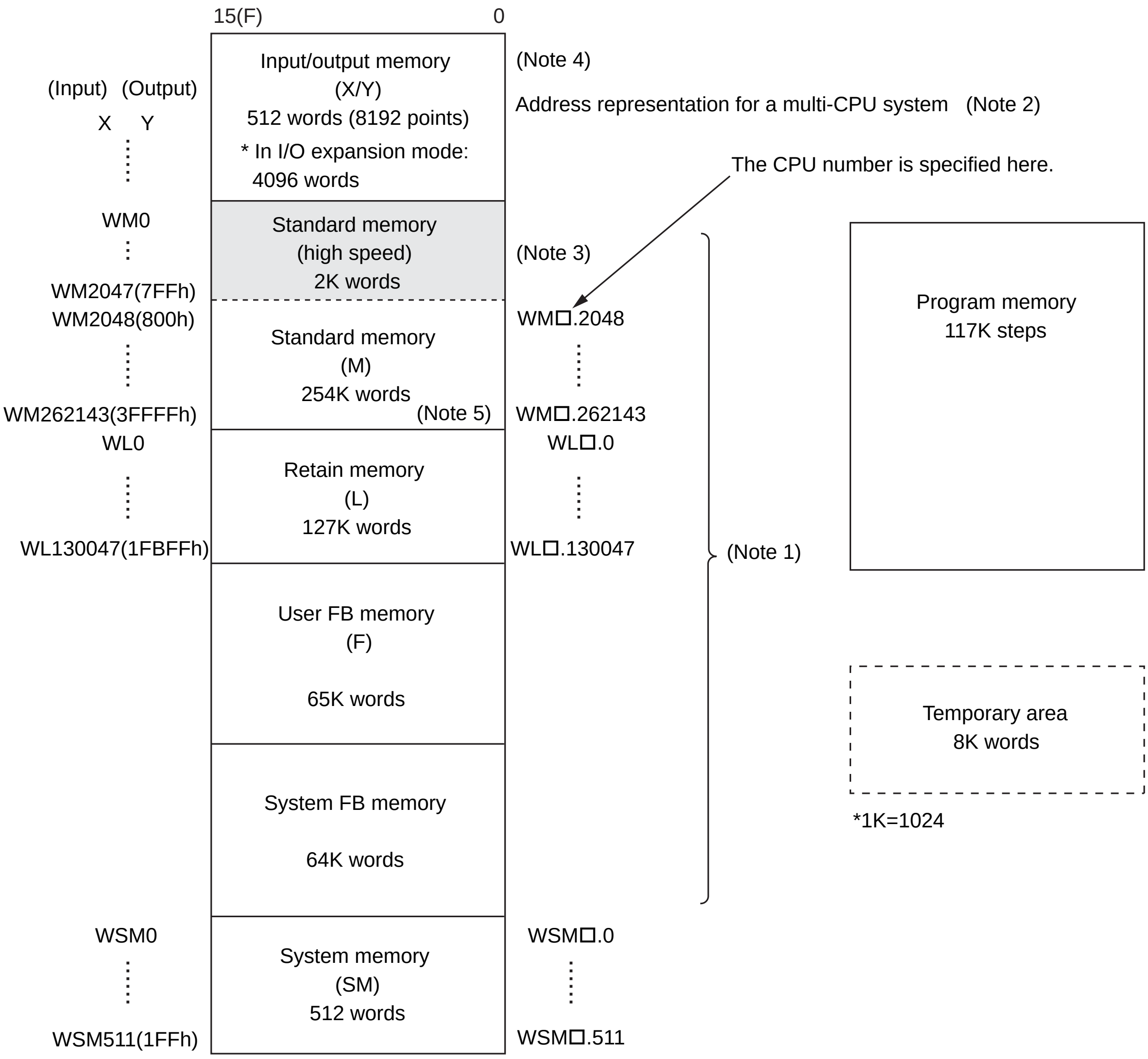
2) When another CPU memory is accessed in a multi-CPU system, the CPU number is specified in the □. This is not necessary to access own memory.

3) The first 2K words of the standard memory (M) are made up of high-speed memory which is accessed at a higher speed. Another CPU can not access this area as global memory. Its size cannot be altered.

4) For the input/output address assignments, refer to “2-3 Input/output Address Assignments.”

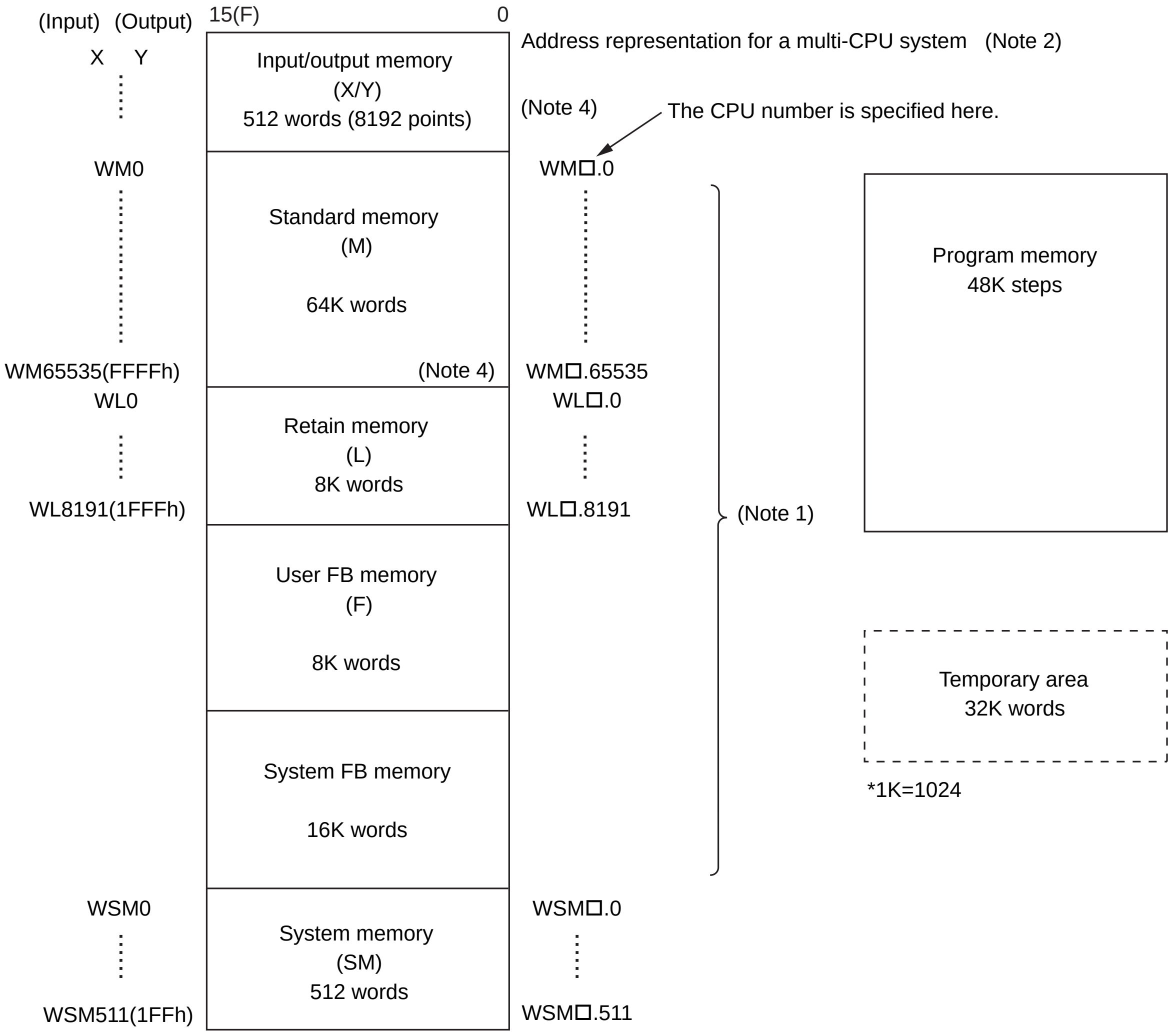
5) The last two words in the standard memory area are reserved by the system.

(4) NP1PS-245R (SPH300)



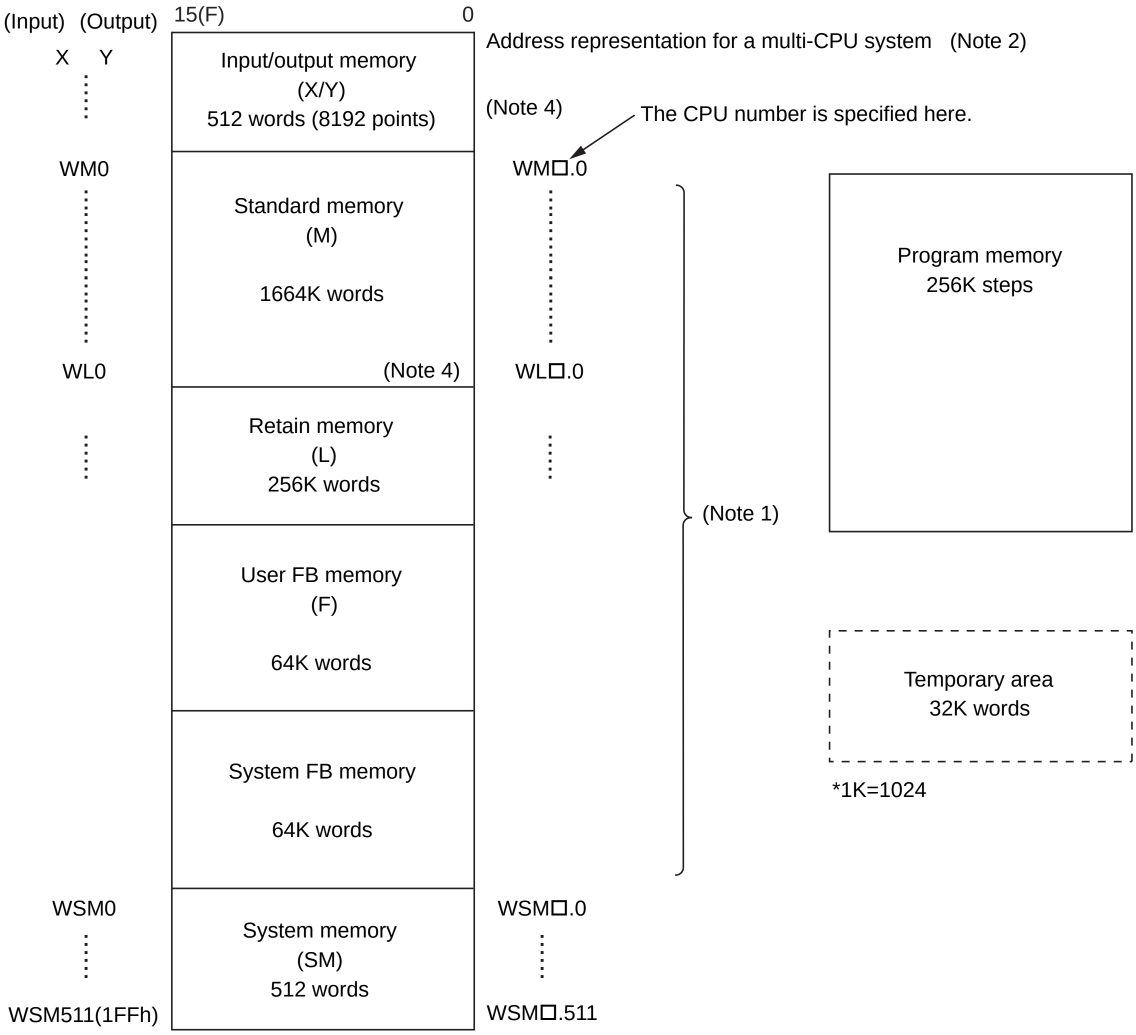
Notes: 1) The size of standard memory (excluding high-speed memory), retain memory, user FB memory, and system FB memory may be increased or decreased by the loader settings. In these memories, default values have been stored. For details refer to “4-4-2 Defining the CPU memory sizes.”
* When the memory size is changed, set it so that the standard memory + retain memory <= 392192 words.
2) When another CPU memory is accessed in a multi-CPU system, the CPU number is specified in the ☐. This is not necessary to access own memory.
3) The first 2K words of the standard memory (M) are made up of high-speed memory which is accessed at a higher speed. Another CPU can not access this area as global memory. Its size cannot be altered.
4) For the input/output address assignments, refer to “2-3 Input/output Address Assignments.”
5) The last two words in the standard memory area are reserved by the system.
6) NP1PS-245R is supported by the SX-Programmer Standard V2.2.1.0 or later.

(5) NP1PM-48R/48E (SPH2000)



Notes: 1) The size of standard memory (excluding high-speed memory), retain memory, user FB memory, and system FB memory may be increased or decreased by the loader settings. In these memories, default values have been stored. For details refer to “4-4-2 Defining the CPU memory sizes.”
2) When another CPU memory is accessed in a multi-CPU system, the CPU number is specified in the □ and the direct read/write FB is used. This is not necessary to access own memory.
3) To accress memories between CPUs using the processor bus, it is necessary to reserve memory for the multi-CPU. For details, refer to “4-4-2 Defining the CPU memory sizes.”
4) For the input/output address assignments, refer to “2-3 Input/output Address Assignments.”
5) The last two words in the standard memory area are reserved by the system.
6) NP1PM-48R is supported by the SX-Programmer Standard V2.2.0.0 or later.
NP1PM-48E is supported by the SX-Programmer Standard V2.2.2.0 or later.

(6) NP1PM-256E (SPH2000)



Notes: 1) The size of standard memory (excluding high-speed memory), retain memory, user FB memory, and system FB memory may be increased or decreased by the loader settings. In these memories, default values have been stored. For details refer to "4-4-2 Defining the CPU memory sizes."

2) When another CPU memory is accessed in a multi-CPU system, the CPU number is specified in the □ and the direct read/write FB is used. This is not necessary to access own memory.

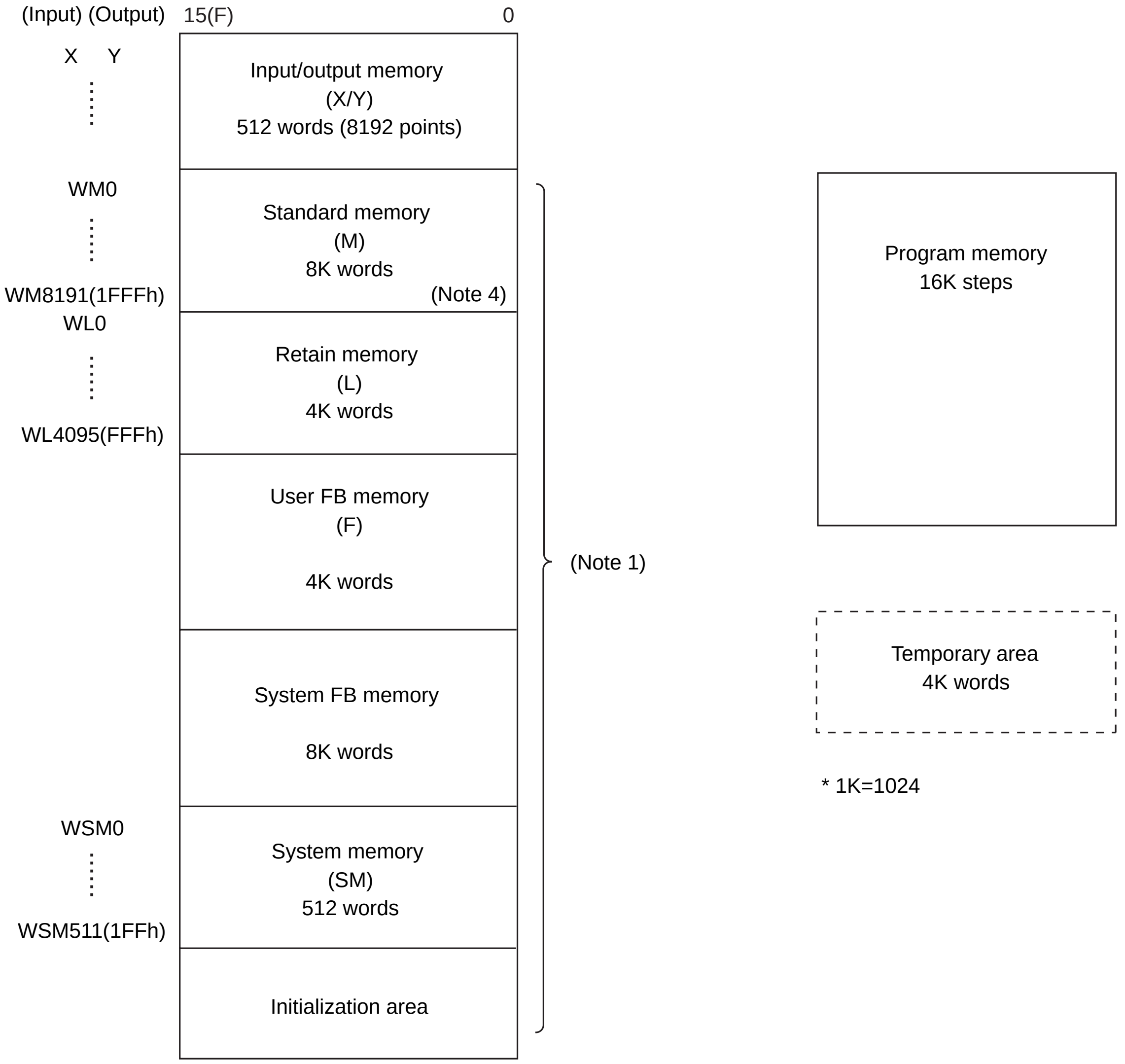
3) To accress memories between CPUs using the processor bus, it is necessary to reserve memory for the multi-CPU. For details, refer to "4-4-2 Defining the CPU memory sizes."

4) For the input/output address assignments, refer to "2-3 Input/output Address Assignments."

5) The last two words in the standard memory area are reserved by the system.

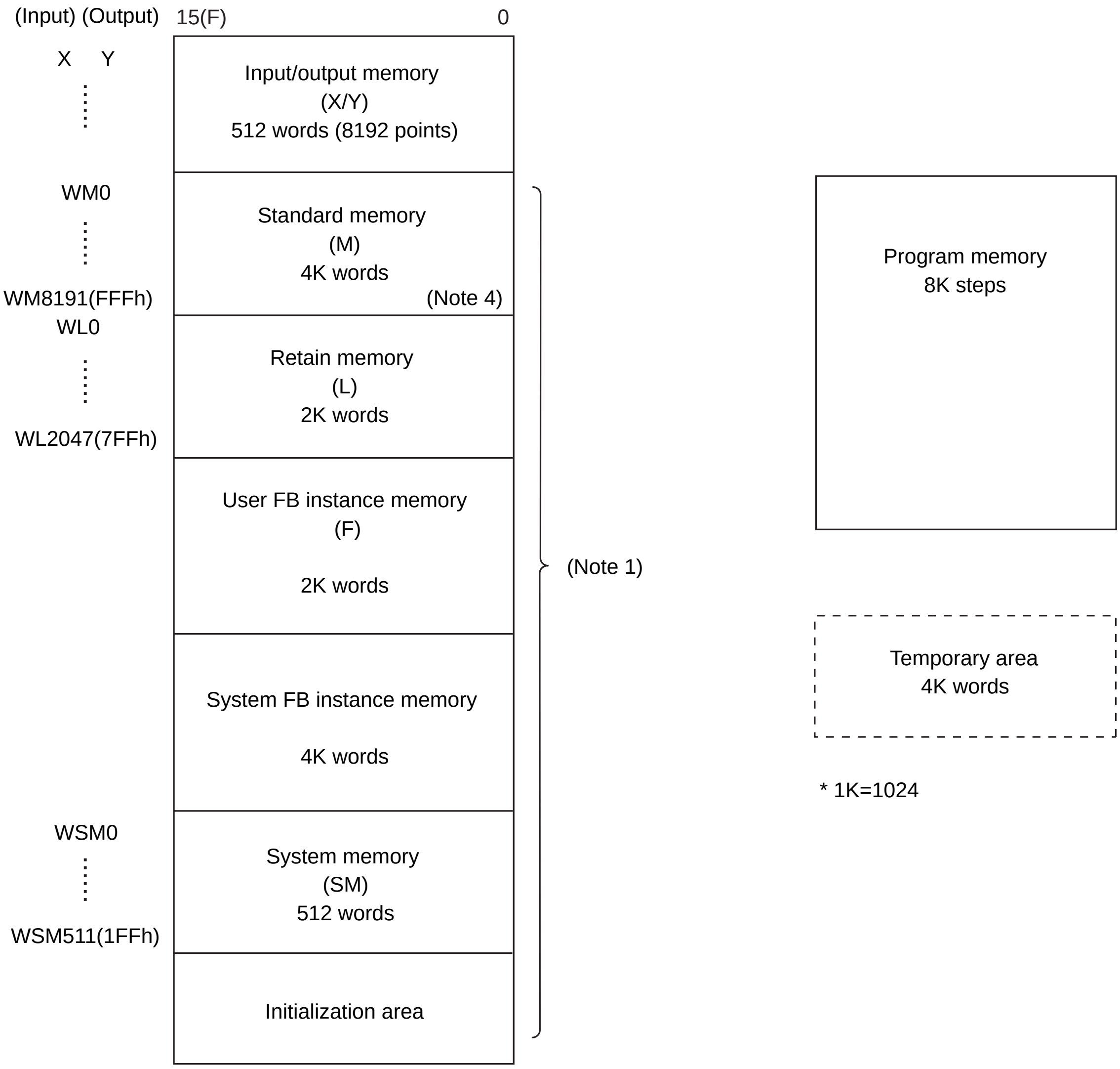
6) NP1PM-256E is supported by the SX-Programmer Standard V2.2.2.0 or later.

(7) NP1PH-16 (SPH200)



Notes: 1) The size of standard memory, retain memory, user FB memory, system FB memory, and Initialization area may be increased or decreased by the loader settings. In these memories, default values have been stored. For details refer to “4-4-2 Defining the CPU memory sizes.”
2) No multi-system can be built up using a standard CPU module.
3) For the input/output address assignments, refer to “2-3 Input/output Address Assignments.”
4) The last two words in the standard memory area are reserved by the system.

(8) NP1PH-08 (SPH200)



Notes: 1) The size of standard memory, retain memory, user FB memory, system FB memory, and Initialization area may be increased or decreased by the loader settings. In these memories, default values have been stored. For details refer to “4-4-2 Defining the CPU memory sizes.”
2) No multi-system can be built up using a standard CPU module.
3) For the input/output address assignments, refer to “2-3 Input/output Address Assignments.”
4) The last two words in the standard memory area are reserved by the system.

2-2-2 Description of memories

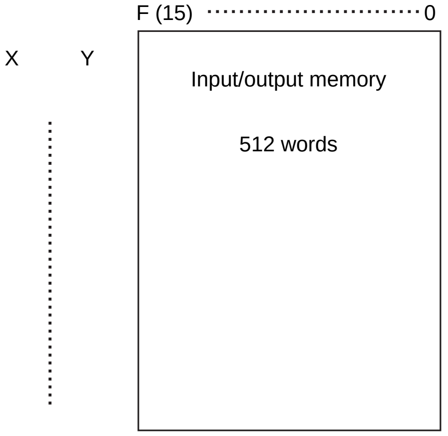
(1) Input/output memory area (512 words)

The input/output memory area is a window through which data is exchanged between the CPU and external devices. It is used by input devices such as pushbuttons, switches, and sensors which send data to the CPU and by output devices such as relays, solenoids, and indicators which show the results of program executions.

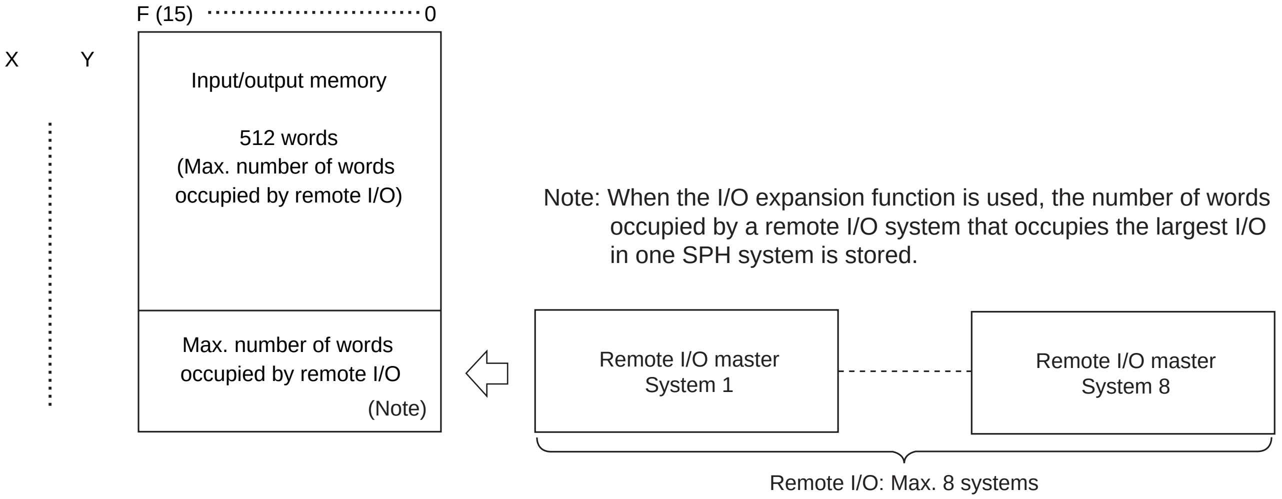
Key points

1) Input is represented by X and output by Y (identifier).
   X/Y: bit, WX/WY: word, DX/DY: double word
   For details, refer to “2-3 Input/Output Address Assignments.”
2) I/Os directly connected to the SX bus and remote I/Os such as T-link are allocated to this area.
   SPH300 requires 81 takts to execute one cycle of data reading or writing operation between this area and external equipment, such as POD. It is recommended to access once the internal memory when reading or writing data from or to external equipment.
3) Both input and output cannot exist in the same word.
4) SPH300 (NP1PS-74: V2364 or later, NP1PS-74R/117/117R: V2664 or later, NP1PS-245R: V2064 or later) supports the I/O expansion function.

<Normal mode>



<I/O expansion mode>



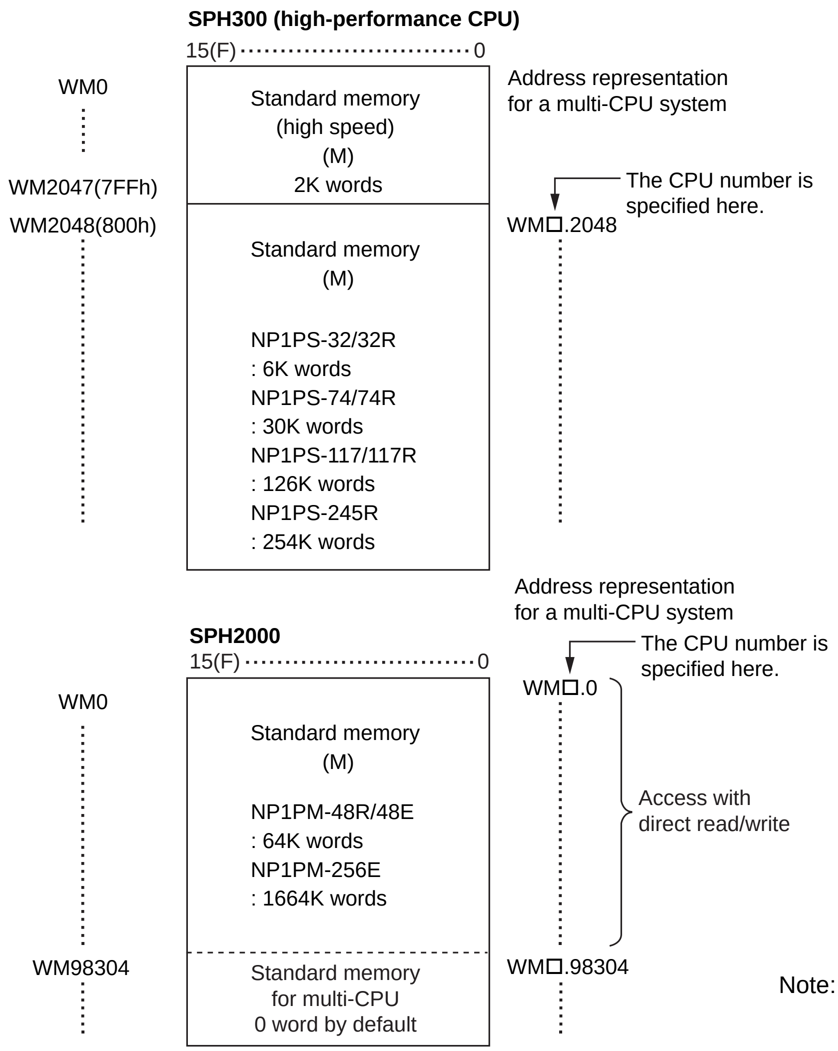
* For details, refer to “Appendix 6 I/O expansion function”.

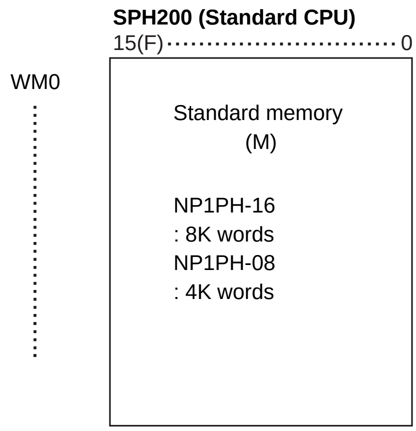
(2) Standard memory area (M)

The standard memory area is used for auxiliary relays that are used internally in the PLC.

Key points

1) Address specifications are represented with M (M: bit, WM: word, DM: double word).
2) The specified memory area is reset to (0) zero when self-PLC is started.
3) For SPH300, 2K words from the top of the standard memory is the area where data access in the CPU is processed at high speed. On the other hand, when data is accessed from external device such as POD, operation requires 81 takts to execute one cycle of data reading or writing operation.
4) In the multi-CPU system, the other areas can be accessed as global memory areas from any other CPU (SPH300, SPH2000).
5) In a multi-CPU system of the SPH2000, to access memories between CPUs with an ordinary instruction using the processor bus, it is necessary to reserve memory for the multi-CPU.
6) The sizes of the standard memory areas can be modified taking those of other areas into account. Note that the size of a fast access memory area is 2K-word fixed in the SPH300 and cannot be modified.
   Refer to “4-4-2 CPU Memory Size Definition” for modifying memory sizes.
7) For SPH300, no continuous access is allowed to the fast access memory area and the boundaries between other areas.





Note: The numbers of words are default values in the memory map on the left side.

(3) Retain memory area (L)

This is a memory area for auxiliary relays with attributes held during power failure used inside PLC.

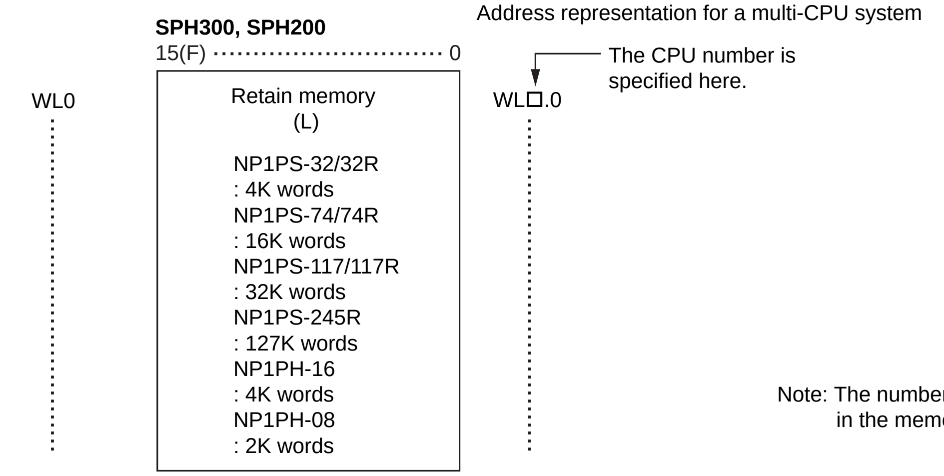
Key points

1) Address specifications are represented with L.
   L: bit, WL: word, DL: double word
2) The following processes are performed at cold or warm start.

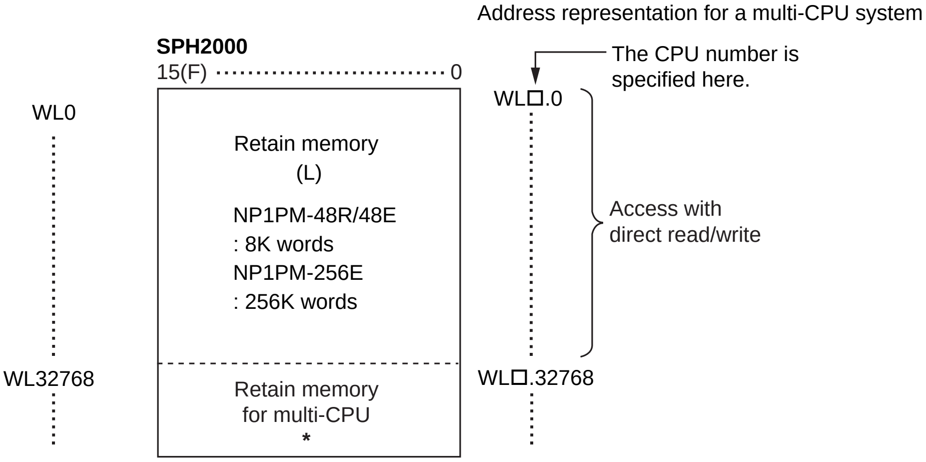
(Note)

	Cold	Warm
Retain memory	Reset to 0 (zero)	Retains old values
Initialized retain memory	Writes specified initial values	Retains old values

3) When a project is transfered, you have an option for selecting whether the area is to be cleared at project transfered. If you select "clear," the system cold-starts and If "not clear," the system warm-starts.
4) In the multi-CPU system, the retain memory area can be accessed as a global memory area from any other CPU (only for SPH300, SPH2000).
5) The size of the retain memory area can be modified taking those of other areas into account. Refer to "4-4-2 CPU Memory Size Definition" for modifying memory sizes.



Note: The numbers of words are default values in the memory map on the left side.



* NP1PM-48R/48E: 0 word by default
* NP1PM-256E: 224K words by default

Note: The system cold-starts when initially starting from the loader while it warm-starts at power-on or when starting from the loader.

(4) User FB memory area

The user FB memory area is an instance memory area unique to each user FB used internally in the PLC.

Key points

1) Do not read out from and write data into the instance memory area from an application program or the other loaders.
   If you neglect this advice, the user FB may not operate correctly.
2) This area can be modified taking those sizes of other areas into account.
   Refer to “4-4-2 CPU Memory Size Definition” for modifying memory sizes.

WF0

User FB memory
(F)

NP1PS-32/32R
: 4K words
NP1PS-74/74R
: 16K words
NP1PS-117/117R
: 32K words
NP1PS-245R
: 65K words
NP1PM-48R/48E
: 8K words
NP1PM-256E
: 64K words
NP1PH-16
: 4K words
NP1PH-08
: 2K words

Note: The numbers of words are default values
in the memory map on the left side.

(5) System FB memory area

The system FB memory area is a memory area for working to each of the system FBs, such as timers, counters, and differential instructions, which are used internally in the PLC.

Key points

1) Do not read out from and write data into the memory area from an application program or the loaders.
   If you neglect this advice, the user FB may not operate correctly.
2) At PLC startup, predefined initialization is performed. (the old values are retained or reset to 0s (zeros).
   Example) The current values for the counters and totalizing timers and the old value for the edge detect instruction counter are retained, while the current values for the timers (excluding totalizing) are reset to 0s (zeros).
3) Eight words/timer, four words/counter, and two words/edge detect instruction counter are used.
4) The size of the system FB memory area can be modified taking those of other areas into account.
   Refer to “4-4-2 CPU Memory Size Definition” for modifying memory sizes.
5) By default, the numbers of timers, totalizing timers, counters, and edge detect instruction counters have been preset per CPU. If necessary, these numbers may be modified.

	Timer (T)	Total. timer (TR)	Counter (C)	Edge detect	Others
NP1PS-32	512 points	128 points	256 points	1024 points	8192 words
NP1PS-74/74R NP1PS-117/117R NP1PS-245R NP1PM-256E	2048 points	512 points	1024 points	4096 points	32768 words
NP1PM-48R/48E	512 points	128 points	256 points	1024 points	8192 words
NP1PH-16	256 points	64 points	128 points	512 points	4096 words
NP1PH-08	128 points	32 points	64 points	256 points	2048 words

To set the timer, totalizing timer, counter, edge detect, and other system FB areas, the following conditions should be met.

(No. of timers) x 8 words + (No. of counters) 4 words + (No. of edge detect counters) x 2 words + others
<= Preset size of system FB memory area

6) If the timer, totalizing timer, or counter is set at a value ranging from 8192 (2000(h)) to 16383 (3FFFh(h)), it becomes a global timer which can be accessed from all programs.
   The numbers of timers, totalizing timers, and counters must be as follows:
   (No. of timers of memory size definition) >= (No.of local timers) + (No. of global timers)
   (No. of totalizing timers of memory size definition) >= (No.of local totalizing timers) + (No. of global totalizing timers)
   (No. of counters of memory size definition) >= (No.of local counters) + (No. of global counters)

   Note: The global timer is supported by the Standard loader V2.2.0.0 or later. The global totalizing timer and global counter is suported by Standard loader V2.2.2.0 or later.

Edge detect NP1PS-32/32R: 2K words NP1PS-74/74R/117/117R/245R, NP1PM-256E: 8K words NP1PM-48R/48E: 2K words NP1PH-08: 0.5K words, NP1PH-16: 1K words
Counter NP1PS-32/32R: 1K words NP1PS-74/74R/117/117R/245R, NP1PM-256E: 4K words NP1PM-48R/48E: 1K words NP1PH-08: 0.25K words, NP1PH-16: 0.5K words
Total. timer NP1PS-32/32R: 1K words NP1PS-74/74R/117/117R/245R, NP1PM-256E: 4K words NP1PM-48R/48E: 1K words NP1PH-08: 0.25K words, NP1PH-16: 0.5K words
Timer NP1PS-32/32R: 4K words NP1PS-74/74R/117/117R/245R, NP1PM-256E: 16K words NP1PM-48R/48E: 4K words NP1PH-08: 1K words, NP1PH-16: 2K words
Others NP1PS-32/32R: 8K words NP1PS-74/74R/117/117R/245R, NP1PM-256E: 32K words NP1PM-48R/48E: 8K words NP1PH-08: 2K words, NP1PH-16: 4K words

Note: The numbers of words are default values in the memory map on the left side.

(6) Initialization area

The initialization area assigned only in the standard CPU stores the initial values for the user function block (FB) and variables.

Key points

1) The sizes of the storing areas are calculated by the following expression.
(Initialization area) = (No. of words in user FB area) x 9/8
+ (No. of variables for which initial values are set) x 5 (words)

2) The default values have been preset as shown below.

	Initialization area (entire)	Initial user FB value storing area	Initial variable value storing area
NP1PH-16	7K words	4608 words	2560 words
NP1PH-08	3K words	2304 words	768 words

From the table shown above, the numbers (default value) of variables for which initial values can be set are calculated by the expressions below.
NP1PH-16: 2560 / 5 = 512 (Any digits under the decimal point are truncated) 512 points
NP1PH-08: 768 / 5 = 153 (Any digits under the decimal point are truncated) 153 points

3) The size of a user FB initialization area requires: No. of words of the preset user FB area x 9/8

Initialization area

Initial user FB value storing area
NP1PH-16: 4608 words (default)
NP1PH-08: 2304 words (default)

Initial variable value storing area
NP1PH-16: 2560 words (default)
NP1PH-08: 768 words (default)

* For SPH300, a storage area where 3200 initial values, total of standard memory and retain memory, can be stored is provided. It is not necessary to consider the “Initial value storing area.”
In addition, for SPH300, available space of the instance memory for user FB can be used for the initialization for variables in the user FB.

* For SPH2000, a storing area where 12800 initial values (3200 values for software version V01), total of standard memory and retain memory, can be stored is provided. It is not necessary to consider the “Initial value storing area.”
In addition, for SPH2000, available space of the instance memory for user FB can be used for the initialization for variables in the user FB.

(7) System memory area (512 words)

The system memory area is allocated to flags which indicate the operating status or the error status of the MICREX-SX series. This area is for exclusive use.

A resource is defined as one CPU system that configured by one CPU module and two or more I/O modules.

<System memory>

WSM0	Resource operating status
WSM1	Resource switch information/User ROM state
WSM2	Resource fatal fault factor
WSM3	Not used
WSM4	Resource nonfatal fault factor
WSM5	Not used
WSM6	CPU error factor
WSM7	Not used
WSM8, 9	Memory error factor
WSM10(A) WSM11(B)	SX bus error factor
WSM12(C)	Application error factor (fatal fault)
WSM13(D)	Application error factor (nonfatal fault)
WSM14(E) | WSM16(10)	User fatal fault factor 0 - factor 47
WSM17(11)	Not used
WSM18(12) | WSM20(14)	User nonfatal fault factor 0 - factor 47
WSM21(15)	Not used
WSM22(16) | WSM29(1D)	System definition error factor
WSM30(1E) | WSM37(25)	Not used
WSM38(26) WSM39(27)	Application program error factor
WSM40(28) WSM41(29)	Not used
WSM42(2A) WSM43(2B)	Annunciator relay
WSM44(2C) WSM45(2D)	Not used
WSM46(2E)	Redundant master annunciator
WSM47(2F)	Redundant operation mode
WSM48(30),49(31)	Resource operation/running information
WSM50(32),51(33)	Resource configuration/fault information
WSM52(34) | WSM67(43)	SX bus configuration information (SPH system configuration information)
WSM68(44) | WSM83(53)	SX bus fault information (SPH system fault information)
WSM84(54) | WSM99(63)	SX bus-connected module fail-soft mode information

WSM100(64) | WSM127(7F)	Not used
WSM128(80) | WSM135(87)	Remote I/O master 0 I/O module configuration information
WSM136(88) | WSM143(8F)	Remote I/O master 0 I/O module fault information
WSM144(90) | WSM151(97)	Remote I/O master 1 I/O module configuration information
WSM152(98) | WSM159(9F)	Remote I/O master 1 I/O module fault information
WSM160(A0) | WSM167(A7)	Remote I/O master 2 I/O module configuration information
WSM168(A8) | WSM175(AF)	Remote I/O master 2 I/O module fault information
WSM176(B0) | WSM183(B7)	Remote I/O master 3 I/O module configuration information
WSM184(B8) | WSM191(BF)	Remote I/O master 3 I/O module fault information
WSM192(C0) | WSM199(C7)	Remote I/O master 4 I/O module configuration information
WSM200(C8) | WSM207(CF)	Remote I/O master 4 I/O module fault information
WSM208(D0) | WSM215(D7)	Remote I/O master 5 I/O module configuration information
WSM216(D8) | WSM223(DF)	Remote I/O master 5 I/O module fault information
WSM224(E0) | WSM231(E7)	Remote I/O master 6 I/O module configuration information
WSM232(E8) | WSM239(EF)	Remote I/O master 6 I/O module fault information
WSM240(F0) | WSM247(F7)	Remote I/O master 7 I/O module configuration information
WSM248(F8) | WSM255(FF)	Remote I/O master 7 I/O module fault information

To be continued →

WSM256(100)	Fail-soft maintenance operation prohibition mode / Error state display hiding mode, etc.
WSM257(101) | WSM272(110)	Station No. with error state display hidden
WSM273(111)	I/O expansion mode status
WSM274(112) | WSM439(1B7)	Not used
WSM440(1B8) | WSM445(1BD)	Ethernet interface information (for CPU with Ethernet function only)
WSM446(1BE) | WSM507(1FB)	Not used
WSM508(1FC) | WSM511(1FF)	SX bus transmission error rate information

* The inside of the parenthesis are hexadecimal indications.

* Do not write data into unused areas.

1) Resource operating status WSM0 (Read only)

The table given below shows the resource (CPU module) operating status and operating modes.

Address	Name	Description	SPH 300	SPH 200
SM00	Run	Set to "ON" while the CPU is running.	O	O
SM01	Stop	Set to "ON" while the CPU is down.	O	O
SM02	Fatal fault	Set to "ON" when a fatal resource error has occurred.	O	O
SM03	Nonfatal fault	Set to "ON" when a non-fatal resource error has occurred.	O	O
SM04	Redundancy working station	Set to "ON" when a working CPU is running in the redundant mode.	O	-
SM05	Redundancy standby station	Set to "ON" when a standby CPU is running in the redundant mode.	O	-
SM06	1:1 redundancy	Set to "ON" when the system is in the 1-to-1 redundant mode.	O	-
SM07	N:1 redundancy	Set to "ON" when the system is in the N-to-1 redundant mode.	O	-
SM08	Non-automatic operation mode	Set to "ON" while in the non-automatic operation mode.	O	O
SM09	Automatic operation mode	Set to "ON" while in the automatic operation mode.	O	O
SM0A	Preceding state mode	Set to "ON" while in the preceding state mode.	O	O
SM0B	Battery-less run mode	Set to "ON" while in the battery-less run mode.	O	O
SM0C	Not used		-	-
SM0D (Note)	SX bus-connected module fail-soft mode	Set to "ON" when fail-soft may be performed for all the modules connected to the SX bus or individually reset.	O	O
SM0E	Processor bus master	Set to "ON" when the CPU module is controlling the processor bus.	O	O
SM0F	SX bus master	Set to "ON" when the CPU module is controlling the SX bus.	O	O

O: Supported, -: Not supported

Non-automatic operation mode

The mode in which the CPU will not start operation when the system power is turned on with the key switch on the CPU module front panel set to “RUN” or “TERM.” This is set by the “running specification at power on” of the CPU parameter.

Automatic operation mode

The mode in which the CPU will start operation when the system power is turned on with the key switch on the CPU module front panel set to “RUN” or “TERM.” This is set by the “running specification at power on” of the CPU parameter. The mode is enabled in the system resource configuration at power-on. (The automatic mode is on by default.)

Preceding state mode

The mode in which the CPU will start operation when system power is turned on with the key switch on the CPU module front panel set to “RUN”; when system power is turned on with the key switch set to “TERM,” the CPU will enter the preceding state (running or stopped) that was established when power was turned off in the preceding run.

Battery-less run mode

At system power-on, the entire memory is initialized to initial values or all zeros. Note that neither battery connection check nor voltage check is done. The mode is enabled by the “battery less run” of the CPU parameter. When the preceding state mode is turned on in the battery-less mode, the CPU enters the automatic operation mode.
For the standard CPU, this mode is not enabled unless a user ROM card is inserted.

Note: <When SM0D is OFF (Fail-soft has been disabled for the modules connected to the SX bus)>
For common modules, fail-soft is not performed, and for the I/O modules connected to the SX bus, it is not performed even if enabled by using the loader.

<When SM0D is ON (Fail-soft has been enabled for the modules connected to the SX bus)>
For common modules, fail-soft is performed, and for the I/O modules connected to the SX bus, it is performed because it has been enabled by using the loader.

2) Resource switch / User ROM state WSM1 (Read only)

This area indicates the state of the CPU module switches that control the resource.

Address	Name	Description	SPH 300	SPH 200
SM10 | SM13	CPU number	Indicates the 4-bit number (0-F) set using the CPU number setting switches on the CPU module front panel. A range of numbers 0-7 is allowed.	O	-
SM14 SM15	Not used		-	-
SM16	User ROM card connection state	1: connected 0: unconnected	O (Note 1)	O
SM17	User ROM card write protect	1: write-protected 0: write-permitted (Valid when SM16 is ON)	O (Note 1)	O
SM18	STOP position	Set to "ON" when the key switch is in the STOP position.	O	O
SM19	TERM position (bottom)	Set to "ON" when the key switch is in the TERM position (bottom).	O	O
SM1A	TERM position (top) (Note 2, 3)	Set to "ON" when the key switch is in the TERM position (top).	O	O
SM1B	RUN position	Set to "ON" when the key switch is in the RUN position.	O	O
SM1C | SM1F	Not used		-	-

Notes: 1) User ROM card (compact flash card) adapted models only.
2) The TERM position flag also turns on when the key switch is in an unknown state.
3) With user ROM card adapted high-performance CPU module, this is set to “ON” when the key switch is set to UROM_TERM position.

3) Resource fatal fault factor WSM2 (Read only)

This area indicates the factors of fatal faults that will stop the resource (one-CPU system).

Address	Name	Description	SPH 300	SPH 200
SM20	CPU error	Set to "ON" when a fatal fault has occurred in the CPU module.	O	O
SM21	Power supply fault	Set to "ON" when a power-off condition has occurred.	O	O
SM22	Memory error	Set to "ON" when an error has occurred in the memory in the CPU module.	O	O
SM23	SX bus error	Set to "ON" when SX bus error occurs, for example, the disconnection of cable or loop-back plug.	O	O
SM24	Application error	Set to "ON" when an error has been found in an application program or system definition.	O	O
SM25	I/O module error	Set to "ON" when a fault has occurred in any of the I/O modules controlled by the self-CPU module and fail-soft has been "disabled." When fail-soft has been "enabled," the entire system continues operating normally even if a fault has occurred in an I/O module.	O	O
SM26	Common module error	Set to "ON" when a fault has occurred in any of the common modules (excluding self-module) connected to the SX bus.	O	O
SM27	Relay-switching error	Set to "ON" when relay-switching cannot be performed in the redundant operation mode.	O	-
SM28 SM29	Not used		-	-
SM2A	Remote I/O module error on remote I/O master board	Set to "ON" when the system is down due to an error occurred on the remote I/O unit or module.	-	-
SM2B	Driver error		-	-
SM2C	Not used		-	-
SM2D	Other hardware error	Set to "ON" when an error has occurred in a CPU number selection switch.	O	O
SM2E	Not used		-	-
SM2F	User fatal fault	Set to "ON" when the user program turns on one of the user fatal fault flags (SM140 to SM16F).	O	O

4) Resource nonfatal fault factor WSM4 (Read only)

This area indicates the factors of faults that allow the resource to continue processing.

Address	Name	Description	SPH 300	SPH 200
SM40 SM41	Not used		-	-
SM42	Memory error	Set to "ON" when an error has occurred in the memory of the self-CPU module.	O	O
SM43	SX bus error	Set to "ON" when an error has occurred in the SX bus.	O	O
SM44	Application error	Set to "ON" when an error has been found in an application program or system definition.	O	O
SM45	I/O module error	Set to "ON" when a fault has occurred in any of the I/O modules controlled by the self-CPU module and fail-soft has been enabled. (Note)	O	O
SM46	Common module error (Note *)	Set to "ON" when a fault has occurred in any of the I/O modules (excluding self-CPU) connected to the SX bus.	O	O
SM47 | SM49	Not used		-	-
SM4A	Remote I/O module error	Set to "ON" when an error has occurred on the remote I/O unit or module.	-	-
SM4B	Not used		-	-
SM4C	User ROM card - CPU mismatch	Set to "ON" when the content of user ROM card does not coincide with that of the CPU. Verification targets are system definition, project and password.	O (Note 1)	-
SM4D	Other hardware error	Set to "ON" when a fault in any of the key switches or loader/general-purpose communication selection switch occurs. The CPU module operates by assuming that "TERM" has been enabled if a fault has been detected in the key switch. It assumes that the loader side has been selected if a fault is detected in the loader/general-purpose communication selection switch.	O	O
SM4E	Battery error	Set to "ON" when the voltage of the data backup battery falls below the threshold level or the battery is dead.	O	O
SM4F	User nonfatal fault	Set to "ON" when the user program turns on one of the user non-fatal fault flags (SM180 to SM20F).	O	O

Note: 1) User ROM card (compact flash card) adapted models only

* The common modules are those connected to the SX bus without occupying an I/O area (for example, a CPU module, communication module, etc.).

5) CPU error factor WSM6 (Read only)

Address	Name	Description
SM60	Arithmetic processor error	Hardware error in the arithmetic LSI in the CPU module
SM61	OS processor error	Hardware error in the OS control LSI in the CPU module
SM62 | SM6F	Not used	

6) Memory error factor WSM8, WSM9 (Read only)

Address	Name	Description	Level
SM80	System ROM error	Set to "ON" when an error has occurred in the system ROM in the CPU module.	Fatal fault (Note 3)
SM81	System RAM error	Set to "ON" when an error has occurred in the system RAM in the CPU module.	Fatal fault (Note 3)
SM82	Application ROM error	Set to "ON" when an error has occurred in the user program ROM in the CPU module.	Fatal fault (Note 1) (Note 3)
SM83	Application RAM error	Set to "ON" when an error has occurred in the user program RAM in the CPU module.	Fatal fault (Note 3)
SM84 | SM8E	Not used		
SM8F	Memory backup error	Set to "ON" when no power-failure-time data is retained.	Fatal fault (Note 2)
SM90 | SM9E	Not used		
SM9F	Memory backup error	Set to "ON" when no power-failure-time data is retained. May clear the error condition by using an application program.	Nonfatal fault (Note 2)

Notes: 1) Set to “ON”when an error has occurred in the user ROM card.
2) For a high-performance CPU, the bit set to “ON” when a memory backup error has occurred depends on the module version.
Earlier than V**.25: SM8F, V10.30 or later: SM9F
3) If one of these errors occurs, execute “clear memory with on opening window” command from the Standard loader, and reload the project. If the error is still not remedied, it is a hardware fault.

System operation after a memory error has occurred
Any memory backup error resets the entire user memory area to 0 (zero). Note that in most cases, since SM80 to SM83 are set to “ON” when a hardware fault has occurred, cycling the power source may cause a memory error to be repeated.

7) SX bus error factor WSM10(A), WSM11(B)

Address	Name	Description	Level	SPH 300	SPH 200
SM100	SX bus LSI error	Set to "ON" when an error has occurred in the LSI controlling the SX bus.	Fatal fault	O	O
SM101	Station number double-assignment	Set to "ON" when the same SX bus station number has been assigned to more than one module in the SPH system.	Fatal fault	O	O
SM102	Module count exceeded	Set to "ON" when the number of modules connected to the SX bus exceeds 254.	Fatal fault	O	O
SM103 | SM10C	Not used			-	-
SM10D	SX bus transmission error	Set to "ON" when an error has occurred in transmission via the SX bus.	Fatal fault	O	O
SM10E	Processor bus access error	Set to "ON" when an error has occurred in accessing the processor bus. (the cause of the accessed error are attributable to the self-CPU module)	Fatal fault	O	O
SM10F	I/O refresh slow-down	Set to "ON" when the input/output data has not been updated on the SX bus for longer than 128ms.	Fatal fault	O	O
SM110 | SM11D	Not used			-	-
SM11E	Processor bus access error	Set to "ON" when an access error has occurred on the processor bus (the cause of the error is attributable to the accessed module). May clear the error condition by using an application program	Nonfatal fault	O	O
SM11F	Not used			-	-

8) Application error factor WSM12(C), WSM13(D) (Read only)

Address	Name	Description	Level	SPH 300	SPH 200
SM120	System definition error	Set to "ON" when an error has been found in the system definition.	Fatal fault	O	O
SM121	Application program error	Set to "ON" when an error has been found in the Application program.	Fatal fault	O	O
SM122 | SM12F	Not used			-	-
SM130	Not used			-	-
SM131	Application program error	Set to "ON" when an error has been found in the Application program.	Nonfatal fault	O	O
SM132 | SM13F	Not used			-	-

9) User fatal fault WSM14(E) to WSM16(10)

Address	Name	Description	SPH 300	SPH 200
SM140 | SM14F	User fatal fault factor 0 | User fatal fault factor 15	A fatal error has occurred and the CPU stops when either one of these bits is set to "ON" by an application program.	O	
SM150 | SM15F	User fatal fault factor 16 | User fatal fault factor 31			
SM160 | SM16F	User fatal fault factor 32 | User fatal fault factor 47			

10) User nonfatal fault WSM18(12) to WSM20(14)

Address	Name	Description	SPH 300	SPH 200
SM180 | SM18F	User nonfatal fault factor 0 | User nonfatal fault factor 15	A non-fatal error has occurred and the CPU continues running when either one of these bits is set to "ON" by an application program. Changing the bit set to "ON" to "OFF" by an application program lets the system recover from the non-fatal error state.	O	
SM190 | SM19F	User nonfatal fault factor 16 | User nonfatal fault factor 31			
SM200 | SM20F	User nonfatal fault factor 32 | User nonfatal fault factor 47			

11) System definition error factor SM22(16) to SM29(1D) (Read only)

Address	Name	Description	Level	SPH 300	SPH 200
SM220	Not used			-	-
SM221	System definition error	Set to "ON" when the contents of the system definition in the CPU module do not match the actual system configuration.	Fatal fault	O	O
SM222	System operation definition error	Set to "ON" when the Takt period is set to 0.5ms in a system in which two or more common modules are connected in one SPH system.	Fatal fault	O	O
SM223	System DO selection error	Set to "ON" when the SX bus direct-connect module defined in system DO (output) is not a digital output module.	Fatal fault	O	O
SM224	Redundant selection error	Set to "ON" when an error is found in the equivalent value range specification in the redundant mode definition.	Fatal fault	O	-
SM225	Fail-soft startup selection error	Set to "ON" when fail-soft startup is enabled if any module is not applicable to fail-soft in the system.	Fatal fault	O	O
SM226 | SM229	Not used			-	-
SM22A	CPU operation definition error	Set to "ON" when the switch setting in the CPU module is different from the CPU number set in the system definition.	Fatal fault	O	O
SM22B	CPU memory boundary definition error	Set to "ON" when the memory space used by an application program exceeds the total memory capacity.	Fatal fault	O	O
SM22C | SM22F	Not used			-	-
SM230	CPU I/O group definition error (for default tasks)	Set to "ON" when an input module is defined for an output module.	Fatal fault	O	O
SM231	CPU I/O group definition error (for level 0 tasks)				
SM232	CPU I/O group definition error (for level 1 tasks)				
SM233	CPU I/O group definition error (for level 2 tasks)				
SM234	CPU I/O group definition error (for level 3 tasks)				
SM235	Direct I/O connect fail-soft definition error	Set to "ON" when fail-soft is set for a module other than the input/output modules.	Fatal fault	O	O
SM236	Remote I/O master 0 fail-soft definition error	Set to "ON" when an error is found in the fail-soft definition.	Fatal fault	O	O
SM237	Remote I/O master 1 fail-soft definition error				
SM238	Remote I/O master 2 fail-soft definition error				
SM239	Remote I/O master 3 fail-soft definition error				
SM23A	Remote I/O master 4 fail-soft definition error				
SM23B	Remote I/O master 5 fail-soft definition error				
SM23C	Remote I/O master 6 fail-soft definition error				
SM23D	Remote I/O master 7 fail-soft definition error				
SM23E SM23F	Not used			-	-

SM240	I/O module hold definition error	Set to "ON" when hold is defined for a module other than output modules or the output module set for the system DO.	Fatal fault	O	O
SM241	I/O initialization error	Set to "ON" when an error is found in the operation setting for the module connected to the SX bus.	Fatal fault	O	O
SM242 | SM24F	Not used			-	-
SM250	Remote I/O master 0 initialization error	Set to "ON" when an error is found in a remote I/O master initialization.	Fatal fault	O	O
SM251	Remote I/O master 1 initialization error				
SM252	Remote I/O master 2 initialization error				
SM253	Remote I/O master 3 initialization error				
SM254	Remote I/O master 4 initialization error				
SM255	Remote I/O master 5 initialization error				
SM256	Remote I/O master 6 initialization error				
SM257	Remote I/O master 7 initialization error				
SM258 | SM25F	Not used			-	-
SM260	Processor-link 0 initialization error	Set to "ON" when an error is found in a P/PE-link/FL-net initialization. Processor-link 0 corresponds to the module of link No. 8. Processor-link 1 corresponds to the module of link No. 9. Processor-link 2 corresponds to the module of link No. 10. Processor-link 3 corresponds to the module of link No. 11. Processor-link 4 corresponds to the module of link No. 12 Processor-link 5 corresponds to the module of link No. 13. Processor-link 6 corresponds to the module of link No. 14. Processor-link 7 corresponds to the module of link No. 15.	Fatal fault	O	O
SM261	Processor-link 1 initialization error				
SM262	Processor-link 2 initialization error				
SM263	Processor-link 3 initialization error				
SM264	Processor-link 4 initialization error				
SM265	Processor-link 5 initialization error				
SM266	Processor-link 6 initialization error				
SM267	Processor-link 7 initialization error				
SM268 | SM29F	Not used			-	-

Note: The system definition error factor includes errors that do not occur during normal operation.

12) Application program error factor WSM38(26), WSM39(27)

Address	Name	Description	Level	SPH 300	SPH 200
SM380	Application WDT error	Set to "ON" when the run time for a default task exceeds the preset value of a watchdog timer.	Fatal fault	O	O
SM381	Application execution error	Set to "ON" when an error has occurred during user program execution that causes "temporary size-over."	Fatal fault	O	O
SM382 | SM38A	Not used			-	-
SM38B	FB instance setup error	Set to "ON" when the specified storage address is not found.	Fatal fault	O	O
SM38C	Initial setup value error	Set to "ON" when the preset initial value exceeds the defined range of a storage area.	Fatal fault	O	O
SM38D	SFM boundary definition error	Set to "ON" when a size greater than the maximum SFM capacity value has been defined.	Fatal fault	O	O
SM38E	Program instruction error	Set to "ON" when an error has been found in the programs instruction.	Fatal fault	O	O
SM38F	Task registration error	Set to "ON" when a task registration error has been found.	Fatal fault	O	O
SM390	Missing level 0 task	Set to "ON" when a task is missing. May clear the error condition by using an application program.	Nonfatal fault	O	O
SM391	Missing level 1 task				
SM392	Missing level 2 task				
SM393	Missing level 3 task				
SM394	Level 0 task slow-down	Set to "ON" when program execution is deferred and the predefined periodic time is not maintained. May clear the error condition by using an application program.	Nonfatal fault	O	O
SM395	Level 1 task slow-down				
SM396	Level 2 task slow-down				
SM397	Level 3 task slow-down				
SM398 | SM39E	Not used			-	-
SM39F	Takt period monitoring error	Set to "ON" when the Takt period does not match the system definition. May clear the error condition by using an application program.	Nonfatal fault	O	O

Note: The system definition error factor includes errors that do not occur during normal operation because the loader suppresses them from occurring (for example, compile check).

13) Annunciator relay WSM42(2A), SM43(2B)

Address	Name	Description	SPH 300	SPH 200
SM420	Initial flag	Set to "ON" at the first startup after program download and at initial startup (cold start). This flag never is set to "OFF" during operation.	O	O
SM421	Power-off flag	Set to "ON" when a power-off condition has occurred in the preceding session.	O	O
SM422 | SM42D	Not used		-	-
SM42E	Dummy module flag	Set to "ON" when more than one dummy module has been installed in one SPH system.	O	O
SM42F	Processor bus access disable flag	Set to "ON" when the processor bus is disabled.	O	O
SM430	Level 0 start flag	Set to "ON" during the first execution of level 0 task.	O	O
SM431	Level 1 start flag	Set to "ON" during the first execution of level 1 task.	O	O
SM432	Level 2 start flag	Set to "ON" during the first execution of level 2 task.	O	O
SM433	Level 3 start flag	Set to "ON" during the first execution of level 3 task.	O	O
SM434 | SM43E	Not used		-	-
SM43F	Default task start flag	Set to "ON" during the first execution of default task.	O	O

14) Redundant annunciator relay WSM46(2E), Redundant operation mode WSM47(2F) (Read only) (Not supported by SPH200)

Address	Name	Description
SM460	Redundant continuation start flag	Set to "ON" during operation in the redundant mode and when the operating system is switched to a waiting one. (The CPU that is changed over from working to standby mode)
SM461 | SM46F	Not used	
SM470 | SM473	Redundant logical CPU number	Indicates a 4-bit CPU logical number in the redundant operation mode (0-7). Allows you to recognize the default working CPU taken over by the standby CPU. It is undefined in the mode other than as "redundant."
SM474 | SM477	Not used	
SM478	Redundant annunciator relay mode 0	Set to "ON" when an annunciator relay has been enabled for a pair of CPUs 0 and 1 during operation in the 1-to-1 redundant mode.
SM479	Redundant annunciator relay mode 1	Set to "ON" when an annunciator relay has been enabled for a pair of CPUs 2 and 3 during operation in the 1-to-1 redundant mode.
SM47A	Redundant annunciator relay mode 2	Set to "ON" when an annunciator relay has been enabled for a pair of CPUs 4 and 5 during operation in the 1-to-1 redundant mode.
SM47B	Redundant annunciator relay mode 3	Set to "ON" when an annunciator relay has been enabled for a pair of CPUs 6 and 7 during operation in the 1-to-1 redundant mode.
SM47C | SM47F	Not used	

15) Resource configuration/operation information WSM48(30), WSM49(31) (Read only) (only for SPH300)

The information can be used to recognize the current status of system (CPU module) operation in the redundant or single mode. Resource configuration information can be used only in the redundant mode.
The types of status listed below are valid when the associated bits (WSM50(32), WSM51(33)) with resource operation/fault information have been set to “ON.”

<In the redundant mode>

Resource operation information	Resource running information	Resource status
OFF	OFF	Waiting CPU stopped
ON	OFF	Operating CPU stopped
ON	ON	Operating CPU running
OFF	ON	Waiting CPU running

<Resource operation information>

Address	Name	Description
SM480	Operating CPU 0 running	Set to "ON" when the operating CPU is running in the redundant mode. (in a mode other than "redundant," undefined)
SM481	Operating CPU 1 running	
SM482	Operating CPU 2 running	
SM483	Operating CPU 3 running	
SM484	Operating CPU 4 running	
SM485	Operating CPU 5 running	
SM486	Operating CPU 6 running	
SM487	Operating CPU 7 running	
SM488 | SM48F	Not used	

<Resource runnig information>

Address	Name	Description
SM490	CPU 0 running	Set to "ON" when the associated CPU module connected to the SX bus is running.
SM491	CPU 1 running	
SM492	CPU 2 running	
SM493	CPU 3 running	
SM494	CPU 4 running	
SM495	CPU 5 running	
SM496	CPU 6 running	
SM497	CPU 7 running	
SM498 | SM49F	Not used	

16) Resource configuration/fault information WSM50(32), WSM51(33) (Read only)

The information can be used to recognize the status of other resources (CPU module) by using an application program.

Resource configuration information	Resource fault information	Resource status
OFF	OFF	Nonexistent
ON	OFF	Normal (running or stopped)
ON	ON	Nonfatal fault (running or stopped)
OFF	ON	Fatal fault (stopped or dropped)

<Resourse configuration information>

Address	Name	Description	SPH 300	SPH 200
SM500	CPU 0 configuration	Set to "ON" when the associated CPU module connected to the SX bus has been found and its resource operation status is "normal" or "Non-fatal fault." Note: For SPH200, only CPU0 is the target.	O	O
SM501	CPU 1 configuration			
SM502	CPU 2 configuration			
SM503	CPU 3 configuration			
SM504	CPU 4 configuration			
SM505	CPU 5 configuration			
SM506	CPU 6 configuration			
SM507	CPU 7 configuration			
SM508 | SM50F	Not used		-	-

<Resourse fauit information>

Address	Name	Description	SPH 300	SPH 200
SM510	CPU 0 error	Set to "ON" when the associated CPU module connected to the SX bus has been found and its resource operation status is "normal" or "Non-fatal fault." Note: For SPH200, only CPU0 is the target.	O	O
SM511	CPU 1 error			
SM512	CPU 2 error			
SM513	CPU 3 error			
SM514	CPU 4 error			
SM515	CPU 5 error			
SM516	CPU 6 error			
SM517	CPU 7 error			
SM518 | SM51F	Not used		-	-

17) SX bus configuration information WSM52(34) to wsm67(43) (Read only)

When a module exists on the SX bus and it is running normally or with a nonfatal fault, the SX station number bit for the module is set to “ON.”
Whether the module is normal or in a nonfatal fault is identified by the combination of the configuration error information items.

SX bus configuration information	SX bus fault information	Module status
OFF	OFF	Nonexistent
ON	OFF	Normal
ON	ON	Nonfatal fault
OFF	ON	Fatal fault or dropped

Word Address ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←Bit Address
WSM52	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1		←Not used
WSM53	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM54	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM55	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM56	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM57	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM58	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM59	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	
WSM60	143	142	141	140	139	138	137	136	135	134	133	132	131	130	129	128	
WSM61	159	158	157	156	155	154	153	152	151	150	149	148	147	146	145	144	
WSM62	175	174	173	172	171	170	169	168	167	166	165	164	163	162	161	160	
WSM63	191	190	189	188	187	186	185	184	183	182	181	180	179	178	177	176	
WSM64	207	206	205	204	203	202	201	200	199	189	197	196	195	194	193	192	
WSM65	223	222	221	220	219	218	217	216	215	214	213	212	211	210	209	208	
WSM66	239	238	237	236	235	234	233	232	231	230	229	228	227	226	225	224	
WSM67		254	253	252	251	250	249	248	247	246	245	244	243	242	241	240	

18) SX bus fault information WSM68(44) to WSM83(53) (Read only)

When there is a module on the SX bus and it is subject to a fatal or nonfatal fault, the bit corresponding to the SX bus station number of the module is set to “ON.”

Word Address ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←Bit Address
WSM68	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1		←Not used
WSM69	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM70	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM71	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM72	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM73	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM74	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM75	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	
WSM76	143	142	141	140	139	138	137	136	135	134	133	132	131	130	129	128	
WSM77	159	158	157	156	155	154	153	152	151	150	149	148	147	146	145	144	
WSM78	175	174	173	172	171	170	169	168	167	166	165	164	163	162	161	160	
WSM79	191	190	189	188	187	186	185	184	183	182	181	180	179	178	177	176	
WSM80	207	206	205	204	203	202	201	200	199	189	197	196	195	194	193	192	
WSM81	223	222	221	220	219	218	217	216	215	214	213	212	211	210	209	208	
WSM82	239	238	237	236	235	234	233	232	231	230	229	228	227	226	225	224	
WSM83		254	253	252	251	250	249	248	247	246	245	244	243	242	241	240	

19) SX bus-connected module fail-soft information WSM84(54) to WSM99(63) (Read only)

When fail-soft or individual reset cannot be done for any of the modules connected to the SX bus, the SX station number bit for the module is set to “ON.”

Word Address ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←Bit Address
WSM84	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1		←Not used
WSM85	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM86	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM87	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM88	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM89	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM90	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM91	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	
WSM92	143	142	141	140	139	138	137	136	135	134	133	132	131	130	129	128	
WSM93	159	158	157	156	155	154	153	152	151	150	149	148	147	146	145	144	
WSM94	175	174	173	172	171	170	169	168	167	166	165	164	163	162	161	160	
WSM95	191	190	189	188	187	186	185	184	183	182	181	180	179	178	177	176	
WSM96	207	206	205	204	203	202	201	200	199	189	197	196	195	194	193	192	
WSM97	223	222	221	220	219	218	217	216	215	214	213	212	211	210	209	208	
WSM98	239	238	237	236	235	234	233	232	231	230	229	228	227	226	225	224	
WSM99		254	253	252	251	250	249	248	247	246	245	244	243	242	241	240	

20) Remote I/O master 0 I/O module configuration/fault information WSM128(80) to WSM143(8F) (Read only)

When there is a module that is under remote I/O master 0 control and it is normal or in a nonfatal fault, the SX station number bit for the pertinent module is set to “ON.”

Remote configuration information	Remote fault information	Module status
OFF	OFF	Does not exist
ON	OFF	Normal
ON	ON	Nonfatal fault
OFF	ON	Fatal fault or disconnected

<Configuration information>

Word Address ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←Bit Address
WSM128	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM129	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM130	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM131	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM132	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM133	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM134	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM135	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

When there is a module that is under remote I/O master 0 control and it is in a fatal or nonfatal fault, the bit corresponding to the remote station number of the module is set to “ON.”

<Fault information>

Word Address ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←Bit Address
WSM136	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM137	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM138	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM139	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM140	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM141	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM142	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM143	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

The interpretation of paragraphs 21) through 27) and 29) is identical to that of paragraph 20).

21) Remote I/O master 1 I/O module configuration/fault information WSM144(90) to WSM159(9F) (Read only)

<Configuration information>

Word Address ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←Bit Address
WSM144	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM145	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM146	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM147	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM148	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM149	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM150	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM151	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

<Fault information>

WSM152	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSM153	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSM154	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
WSM155	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
WSM156	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64
WSM157	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
WSM158	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
WSM159	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112

22) Remote I/O master 2 I/O module configuration/fault information WSM160(A0) to WSM175(AF) (Read only)

<Configuration information>

Word Address ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←Bit Address
WSM160	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
WSM161	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM162	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM163	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM164	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM165	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM166	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM167	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	

<Fault information>

WSM168	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSM169	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSM170	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
WSM171	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
WSM172	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64
WSM173	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
WSM174	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
WSM175	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112

23) Remote I/O master 3 I/O module configuration/fault information WSM176(B0) to WSM191(BF) (Read only)

<Configuration information>

←Bit Address

Word Address ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
WSM176	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSM177	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSM178	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
WSM179	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
WSM180	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64
WSM181	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
WSM182	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
WSM183	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112

<Fault information>

WSM184	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSM185	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSM186	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
WSM187	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
WSM188	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64
WSM189	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
WSM190	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
WSM191	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112

24) Remote I/O master 4 I/O module configuration/fault information WSM192(C0) to WSM207(CF) (Read only)

<Configuration information>

←Bit Address

Word Address ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
WSM192	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSM193	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSM194	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
WSM195	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
WSM196	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64
WSM197	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
WSM198	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
WSM199	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112

<Fault information>

WSM200	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSM201	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSM202	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
WSM203	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
WSM204	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64
WSM205	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
WSM206	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
WSM207	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112

25) Remote I/O master 5 I/O module configuration/fault information WSM208(D0) to WSM223(DF) (Read only)

<Configuration information>

Word Address ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0 ←Bit Address
WSM208	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSM209	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSM210	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
WSM211	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
WSM212	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64
WSM213	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
WSM214	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
WSM215	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112

<Fault information>

WSM216	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSM217	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSM218	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
WSM219	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
WSM220	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64
WSM221	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
WSM222	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
WSM223	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112

26) Remote I/O master 6 I/O module configuration/fault information WSM224(E0) to WSM239(EF) (Read only)

<Configuration information>

Word Address ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0 ←Bit Address
WSM224	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSM225	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSM226	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
WSM227	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
WSM228	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64
WSM229	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
WSM230	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
WSM231	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112

<Fault information>

WSM232	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSM233	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSM234	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
WSM235	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
WSM236	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64
WSM237	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
WSM238	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
WSM239	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112

27) Remote I/O master 7 I/O module configuration/fault information WSM240(F0) to WSM255(FF) (Read only)

<Configuration information>

Word Address ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0 ←Bit Address
WSM240	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSM241	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSM242	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
WSM243	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
WSM244	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64
WSM245	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
WSM246	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
WSM247	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112

<Fault information>

WSM248	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSM249	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSM250	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
WSM251	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
WSM252	79	78	76	76	75	74	73	72	71	70	69	68	67	66	65	64
WSM253	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
WSM254	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
WSM255	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112

28) Fail-soft maintenance operation prohibition mode / Error state display hiding mode / Compulsion setting hold state / System start watch time WMS256(100) (Read only) (With V63 or later software version of SPH300)

- **Fail-soft maintenance operation prohibition mode (With V63 or later software version of SPH300)**
  If this mode is selected, additional connection and recconection of a module or unit directly connected to the SX bus are not performed even when fail-soft start-up is set for the SPH system.

- **Error state display hiding mode (With V63 or later software version of SPH300)**
  When the SPH system is placed in fail-soft start up mode and a station to be fail-soft started does not exist, the SPH system starts its operation in a nonfatal fault state. In this case, ALM LED of the CPU stays on.
  Even if a station with fail-soft enabled is disconnected under these conditions, the state of the LED does not change, therefore, it is not possible to know the condition of the system by the state of LED.
  If this mode is selected, even when a station to be fail-soft started does not exist at system start-up, it is not regarded as a nonfatal fault. Therefore, it becomes possible to know disconnection of an existing station with fail-soft enabled by the ALM LED.

- **Compulsion setting hold state**
  In this operation mode, the compulsion setting for I/O is held.

- **System start watch time**
  If the system is placed in fail-soft start-up mode, the system start watch time is indicated.

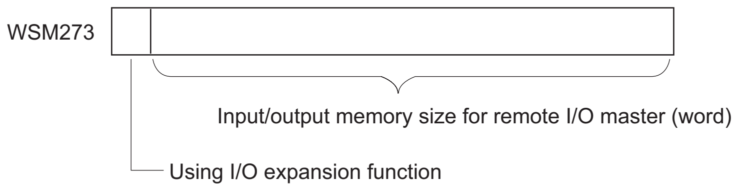
Address	Name	Description
SM2560	Partial fail-soft start up mode	Set to "ON" when the system is placed in partial fail-soft start up mode.
SM2561 | SM2564	Not used	
SM2565	System start watch time	ON: Watch time is set at 10s. OFF: Watch time is set at 30s. (default)
SM2566	Fail-soft maintenance operation prohibition mode	Set to "ON" when the system is placed in partial fail-soft start up mode and fail-soft maintenance operation prohibition mode.
SM2567	Error state display hiding mode	Set to "ON" when the system is placed in partial fail-soft start up mode and error state display hiding mode.
SM2568 | SM256D	Not used	
SM256E	Compulsion setting hold state operation mode	ON: The compulsion setting is held at start-up. OFF: The compultion setting is cleared at start-up.
SM256F	Compulsion setting hold state	ON: There is compulsion setting to hold. OFF: There is no compulsion setting to hold.

29) Station No. with error state display hidden WMS257 (101) to WSM272 (110) (Read only)
(With V63 or later software version of SPH300)

If error state display hiding mode is selected and a concerned station exists, a bit corresponding to the station No. is set to ON.

Word Address ↓	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	←Bit Address
WSM257	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1		←Not used
WSM258	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
WSM259	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32	
WSM260	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	
WSM261	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64	
WSM262	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80	
WSM263	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	
WSM264	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112	
WSM265	143	142	141	140	139	138	137	136	135	134	133	132	131	130	129	128	
WSM266	159	158	157	156	155	154	153	152	151	150	149	148	147	146	145	144	
WSM267	175	174	173	172	171	170	169	168	167	166	165	164	163	162	161	160	
WSM268	191	190	189	188	187	186	185	184	183	182	181	180	179	178	177	176	
WSM269	207	206	205	204	203	202	201	200	199	189	197	196	195	194	193	192	
WSM270	223	222	221	220	219	218	217	216	215	214	213	212	211	210	209	208	
WSM271	239	238	237	236	235	234	233	232	231	230	229	228	227	226	225	224	
WSM272		254	253	252	251	250	249	248	247	246	245	244	243	242	241	240	

30) I/O expansion mode status WMS273 (111) (Read only) (With V64 or later software version of SPH300)



- Using I/O expansion function
  This is set to ON if the target system is operating in I/O expansion mode.
- Input/output memory size for remote I/O master
  The input/output memory size of a remote I/O system that occupies the largest number of words for input/output memory among multiple remote I/O systems in a SPH system is indicated in units of words.

31) Ethernet interface information WSM440(1B8) to WSM445(1BD) (Read only)

The MAC address, IP address, and loader command port of the Ethernet interfac part are indicated.

Address	Name	Description
WSM440	MAC address (H)	Indicates the MAC address set in the Ethernet built into the CPU. The MAC address is set to a fixed address before shipment.
WSM441	MAC address (M)	
WSM442	MAC address (L)	
WSM443	IP address (H)	Indicates the IP address set in the Ethernet built into the CPU. The IP address is set to "192.168.0.1." before shipment. This can be changed in the system definition.
WSM444	IP address (L)	
WSM445	Loader command port No.	Indicates the loader command port No. Fixed to "507".

* For NP1PM-48E and NP1PM-256E only

32) SX bus transmission error rate information WSM508(1FC) to WSM511(1FF) (Read only)

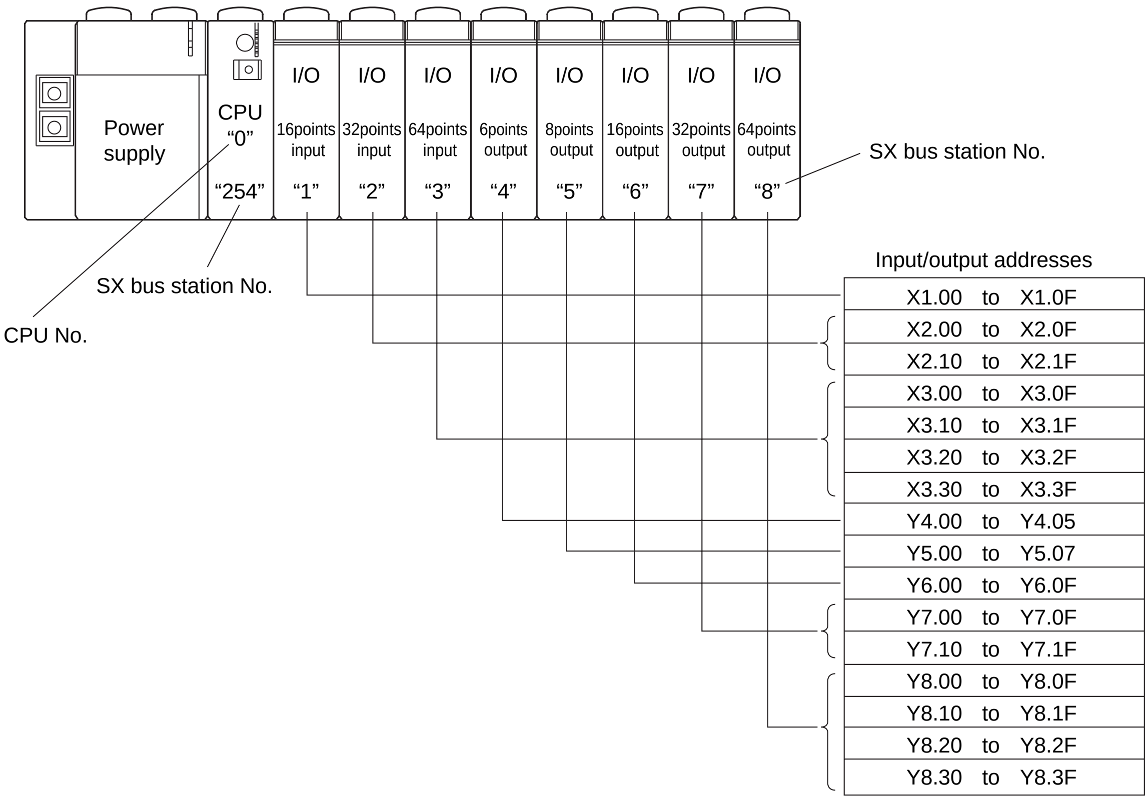
Executing 100,000 Takts, the number of the Takts where SX bus error occurred is expressed by the ppm. If, of executed Takts, even one Takt is erroneous, the value becomes “10.” These data are updated every 100,000 Takts.

Address	Name	Description
WSM508 WSM509	Maximum value (lower word) Maximum value (upper word)	Set to the maximum of the error rate values for the SX bus detected by self-CPU module.
WSM510 WSM511	Current value (lower word) Current value (upper word)	Set to the current of the error rate values for the SX bus detected by self-CPU module.

Note: Various types of system flag information for system memory areas may be referenced from within an application program. Be sure not to use the information for “event variables,” which start the event tasks of an application program (otherwise, some variables may not start the associated task).

2-3-1 Address assignment example

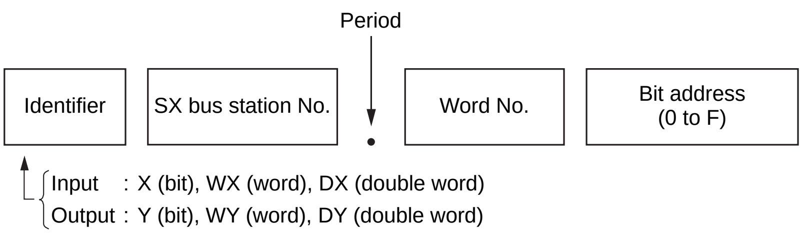
This subsection shows an example of address assignment. The sample system configuration is illustrated below.



2-3-2 Address assignment conventions

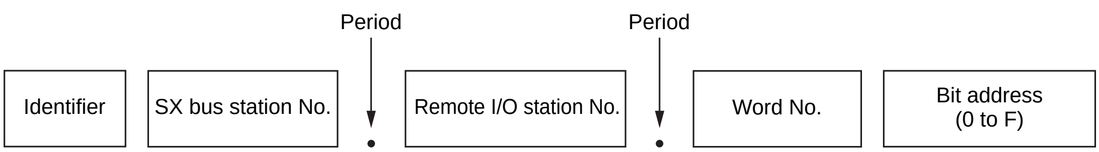
The following conventions are used to assign input/output addresses to the MICREX-SX SPH series CPU modules:

1) SX bus direct-connect I/O



2) Remote I/O

For input/output on remote I/O's such as T-link, OPCN-1, DeviceNet are addressed as shown below.



2-4 Array and Structure

2-4-1 Array

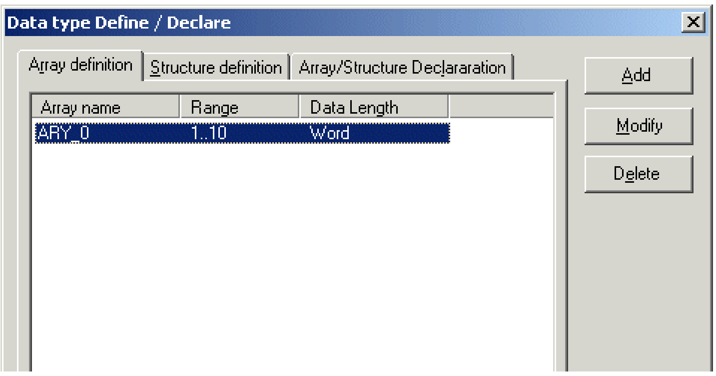
An array is made up of two or more elements of the same data length.

(1) Sample definition and declaration of 1-dimensional array

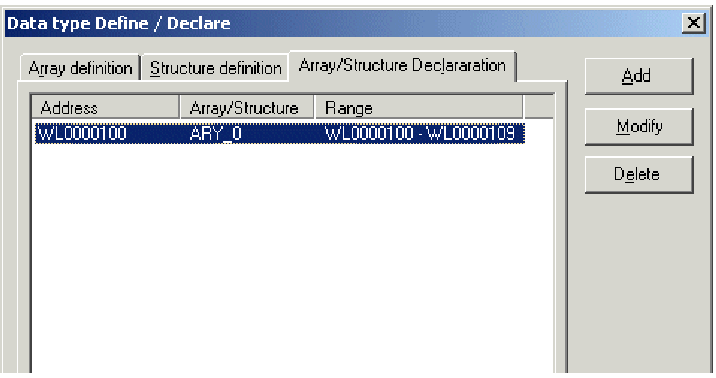
An example of array data with 10 elements of 16-bit data is shown below.

Element No. ↓	
1	16-bit data
2	16-bit data
3	16-bit data
4	16-bit data
5	16-bit data
6	16-bit data
7	16-bit data
8	16-bit data
9	16-bit data
10	16-bit data

<Sample data definition>
Sample definition of array data with 10 elements of 16-bit data is shown below. The element No. starts with 1.



<Sample data declaration>
Assign a defined array or structure to an actual address. This operation is called "declaratoin".
If the array data defined in the data definition above is assigned to WL100, it becomes as shown below.



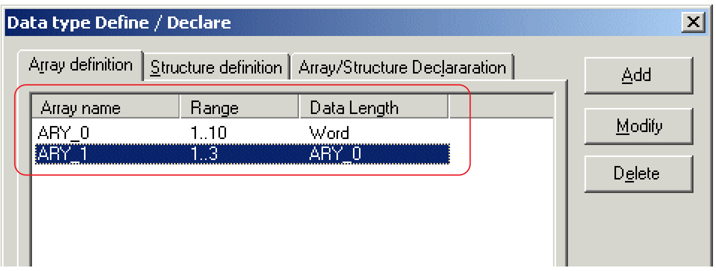
* In the "Range" section, address range to which array data is assigned is displayed.

(2) Sample definition and declaration of array of arrays

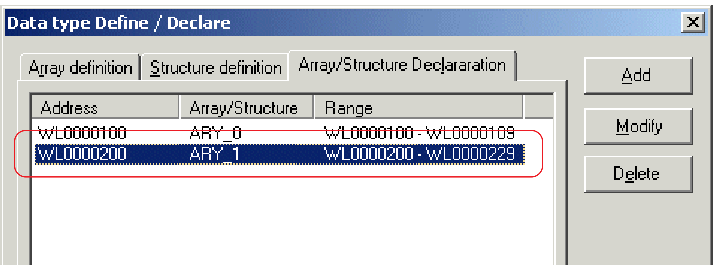
An example of “array of arrays” with 10 rows and 3 columns of 16-bit data is shown below.

Element No. ↓	1	2	3
1	16-bit data	16-bit data	16-bit data
2	16-bit data	16-bit data	16-bit data
3	16-bit data	16-bit data	16-bit data
4	16-bit data	16-bit data	16-bit data
5	16-bit data	16-bit data	16-bit data
6	16-bit data	16-bit data	16-bit data
7	16-bit data	16-bit data	16-bit data
8	16-bit data	16-bit data	16-bit data
9	16-bit data	16-bit data	16-bit data
10	16-bit data	16-bit data	16-bit data

<Sample data definition>
Sample definition of “array of arrays” with 10 rows and 3 columns of 16-bit data is shown below.



<Sample data declaration>
If the array data defined in the data definition above is assigned to WL200, it becomes as shown below.



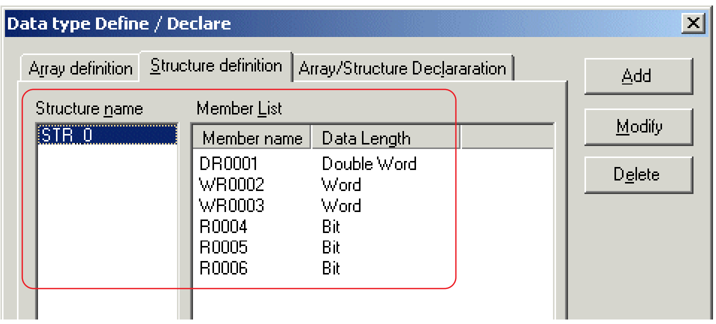
* In the “Range” section, address range to which array data is assigned is displayed.

2-4-2 Structure

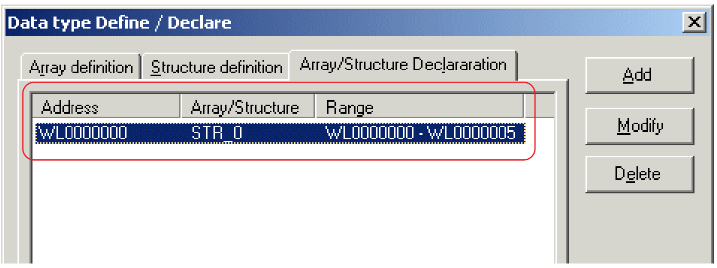
Structured data type is made up of two or more similar or different data types (members).

(1) Sample definition and declaration of structure

<Sample data definition>
The following figure shows sample data definition of structure data with the following members: one piece of 32-bit data, two pieces of 16-bit data and three pieces of bit data.



<Sample data declaration>
If the structure data defined in the data definition above is assigned to WL0, it becomes as shown below.

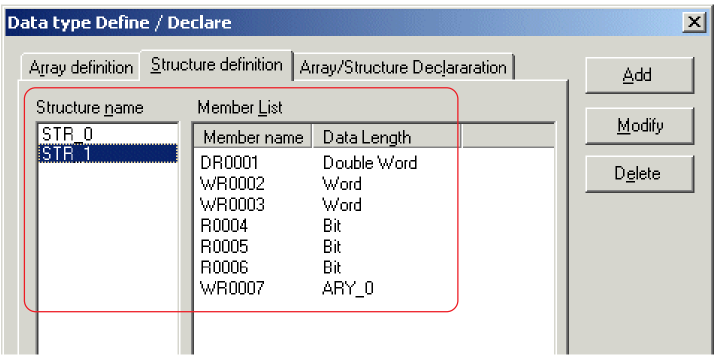


* In the “Range” section, address range to which array data is assigned is displayed. However, details (which member is assigned to which address) are not displayed.

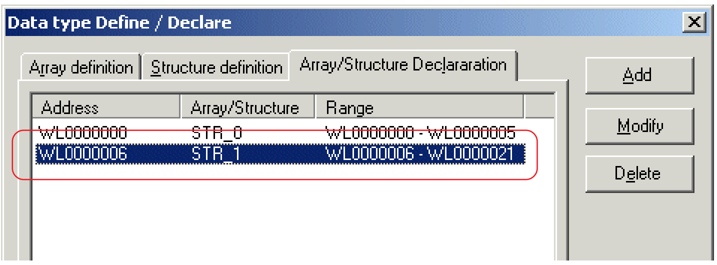
(2) Sample definition and declaration of structure of arrays

“Structure of arrays” is structure data that contains array data as a member of structure.

<Sample data definition>
The following figure shows sample data definition of structure data with the following members: one piece of 32-bit data, two pieces of 16-bit data, three pieces of bit data and one piece of array data.



<Sample data declaration>
If the structure data defined in the data definition above is assigned to WL6, it becomes as shown below.



* In the “Range” section, address range to which array data is assigned is displayed. However, details (which member is assigned to which address) are not displayed.

2-4-3 Specifications of array/structure

(1) Specifications

Item		Specification	
Nesting depth		1 level (array of arrays, structure of arrays)	
Array	Max. No. of difinitions	255 (See note 1.)	
	Max. No. of elements	32767	
	Element No. specification	1 to specified No. of elements No. of elements: max. 32767 * When element No. specification exceeds the range, upper limit or lower limit data is accessed.	
	Max. No. of declarations	583 (See note 1.) * "Declaration" means assigning a defined array to an address.	
	Data type	Bit, word, double word, array (defined by user)	
	Array name	ARY_n n : array definition No. (from 0 (decimal))	
	Available memories	Entire program:	I/O memory (W, Y), standard memory (M), retain memory (L)
		Inside of user FB:	I/O memory (X, Y), standard memory (M), retain memory (L), memory for user FB (F), parameter (V), (I/O parameter only)
Structure	Max. No. of difinitions	255 (See note 1.)	
	Max. No. of members	255	
	Data type of member	Bit, word, double word, array (defined by user)	
	Structure name	STR_n n : structure definition No. (from 0 (decimal))	
	Member name specification	Member name identifying sign + member definition No. (from 1) * The member name definition No. is automatically assigned when the structure is defined. Member name identifying sign: R (bit), WR (word), DR (double word)	
	Max. No. of declarations	584 (When No. of members: 1 and No. of structure definitions: 1) (See note 1.) * "Declaration" means assigning a defined structure to an address.	
	Available memories (See note 2.)	Entire program:	I/O memory (W, Y), standard memory (M), retain memory (L)
		Inside of user FB:	I/O memory (X, Y), standard memory (M), retain memory (L), Memory for user FB (F), parameter (V), (I/O parameter only)

Notes: 1) Definitions and declarations of arrays and structures are stored in the program memory of the CPU. The amount of program memory used by arrays and structures is obtained by the following equation.

[Equation]

(Program memory used by arrays and structures)
= 4 + (No. of array definitions) x 5 + (Structure 0 No. of steps of definition) + ...
+ (Structure n No. of steps of definition) + (No. of declarations of arrays and structures) x 7
* No. of steps of structure definition = (No. of members) + 3 (steps)

[Sample calculation]

Supposing,
- No. of array definitions: 3,
- Structure 0 (No. of members: 10)
- Structure 1 (No. of members: 5)
- No. of declarations of arrays and structures: 5

then,
(Program memory used by arrays and structures) = 4 + (3 x 5) + (10 + 3) + (5 + 3) + (5 x 7) = 75 steps

2) The system memory (SM), timer (T), counter (C), integrating timer (TR) and step control (SC) are not available.

3) Information about definitions and declarations of arrays and structures is stored in the user function No. 511 and then in the program memory of the CPU. Therefore, when using arrays and structures, the user function No. 511 cannot be used.

(2) Representation in programs

Arrays and structures are represented as shown below.

[Array]

WM100 [5]

- WM100: Top address
- [5]: Array element No.

[Structure]

WM100. WR2

- WM100: Top address
- WR2: Member name

[Array of arrays]

WM100 [5] [WM0]

- WM100: Top address
- [5]: Array element No.
- [WM0]: Array element No.

[Structure of arrays]

WM100. WR2 [3]

- WM100: Top address
- WR2: Member name
- [3]: Array element No.

2-5 Tasks

A task is a time schedule for program execution.
Tasks determine the sequence (time schedule) of program execution. The MICREX-SX series CPU modules use three types of tasks: default task for cyclic processing, periodic tasks, and event tasks.
Programs that will always be executed need be assigned to tasks so that their execution sequence can be determined.

2-5-1 Task specifications

Item	Specification
Task type	Default task (cyclic processing) Periodic task Event task
Number of tasks	1 (default) + 4 (periodic and event tasks)
Task priority	0 > 1 > 2 > 3 > default

2-5-2 Types and operations of tasks

1) Default task

- Always repeat execution in synchronization with Takt. Assign Programs requiring no responsibility and periodicity in arithmetical operations.
- Two or more Programs may be assigned to the default task.

Note: A user WDT is the timer which monitors the execution time of the default task. It checks the time when execution has be done. When no default task is used, the CPU processes the tasks equivalent to the default ones to execute internal processes such as the user WDT check.

2) Periodic task

- A periodic interrupt task is executed once at a predetermined interval (0.5ms, 1ms to 32s). It is assigned to programs and filters that require high responsiveness to adjust to the speed of the control target and programs such as integral instructions which need be executed at predetermined intervals.
- A period interrupt task is given a priority of 0 to 3 (0 has the highest priority).
- Multiple programs can be assigned to one task.
- Two or more programs can be assigned to a periodic interrupt task (only for SPH).
  When SPS is used, fixed cycle task asynchronously interrupts the currently executed task.

Note: A Takt period is an SX bus communication period. For the Takt period, 0.5ms, 1ms, ..., and 10ms may be set. Note that when a standard CPU is used, 0.5ms cannot be specified. The Takt period depends on the scale of system configuration (the numbers of I/O points, remote I/O master modules, communication modules, and CPU modules) and the number of application program executable steps. A 0.5ms Takt period may be executed under the condition of a powerful single CPU, 256 or less I/O modules connected to the SX bus and no remote I/O and communication modules. In the standard CPU system, setting a Takt period to 0.5ms causes “system operation definition error” to occur, resulting in CPU shutdown with a fatal fault. (Refer to the appropriate appendix for calculating a Takt period.)

3) Event task

- An event task is executed once each time a specified bit device turns to “1” It is assigned to a program that handles interrupts from a communications module or high-speed counter module.
- Two or more programs can be assigned to an event task.

2-5-3 Example of periodic task operation

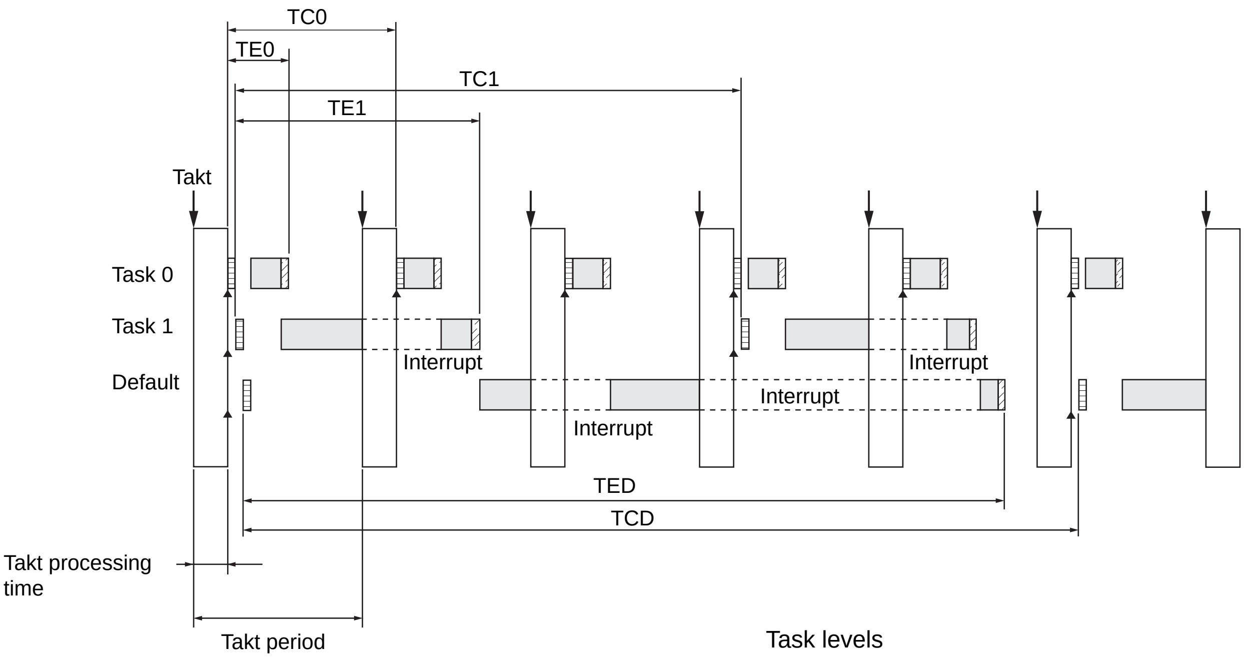
Example of the operation of fixed task when SPH is used is explained under the following operating conditions.

<Operating conditions>

Task priority ⇨ Task 0 > Task 1 > Default task (cyclic)
Takt period ⇨ 1 ms
Task type ⇨ Task 0 : Fixed task (1 Takt period)
Task 1 : Fixed task (3 Takt periods)
Default task : Asynchronous with the Takt period

Note: The task period of the fixed tasks must be set to an integral multiple of the SX bus Takt period.

<Task operation>



Task levels
Task 0 > Task 1 > Default

TE0: Task 0 execution time

TC0: Task 0 execution period

TE1: Task 1 execution time

TC1: Task 1 execution period

TED: Default task execution time

TCD: Default task execution period

▲ : Task activation request

: Data input processing

: Data output processing

The default tasks run while no fixed task or event task is operating. (They start in synchronization with a Takt period.) Be sure to adjust the run times to the start periods of upper tasks so that the run times may be reserved for default tasks. (Otherwise, a user WDTUP or upper task may be delayed.)

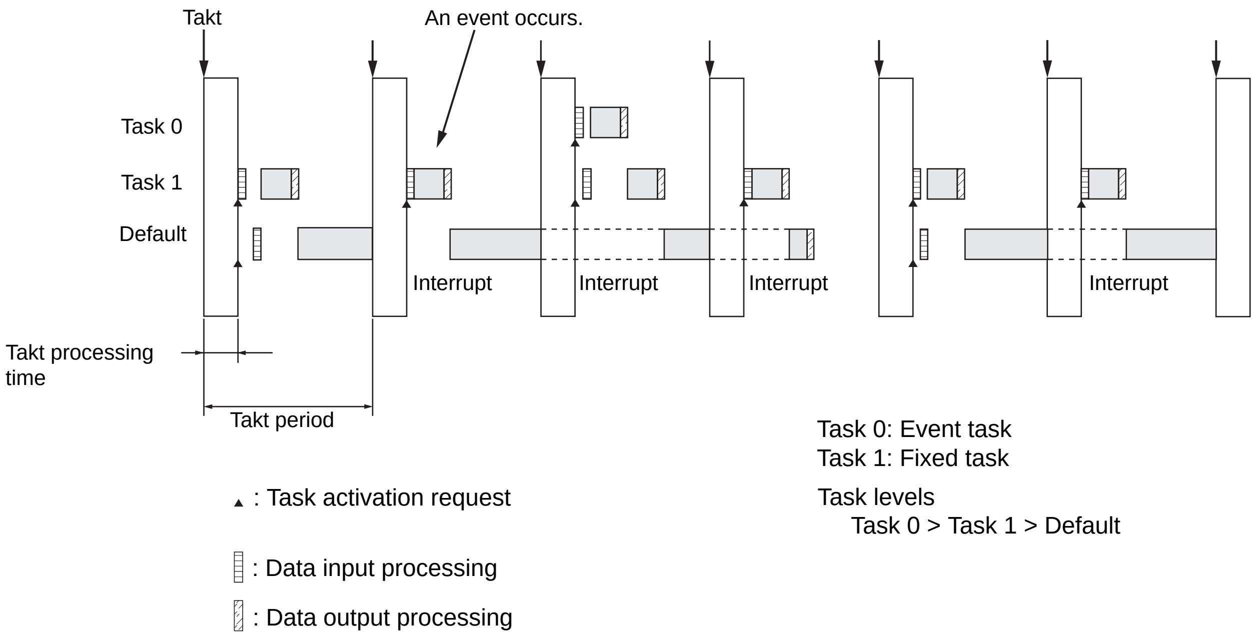
2-5-4 Example of event task operation

An example of operation of an event task is explained below.

<Operating conditions>

Task priority ➩ Task 0 > Task 1 > Default task (cyclic)
Takt period ➩ 1 ms
Task type ➩ Task 0: Event task
Task 1: Fixed task (1 Takt period)
Default task : Asynchronous with the Takt period

<Task operation>



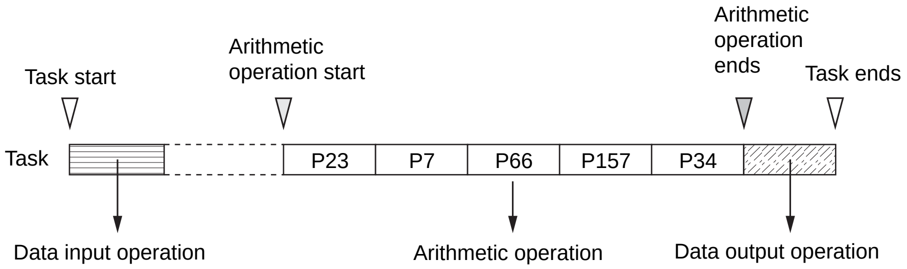
An event task is not started immediately when an event occurs but at the beginning of the next Takt period after the event is recognized.

Monitoring the run times and run periods of tasks
The run time and run period of a task may be monitored on the resource information screen displayed from the "Loader Resource Control" dialog box.
Task run time: The time after input to the task starts until output from the task has been finished.
Task run period: The time after input to the task starts until input to the next task starts.

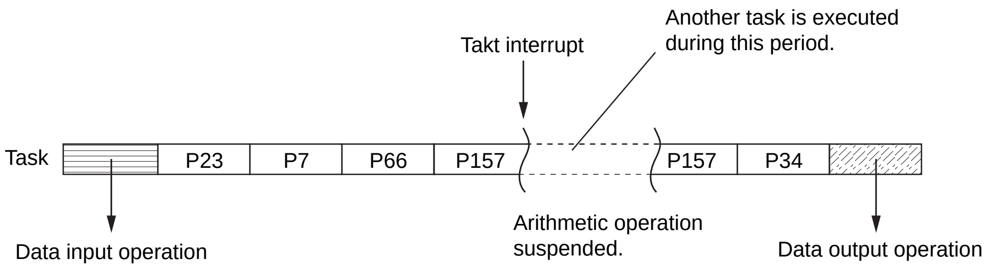
The task run period appears on the open PLC information screen after the task is executed two times. When the CPU stops/starts or the CPU is switched between the operating and waiting sides in the redundant mode while the PLC information screen is open, measurement stops temporarily and then restarts.
Before the run period can appear on the screen, the task must have been executed two times after the CPU stopped/started or was switched between the operating and waiting sides.

2-5-5 Task interrupt processing

The processing of a task is divided into data input processing, arithmetic operation, and data output processing as illustrated below. A task operation sequence is considered to terminate when all of these operations have completed.
Takt interrupts can be generated via the SX bus during the arithmetic and data output operations (no Takt interrupt can occur during the data input operation).



When a Takt interrupt occurs during the arithmetic operation, the operation is interrupted as illustrated in the figure below.
When a Takt interrupt occurs, the system checks the startup conditions for a new task. If a task startup request is present, the system performs the data input operation for that task and starts the task having the highest priority. Consequently, another task is likely to be executed while the current task is suspended as shown in the figure below.
When the arithmetic operation ends, the system checks the time until the next Takt interrupt occurs and, if it is longer than the time required to perform the data output operation, performs the data output processing and terminates the task. If the time to the next Takt interrupt is shorter than the time required to perform the data output operation, the system keeps the task suspended and executes no output operation for the task. The data output operation is carried out only after the system executes the next Takt processing. Since the time up to the next Takt interrupt is computed at the end of the arithmetic operation, no Takt interrupt can occur during the execution of the data output operation.



2-6 Program Types

Programs are classified into programs, FCT and FBs.

- Programs
  - Programs — Created by the user and allocated to tasks.
  - Functions (FCTs)
    - System FCTs → FCTs owned by MICREX-SX
    - User FCTs → FCTs created by the user
  - Function Blocks (FBs)
    - System FBs → FBs owned by MICREX-SX
    - User FBs → FBs created by the user
    - Expansion FBs → FBs provided by Fuji Electric FA Components & Systems Co., Ltd.

<About calling functions and FBs>

Types of caller programs	Types of programs that can be called
Programs	System FCTs, user FCTs, system FBs, user FBs and expansion FBs
User FBs	System FCTs, user FCTs, system FBs, user FBs and expansion FBs
User FCTs	System FCTs and user FCTs

* For the procedures for creating user FBs, refer to “Appendix 4 Procedures for Creating FBs”.

2-7 Calendar Function

The MICREX-SX series CPU modules incorporate a clock that provides calendar functions. The values of the calendar can be monitored and set from the loader. They can also be monitored and set from an application program.

Note: Even if the values of the calendar are not used in an application program, perform the setting; the SX series use the values of the calendar for recording times of power failure of the system and occurrence of an error.

(1) Calendar's value range

The calendar can measure calendar values from January 1st, 00:00:00, 1970 through December 31st, 23:59:59, 2069.

Note: One second after December 31, 2069 23:59:59, the date and time will turn back to January 1, 1970.

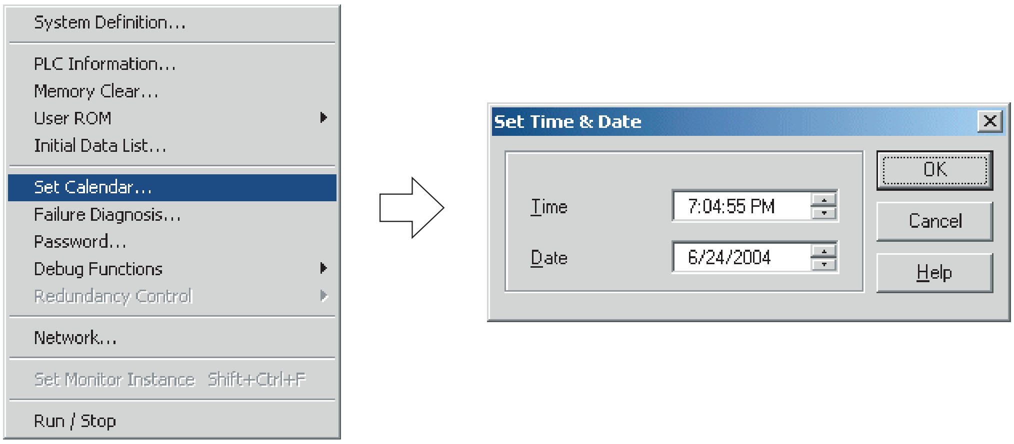
(2) Calendar accuracy

The accuracy of the calendar (clock) in the CPU is 27 seconds per month (at an ambient temperature of 25 deg C).

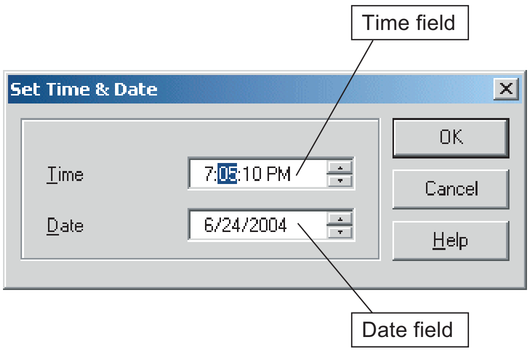
Note: The accuracy of the calendar clock varies depending on environmental conditions such as ambient temperature. When the CPU module is to be used in a system where a high calendar accuracy is required, measure the actual calendar accuracy and review the inspection (clock calibration) period.

(3) Monitoring and setting up the calendar clock from the loader

1) Execute [Set Calendar...] from [PLC Function] menu to open [Set Time & Date] dialog.



2) Move the cursor to the position in the calendar to be changed and set the value.



Time: Allows setting hours, minutes and seconds.
Date: Allows setting year, month and day.

3) After setting the time and date, click [OK] button to register the set data to the calendar in the CPU module.

(4) Monitoring and setting up the calendar clock from an application program

1) HW_RTC (Hardware RTC) - Original FB
Use the hardware RTC function block (HW_RTC) to monitor and set up the calendar clock from an application program. For detailed instructions, refer to the manual for HW_RTC FB.

Note: DT type (date and time type) data should be specified for HW_RTC. The range of DT data is January 1, 1970 00:00:00 to February 7, 2160 6:18:15. The data allowed for HW_RTC ranges from January 1, 1970 00:00:00 to December 31 23:59:59. (If any date and time is specified, an expected value will be set.)

When setting up the calendar clock using data supplied from an external device, it is necessary to convert the input data to the DT type. The DT type data is equivalent to a 32-bit unsigned integer in seconds that starts at January 1st, 00:00:00, 1970.

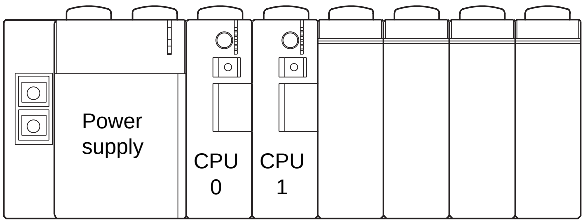
Examples:

(1) January 1st, 12:34:54, 1970	→	DINT#45296	WORD#16#0000B0F0
(2) January 1st, 00:00:00, 1998	→	DINT#883612800	WORD#16#34AADC80

2) RTC (real-time clock)
The RTC cannot be used to set up the calendar clock. If RTC is used to set up the calendar clock, a relative value is stored in an area on which the calendar clock will run.

(5) Time adjustment function

In a multi-CPU system the MICREX-SX series CPU modules provide a function that automatically adjusts the time of their internal clock (real-time clock).

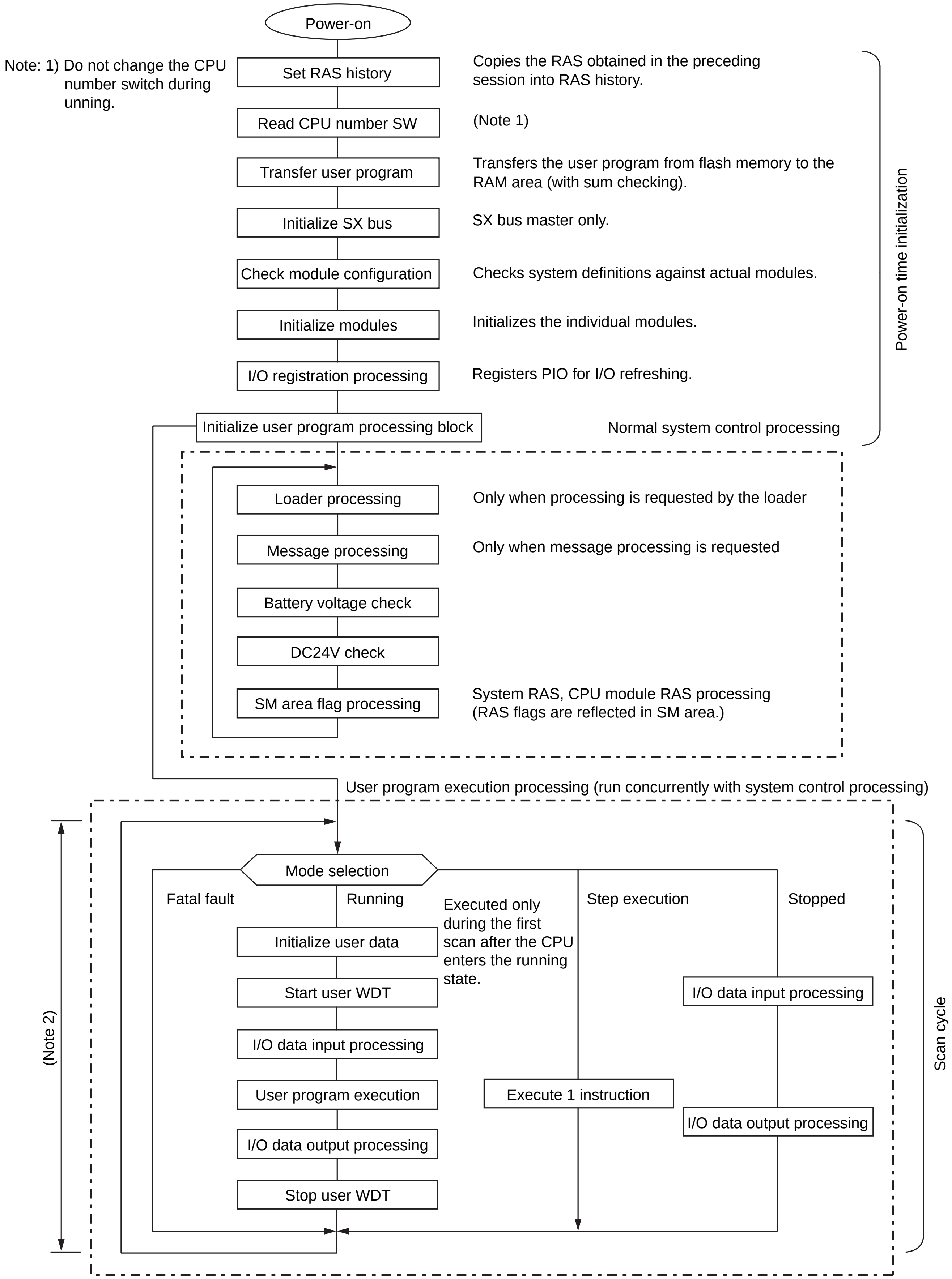


1) Time adjustment management
The number “0” CPU module adjusts the real-time clock of the other CPU modules. If the CPU0 goes down, another CPU module is assigned to adjust the real-time clock of the other CPU modules.

2) Timing of time adjustment
- When the system is powered on. Subsequently, it is adjusted at predetermined intervals (every minute).
- When the real-time clock is updated from the loader or an application program.

2-8 Operating Flowchart

The operating flowchart given below shows the power-on sequence of the MICREX-SX series SPH system and the subsequent operation sequence.



Notes: 2) The execution time of the default task specified by the programming loader refers to the time in this section.
3) Sometimes it takes time to finish message related instruction when mode transition has occured.

Section 3 Instruction

Section 3 Instruction

3-1 Instruction Types

3-1-1 Instruction types

The MICREX-SX Series instructions are classified into the types listed below.

Instruction types	Description	Related section
Basic instructions	Instructions for basic elements configuring the ladder sequence such as contacts and coils.	3-2-1
Program control instructions	Instructions for controlling program execution (jumps, loops, etc.).	3-2-2
Conversion instructions	Instructions for converting device data types.	3-2-3
Numeric operation instructions	Instructions such as arithmetic operations, trigonometric functions, as well as transfer-related instructions.	3-2-4
Bit string operation instructions	Instructions for logical, shift and rotation operations.	3-2-5
Selection/comparison instructions	Instructions for selection and comparison operations.	3-2-6
Character string instructions	Instructions for fetching characters from character-string data and controlling character string data such as connection, substitution, comparison, etc.	3-2-7
Time instructions	Instructions for controlling time-related data.	3-2-8
Original functions	Original instructions of MICREX-SX SPH Series.	3-2-9
IEC standard function blocks	Function blocks specified by IEC 61131-3 of the international standards related to the programming languages. Timers and counters are classified under this group.	3-2-10
Original function blocks	Original function blocks of MICREX-SX SPH Series.	3-2-11
User functions/user function blocks	Instructions created by user.	Appendix 4

<About contents in description field for instructions>
“Available devices” are described for each instruction. Availability of a device is determined depending on whether it is available when creating an actual application program. For an unrealistic device, it is indicated as “-” (unavailable)e even when it can be specified by PLC Programmer.

(Description example)

Symbol

S: Loop count (0 to 2147483647)

Available devices

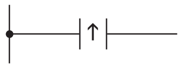
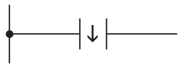
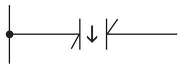
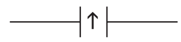
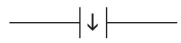
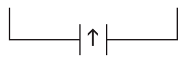
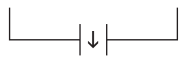
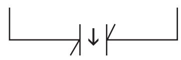
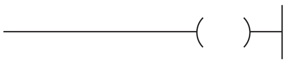
	Word devices (W*)								Double-word devices (D*)								Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	
S	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	○	○

*1 *2 *3

*1. Input/output memories X and Y are indicated as unavailable because they are unavailable in principle unless a module actually exists.
*2. System memory SM is memory with defined usage. It is indicated as unavailable because using it as an auxiliary memory for applications is unrealistic.
*3. Current timer values T and TR are indicated as unavailable because controlling the loop count with them is difficult.

3-1 Instruction Types

3-1-2 Instruction list

Type	Instruction	Symbol	Functions	No. of steps	Related page
Basic instructions	LD		Specifies normal open contact from bus.	1	3-19
	LD+		Specifies normal open contact from bus. (rising edge differential)	8	3-20
	LD–		Specifies normal open contact from bus. (falling edge differential)	8	3-22
	LDI		Specifies normal close contact from bus.	1	3-19
	LDI+		Specifies normal close contact from bus. (rising edge differential)	8	3-20
	LDI–		Specifies normal close contact from bus. (falling edge differential)	8	3-22
	AND		Connects normal open contact in series.	1	3-19
	AND+		Connects normal open contact in series. (rising edge differential)	8	3-20
	AND–		Connects normal open contact in series. (falling edge differential)	8	3-22
	ANI		Connects normal close contact in series.	1	3-19
	ANI+		Connects normal close contact in series. (rising edge differential)	8	3-20
	ANI–		Connects normal close contact in series. (falling edge differential)	8	3-22
	OR		Connects normal open contact in parallel.	1	3-19
	OR+		Connects normal open contact in parallel. (rising edge differential)	8	3-20
	OR–		Connects normal open contact in parallel. (falling edge differential)	8	3-22
	ORI		Connects normal close contact in parallel.	1	3-19
	ORI+		Connects normal close contact in parallel. (rising edge differential)	8	3-20
	ORI–		Connects normal close contact in parallel. (falling edge differential)	8	3-22
	ANB		Connects in series between logical blocks.	1	3-24
	ORB		Connects in parallel between logical blocks.	1	3-24
	MPS	Not marked in a ladder diagram	Writes operation result into register.	4	3-25
	MRD	Not marked in a ladder diagram	Reads operation result from register.	3	3-25
	MPP	Not marked in a ladder diagram	Reads and resets operation result.	3	3-25
	OUT		Coil	1	3-27

3-1 Instruction Types

Type	Instruction	Symbol	Functions	No. of steps	Related page
Basic instructions	OUTI	(/)	Inverted coil	1	3-27
	OUT+	(↑)	Rising edge differential coil	10	3-28
	OUT-	(↓)	Falling edge differential coil	10	3-28
	SET	(S)	Set coil	1	3-29
	RST	(R)	Reset coil	1	3-29
	NOT		Invert	1	3-30
	OUTSC	(SC)	Step control	8	3-31
Program control instructions	JEND	JEND P□	Jump end	1	3-33
	JMP	JMP P□	Unconditional jump	2	3-33
	JMPC	JMPC P□	Conditional jump	2	3-33
	JMPCN	JMPCN P□	Negative conditional jump	2	3-33
	LOOP	LOOP S	Loop	14	3-34
	CONT	CONT	Loop end	3	3-34
	RET	RET	Unconditional return	3	3-35
	RETC	RETC	Conditional return	3	3-35
	RETCN	RETCN	Negative conditional return	3	3-35
	MC	MC N D	Master control set	2	3-36
	MCR	MCR N	Master control reset	1	3-36
	SECTION	SECTION □	Section of program	2	3-37
Conversion instructions	BCD	BCD S → D	BIN to BCD	6	3-38
	BIN	BIN S → D	BCD to BIN	6	3-39
	I_TO_R	I_TO_R S → D	Integer to real (Signed integer 16 bits)	6	3-40
	DI_TO_R	DI_TO_R S → D	Integer to real (Signed integer 32 bits)	6	3-40
	UI_TO_R	UI_TO_R S → D	Integer to real (Unsigned integer 16 bits)	6	3-40

3-1 Instruction Types

Type	Instruction	Symbol	Functions	No. of steps	Related page
Conversion instructions	UDI_TO_R	UDI_TO_R S → D	Integer to real (Unsigned integer 32 bits)	6	3-40
	R_TO_I	R_TO_I S → D	Real to integer (Signed integer 16 bits)	6	3-41
	R_TO_DI	R_TO_DI S → D	Real to integer (Signed integer 32 bits)	6	3-41
	R_TO_UI	R_TO_UI S → D	Real to integer (Unsigned integer 16 bits)	6	3-41
	R_TO_UDI	R_TO_UDI S → D	Real to integer (Unsigned integer 32 bits)	6	3-41
	TRUNC_I	TRUNC_I S → D	Real to integer (Signed integer 16 bits, fractional part truncation)	6	3-43
	TRUNC_DI	TRUNC_DI S → D	Real to integer (Signed integer 32 bits, fractional part truncation)	6	3-43
	TRUNC_UI	TRUNC_UI S → D	Real to integer (Unsigned integer 16 bits, fractional part truncation)	6	3-43
	TRUNC_UDI	TRUNC_UDI S → D	Real to integer (Unsigned integer 32 bits, fractional part truncation)	6	3-43
	I_TO_DI	I_TO_DI S → D	INT to DINT	6	3-45
	DI_TO_I	DI_TO_I S → D	DINT to INT	6	3-45
Numeric operation instructions	ADD	ADD S1, S2 → D	Addition (Signed)	6	3-46
	ADD_UAI	ADD_UAI S1, S2 → D	Addition (Unsigned)	6	3-46
	ADD_R	ADD_R S1, S2 → D	Addition (Real)	6	3-46
	SUB	SUB S1, S2 → D	Subtraction (Signed)	6	3-47
	SUB_UAI	SUB_UAI S1, S2 → D	Subtraction (Unsigned)	6	3-47
	SUB_R	SUB_R S1, S2 → D	Subtraction (Real)	6	3-47
	MUL	MUL S1, S2 → D	Multiplication (Signed)	6	3-48
	MUL_UAI	MUL_UAI S1, S2 → D	Multiplication (Unsigned)	6	3-48
	MUL_R	MUL_R S1, S2 → D	Multiplication (Real)	6	3-48
	DIV	DIV S1, S2 → D	Division (Signed)	6	3-49
	DIV_UAI	DIV_UAI S1, S2 → D	Division (Unsigned)	6	3-49
	DIV_R	DIV_R S1, S2 → D	Division (Real)	6	3-49

3-1 Instruction Types

Type	Instruction	Symbol	Functions	No. of steps	Related page
Numeric operation instructions	MOD	MOD S1, S2 → D	Division remainder (Signed)	6	3-50
	MOD_UAI	MOD_UAI S1, S2 → D	Division remainder (Unsigned)	6	3-50
	EXPT	EXPT S1, S2 → D	Exponent	6	3-51
	ABS	ABS S → D	Absolute	6	3-52
	ABS_R	ABS_R S → D	Absolute (Real)	6	3-52
	SQRT	SQRT S → D	Square root	6	3-53
	LN	LN S → D	Natural logarithm	6	3-54
	LOG	LOG S → D	Common logarithm	6	3-55
	EXP	EXP S → D	Exponent	6	3-56
	SIN	SIN S → D	Sine	6	3-57
	COS	COS S → D	Cosine	6	3-58
	TAN	TAN S → D	Tangent	6	3-59
	ASIN	ASIN S → D	Arcsine	6	3-60
	ACOS	ACOS S → D	Arccosine	6	3-61
	ATAN	ATAN S → D	Arctangent	6	3-62
Transfer instructions	MOVE	MOVE S → D	Move	7	3-63
	NEG	NEG S → D	Negation	6	3-64
	NEG_R	NEG_R S → D	Negation (Real)	6	3-64
	BMOV	BMOV S, D, N	Block move	7	3-65
	FMOV	FMOV S, D, N	Fill move	19	3-66
	XCH	XCH, D1, D2	Exchange	11	3-67
	BDMPX	BDMPX S1, D, S2, N	Indirect put (Block move)	12	3-68
	BMPX	BMPX S1, S2, D, N	Indirect get (Block move)	12	3-69

3-1 Instruction Types

Type	Instruction	Symbol	Functions	No. of steps	Related page
Bit string operation instructions	AND_AW	AND_AW S1, S2 → D	Logical AND	6	3-70
	OR_AW	OR_AW S1, S2 → D	Logical OR	6	3-71
	XOR_B	XOR_B S1, S2 → D	Logical exclusive OR (Bit)	6	3-72
	XOR_AW	XOR_AW S1, S2 → D	Logical exclusive OR (Word)	6	3-72
	XORN_B	XORN_B S1, S2 → D	Logical exclusive NOR (Bit)	6	3-73
	XORN_AW	XORN_AW S1, S2 → D	Logical exclusive NOR (Word)	6	3-73
	NOT_AW	NOT_AW S → D	Logical NOT	6	3-74
	ROR	ROR S, N → D	Rotation right	6	3-75
	ROL	ROL S, N → D	Rotation left	6	3-75
	SHR	SHR S, N → D	Shift right	6	3-76
	SHL	SHL S, N → D	Shift left	6	3-76
Selection/comparison instructions	LD > AND > OR >	> S1, S2	Comparison (S1 > S2)	3	3-77
	LD_UAI > AND_UAI > OR_UAI >	UAI > S1, S2		3	3-77
	LD_R > AND_R > OR_R >	R > S1, S2		3	3-77
	LD_B > AND_B > OR_B >	B > S1, S2		3	3-77
	LD_AW > AND_AW > OR_AW >	AW > S1, S2		3	3-77
	LD_TOD > AND_TOD > OR_TOD >	TOD > S1, S2		3	3-77
	LD_DT > AND_DT > OR_DT >	DT > S1, S2		3	3-77
	LD_D > AND_D > OR_D >	D > S1, S2		3	3-77

3-1 Instruction Types

Type	Instruction	Symbol	Functions	No. of steps	Related page
Selection/comparison instructions	LD_T > AND_T > OR_T >	T > S1, S2	Comparison (S1 > S2)	3	3-77
	LD >= AND >= OR >=	>= S1, S2	Comparison (S1 >= S2)	3	3-78
	LD_UAI >= AND_UAI >= OR_UAI >=	UAI >= S1, S2		3	3-78
	LD_R >= AND_R >= OR_R >=	R >= S1, S2		3	3-78
	LD_B >= AND_B >= OR_B >=	B >= S1, S2		3	3-78
	LD_AW >= AND_AW >= OR_AW >=	AW >= S1, S2		3	3-78
	LD_TOD >= AND_TOD >= OR_TOD >=	TOD >= S1, S2		3	3-78
	LD_DT >= AND_DT >= OR_DT >=	DT >= S1, S2		3	3-78
	LD_D >= AND_D >= OR_D >=	D >= S1, S2		3	3-78
	LD_T >= AND_T >= OR_T >=	T >= S1, S2		3	3-78
	LD = AND = OR =	= S1, S2	Comparison (S1 = S2)	3	3-79
	LD_UAI = AND_UAI = OR_UAI =	UAI = S1, S2		3	3-79
	LD_R = AND_R = OR_R =	R = S1, S2		3	3-79
	LD_B = AND_B = OR_B =	B = S1, S2		3	3-79

3-1 Instruction Types

Type	Instruction	Symbol	Functions	No. of steps	Related page
Selection/comparison instructions	LD_AW = AND_AW = OR_AW =	AW = S1, S2	Comparison (S1 = S2)	3	3-79
	LD_TOD = AND_TOD = OR_TOD =	TOD = S1, S2		3	3-79
	LD_DT = AND_DT = OR_DT =	DT = S1, S2		3	3-79
	LD_D = AND_D = OR_D =	D = S1, S2		3	3-79
	LD_T = AND_T = OR_T =	T = S1, S2		3	3-79
	LD <= AND <= OR <=	<= S1, S2	Comparison (S1 <= S2)	3	3-80
	LD_UAI <= AND_UAI <= OR_UAI <=	UAI <= S1, S2		3	3-80
	LD_R <= AND_R <= OR_R <=	R <= S1, S2		3	3-80
	LD_B <= AND_B <= OR_B <=	B <= S1, S2		3	3-80
	LD_AW <= AND_AW <= OR_AW <=	AW <= S1, S2		3	3-80
	LD_TOD <= AND_TOD <= OR_TOD <=	TOD <= S1, S2		3	3-80
	LD_DT <= AND_DT <= OR_DT <=	DT <= S1, S2		3	3-80
	LD_D <= AND_D <= OR_D <=	D <= S1, S2		3	3-80
	LD_T <= AND_T <= OR_T <=	T <= S1, S2		3	3-80
	LD < AND < OR <	< S1, S2	Comparison (S1 < S2)	3	3-81
	LD_UAI < AND_UAI < OR_UAI <	UAI < S1, S2		3	3-81

Type	Instruction	Symbol	Functions	No. of steps	Related page
Selection/comparison instructions	LD_R < AND_R < OR_R <	R < S1, S2	Comparison (S1 < S2)	3	3-81
	LD_B < AND_B < OR_B <	B < S1, S2		3	3-81
	LD_AW < AND_AW < OR_AW <	AW < S1, S2		3	3-81
	LD_TOD < AND_TOD < OR_TOD <	TOD < S1, S2		3	3-81
	LD_DT < AND_DT < OR_DT <	DT < S1, S2		3	3-81
	LD_D < AND_D < OR_D <	D < S1, S2		3	3-81
	LD_T < AND_T < OR_T <	T < S1, S2		3	3-81
	LD <> AND <> OR <>	<> S1, S2	Comparison (S1 <> S2)	3	3-82
	LD_UAI <> AND_UAI <> OR_UAI <>	UAI <> S1, S2		3	3-82
	LD_R <> AND_R <> OR_R <>	R <> S1, S2		3	3-82
	LD_B <> AND_B <> OR_B <>	B <> S1, S2		3	3-82
	LD_AW <> AND_AW <> OR_AW <>	AW <> S1, S2		3	3-82
	LD_TOD <> AND_TOD <> OR_TOD <>	TOD <> S1, S2		3	3-82
	LD_DT <> AND_DT <> OR_DT <>	DT <> S1, S2		3	3-82
	LD_D <> AND_D <> OR_D <>	D <> S1, S2		3	3-82
	LD_T <> AND_T<> OR_T <>	T <> S1, S2		3	3-82

3-1 Instruction Types

Type	Instruction	Symbol	Functions	No. of steps	Related page
Selection/comparison instructions	SEL_B	SEL_B S1, S2, S3 → D	Select S1: When ON, S3 → D When OFF, S2 → D	12	3-83
	SEL	SEL S1, S2, S3 → D		12	3-83
	SEL_UAI	SEL_UAI S1, S2, S3 → D		12	3-83
	SEL_R	SEL_R S1, S2, S3 → D		12	3-83
	SEL_AW	SEL_AW S1, S2, S3 → D		12	3-83
	SEL_T	SEL_T S1, S2, S3 → D		12	3-83
	SEL_STR	SEL_STR S1, S2, S3 → D		12	3-83
	MAX	MAX S1, S2 → D	Maximum value	6	3-84
	MAX_UAI	MAX_UAI S1, S2 → D		6	3-84
	MAX_R	MAX_R S1, S2 → D		6	3-84
	MIN	MIN S1, S2 → D	Minimum value	6	3-85
	MIN_UAI	MIN_UAI S1, S2 → D		6	3-85
	MIN_R	MIN_R S1, S2 → D		6	3-85
	LIMIT	LIMIT S1, S2, S3 → D	Limit When S2<S1, S1 → D When S1<=S2<=S3, S2 → D When S2>S3, S3 → D	9	3-86
	LIMIT_UAI	LIMIT_UAI S1, S2, S3 → D		9	3-86
	LIMIT_R	LIMIT_R S1, S2, S3 → D		9	3-86
Character string instructions	LEN	LEN S → D	Get length	6	3-87
	LEFT	LEFT S, N → D	Get left sub-string	8	3-88
	RIGHT	RIGHT S, N → D	Get right sub-string	8	3-89
	MID	MID S, N1, N2 → D	Get middle sub-string	9	3-90
	CONCAT	CONCAT S1, S2 → D	Concatenate	9	3-91
	INSERT	INSERT S1, S2, N → D	Insert string	9	3-92
	DELETE	DELETE S, N1, N2 → D	Delete string	9	3-93

3-1 Instruction Types

Type	Instruction	Symbol	Functions	No. of steps	Related page
Character string instructions	REPLACE	REPLACE S1, S2, N1, N2 → D	Replace string	10	3-94
	FIND	FIND S1, S2 → D	Find string	7	3-95
	LD_STR> AND_STR> OR_STR>	STR> S1, S2	Compare string (S1>S2)	4	3-96
	LD_STR>= AND_STR>= OR_STR>=	STR>= S1, S2	Compare string (S1>=S2)	4	3-97
	LD_STR= AND_STR= OR_STR=	STR= S1, S2	Compare string (S1=S2)	4	3-98
	LD_STR<= AND_STR<= OR_STR<=	STR<= S1, S2	Compare string (S1<=S2)	4	3-99
	LD_STR< AND_STR< OR_STR<	STR< S1, S2	Compare string (S1<S2)	4	3-100
	LD_STR<> AND_STR<> OR_STR<>	STR<> S1, S2	Compare string (S1<>S2)	4	3-101
	MOVE_STR	MOVE_STR S → D	Move string	8	3-102
Time instructions	ADD_T_T	ADD_T_T S1, S2 → D	Add time	8	3-103
	ADD_TD_T	ADD_TD_T S1, S2 → D	Add time (Duration + Time)	15	3-103
	ADD_DT_T	ADD_DT_T S1, S2 → D	Add time (Date and Time + Time)	10	3-103
	SUB_T_T	SUB_T_T S1, S2 → D	Subtract time	8	3-105
	SUB_D_D	SUB_D_D S1, S2 → D	Subtract time (Date - Date)	10	3-105
	SUB_TD_T	SUB_TD_T S1, S2 → D	Subtract time (Duration - Time)	13	3-105
	SUB_TD_TD	SUB_TD_TD S1, S2 → D	Subtract time (Duration - Duration)	10	3-105
	SUB_DT_T	SUB_DT_T S1, S2 → D	Subtract time (Date and Time - Time)	10	3-105
	SUB_DT_DT	SUB_DT_DT S1, S2 → D	Subtract time (Date and Time - Data and Time)	10	3-105
	MUL_T_UDI	MUL_T_UDI S1, S2 → D	Multiply time	8	3-107
	MUL_T_R	MUL_T_R S1, S2 → D	Multiply time (Time × real)	9	3-107

3-1 Instruction Types

Type	Instruction	Symbol	Functions	No. of steps	Related page
Time instructions	DIV_T_UDI	DIV_T_UDI S1, S2 → D	Divide time	8	3-108
	DIV_T_R	DIV_T_R S1, S2 → D	Divide time (Time / real)	9	3-108
	CONCAT_D_D	CONCAT_D_D S1, S2 → D	Concatenate time	8	3-110
	DT_TO_TOD	DT_TO_TOD S → D	Date and time - duration conversion	9	3-111
	DT_TO_D	DT_TO_D S → D	Date and time - date conversion	10	3-111
Original functions	SBIT	SBIT S, N → D	Set bit	6	3-112
	RBIT	RBIT S, N → D	Reset bit	6	3-113
	TBIT	TBIT S, N → D	Test bit	6	3-114
	DECODE	DECODE S → D	Decode	6	3-115
	ENCODE	ENCODE S → D	Encode	6	3-116
	BITCOUNT	BITCOUNT S → D	Bit count	6	3-117
	STR_TO_UI	STR_TO_UI S → D	Convert string to number	6	3-118
	UI_TO_STR	UI_TO_STRI S → D	Convert number to string	7	3-118
	SJ_TO_STR	SJ_TO_STR S → D	Convert shift-JIS to string	8	3-119
	STR_TO_SJ	STR_TO_SJ S → D	Convert string to shift-JIS	7	3-120
	BYTE_LEN	BYTE_LEN S → D	Byte length	6	3-121
	DBAND	DBAND S1, S2 → D	Dead band	8	3-122
	DBAND_R	DBAND_R S1, S2 → D	Dead band (real)	8	3-122
	BIAS	BIAS S1, S2 → D	Bias	8	3-123
	BIAS_R	BIAS_R S1, S2 → D	Bias (real)	8	3-123
	SC_COIL	SC_COIL C, N1, N2 → D	Step sequence coil	9	3-124
	SC	SC C, N1, N2 → D	Step sequence bit	9	3-124
	SLC	SLC S1, S2 → D	Shift left 32 bits with carry	8	3-125

3-1 Instruction Types

Type	Instruction	Symbol	Functions	No. of steps	Related page
Original functions	SRC	SRC S1, S2 → D	Shift right 32 bits with carry	8	3-125
	SLCO	SLCO S → D	Shift left 32 bits carry	7	3-126
	SRCO	SRCO S → D	Shift right 32 bits carry	7	3-126
	ADC	ADC S1, S2, S3 → D	32 bits addition with carry	9	3-127
	ADCO	ADCO S1, S2, S3 → D	32 bits addition carry	9	3-127
	SBB	SBB S1, S2, S3 → D	32 bits subtraction with borrow	9	3-128
	SBBO	SBBO S1, S2, S3 → D	32 bits subtraction borrow	9	3-128
	MULL	MULL S1, S2 → D	64 bits multiplication (Lower-order digit)	8	3-129
	MULU	MULU S1, S2 → D	64 bits multiplication (Higher-order digit)	8	3-129
	DIVL	DIVL S1, S2, S3 → D	64 bits division (Lower-order digit)	9	3-130
	DIVU	DIVU S1, S2, S3 → D	64 bits division (Higher-order digit)	9	3-130
IEC standard function blocks	SR	SR S1 PARA S2 PARA S3 PARA D	Set reset flip-flop	9	3-131
	RS	RS S1 PARA S2 PARA S3 PARA D	Reset set flip-flop	9	3-131
	R_TRIG	R_TRIG S1 PARA S2 PARA D	Rising edge trigger	7	3-132
	F_TRIG	F_TRIG S1 PARA S2 PARA D	Falling edge trigger	7	3-132

3-1 Instruction Types

Type	Instruction	Symbol	Functions	No. of steps	Related page
IEC standard function blocks	CTU	S1 CTU S2, S3 S4 CRST	Up counter	13	3-133
	CTD	S1 CTD S2, S3 S4 CRST	Down counter	13	3-133
	TP	S1 TP S2, S3	Pulse	10	3-134
	TON	S1 TON S2, S3	On-delay timer	10	3-135
	TOF	S1 TOF S2, S3	Off-delay timer	10	3-135
	RTC	S1 RCT S2, S3	Real-time clock	10	3-136
Original function blocks	RCT	S1 RCT S2, S3 S4 CRST	Ring counter	13	3-137
	TMR	S1 TMR S2, S3 S4 TRST	Integrating timer	14	3-139
	MR	S1 MR S2, S3	Retriggerable timer	10	3-141
	M_OPEN	M_OPEN S1 PARA S2 ⋮ PARA S10 PARA D1 ⋮ PARA D4	Open channel	32	3-142
	M_SEND	M_SEND S1 PARA S2 ⋮ PARA S5 PARA D1 ⋮ PARA D3	Send message	23	3-145

Type	Instruction	Symbol	Functions	No. of steps	Related page
Original function blocks	M_RECEIVE	M_RECEIVE S1 PARA S2 ⋮ PARA S5 PARA D1 ⋮ PARA D3	Receive message	23	3-147
	READ_W READ_B	READ_W (B) S1 PARA S2 ⋮ PARA S5 PARA D1 ⋮ PARA D3	Direct read	24 20	3-149
	WRITE_W WRITE_B	WRITE_W (B) S1 PARA S2 ⋮ PARA S5 PARA D1 ⋮ PARA D3	Direct write	24 20	3-151
	R_READ	R_READ S1 PARA S2 ⋮ PARA S9 PARA D1 ⋮ PARA D3	Remote data read	32	3-153
	R_WRITE	R_WRITE S1 PARA S2 ⋮ PARA S9 PARA D1 ⋮ PARA D3	Remote data write	32	3-156

3-1 Instruction Types

Type	Instruction	Symbol	Functions	No. of steps	Related page
Original function blocks	F_READ	F_READ S1 PARA S2 ⋮ PARA S6 PARA D1 ⋮ PARA D4	File data read	28	3-160
	F_WRITE	F_WRITE S1 PARA S2 ⋮ PARA S6 PARA D1 ⋮ PARA D4	File data write	26	3-163
	EXT_T_S	EXT_T_S S1 PARA S2 PARA S3 PARA D1 ⋮ PARA D4	Extension test & set	19	3-166
	FFST	FFST S1 PARA S2 ⋮ PARA S7 PARA D	Sequential file store	24	3-168
	FIFO	FIFO S1 PARA S2 ⋮ PARA S7 PARA D	Sequential file load first	24	3-170

3-1 Instruction Types

Type	Instruction	Symbol	Functions	No. of steps	Related page
Original function blocks	FILO	FILO S1 PARA S2 ⋮ PARA S7 PARA D	Sequential file load last	24	3-172
	FILTER_DI FILTER_R	FILTER_DI (R) S1 PARA S2 PARA S3 PARA S4 PARA D	Filter	12 12	3-174
	INT_DI INT_R	INT_DI (R) S1 PARA S2 ⋮ PARA S6 PARA D1 PARA D2	Integrate	20 20	3-176
	DIF_DI DIF_R	DIF_DI (R) S1 PARA S2 ⋮ PARA S4 PARA D	Differentiate	12 12	3-178
	PULSE_CNT	PULSE_CNT S1 PARA S2 ⋮ PARA S4 PARA D1 PARA D2	Pulse count	16	3-180

3-1 Instruction Types

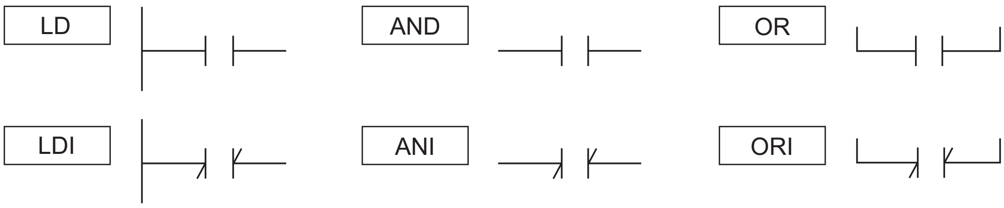
Type	Instruction	Symbol	Functions	No. of steps	Related page
Original function blocks	PULSE_OUT	PULSE_OUT S1 PARA S2 ⋮ PARA S4 PARA D1 PARA D2	Pulse output	16	3-181
	PWM	PWM S1 PARA S2 PARA S3 PARA S4 PARA D	Modulate pulse width	13	3-182
	HW_RTC	HW_RTC S1 PARA S2 PARA S3 PARA D1 PARA D2	Hardware RTC	13	3-183
	T_S	T_S S1 PARA S2 PARA D1 PARA D2	Test & set	12	3-184
	BANK_CHG	BANK_CHG S1 PARA S2 PARA S3 PARA D1 ⋮ PARA D4	Change bank	19	3-185

3-2-1 Basic instructions

(1) Operation start, serial connect, and parallel connect

Symbol

Operation start Serial connect Parallel connect



Available devices

Bit devices									
X	Y	M	L	SM	T	TR	C	F	V
○	○	○	○	○	○	○	○	○	○

Function

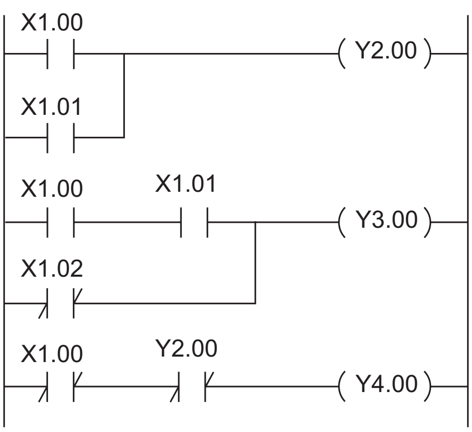
LD, LDI 1) The LD instruction is the NO contact instruction to start operation. The LDI instruction is the NC contact to start operation. Each instruction fetches the ON or OFF state of the specified device as the operation result.

AND, ANI 1) The AND instruction connects a NO contact in series. The ANI instruction connects a NC contact in series. These instructions AND the ON or OFF state of the specified device and the result of the preceding operation. The result of this AND operation is the result of the instruction.

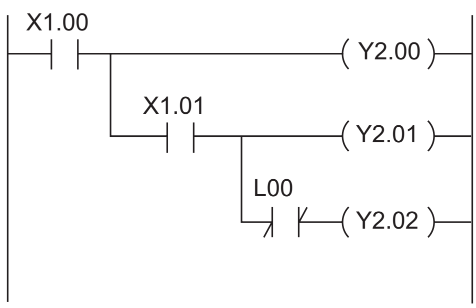
OR, ORI 1) The OR instruction connects a NO contact in parallel. The ORI instruction connects a NC contact in parallel. These instructions OR the ON or OFF state of the specified device and the result of the preceding operation. The result of this OR operation is the result of the instruction.

Program examples

1) Ladder diagram representation



2) Ladder diagram representation

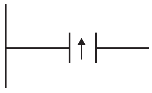


(2) Rising edge differentiate

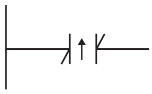
Symbol

Operation start

LD+



LDI+



Serial connect

AND+

ANI+

Parallel connect

OR+

ORI+

Available devices

Bit devices									
X	Y	M	L	SM	T	TR	C	F	V
○	○	○	○	○	○	○	○	○	○

Specifies the instance number (0 to 65535) of rising edge differential contact .

Function

LD+ , LDI+

1) The LD+ instruction is an OFF-to-ON differentiation NO contact instruction for starting operation. This instruction sets the specified contact ON for one program scan-cycle when an OFF-to-ON transition from the specified device is detected.
The LDI+ instruction is an OFF-to-ON differentiation NC contact instruction for starting operation. This instruction sets specified contact OFF for one program scan-cycle when an OFF-to-ON transition from the specified device is detected.

AND+ , ANI+

1) The AND+ instruction is an OFF-to-ON differentiation contact instruction for connecting a NO contact in series. This instruction sets the specified contact ON for one program scan-cycle when an OFF-to-ON transition from the specified device is detected. This instruction then ANDs the result of executing the program and the result of the preceding operation. The result of this AND operation becomes the result of the instruction.
The ANI+ instruction is an OFF-to-ON differentiation contact instruction for connecting a NC contact in series. This instruction sets the specified contact OFF for one program scan-cycle when an OFF-to-ON transition from the specified device is detected. This instruction then ANDs the result of executing the program and the result of the preceding operation. The result of this AND operation becomes the result of the instruction.

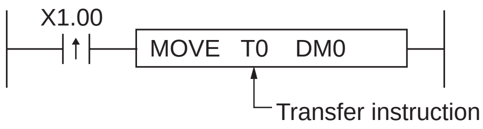
OR+ , ORI+

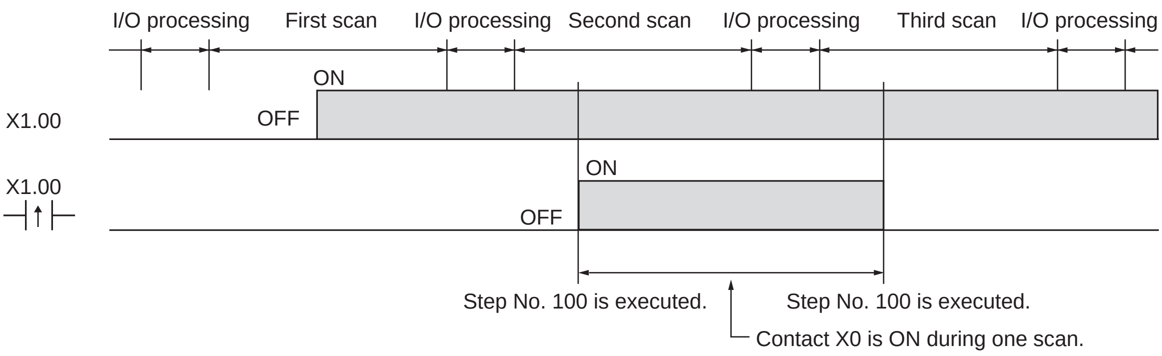
1) The OR+ instruction is an OFF-to-ON differentiation contact instruction for connecting a NO contact in parallel. This instruction sets the specified contact ON for one program scan-cycle when an OFF-to-ON transition from the specified device is detected. This instruction then ORs the result of executing the program and the result of the preceding operation. The result of this OR operation becomes the result of the instruction.
The ORI+ instruction is an OFF-to-ON differentiation contact instruction for connecting a NC contact in parallel. This instruction sets the specified contact OFF for one program scan-cycle when an OFF-to-ON transition from the specified device is detected. This instruction then ORs the result of executing the program and the result of the preceding operation. The result of this OR operation becomes the result of the instruction.

Program examples

1) When contact X1.00 is set ON, T0 (timer current value) is transferred just once to DM0.

(Ladder diagram representation)



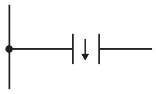


(3) Falling edge differentiate

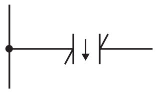
Symbol

Operation start

LD–



LDI–



Serial connect

AND–

ANI–

Parallel connect

OR–

ORI–

Available devices

Bit devices									
X	Y	M	L	SM	T	TR	C	F	V
○	○	○	○	○	○	○	○	○	○

Specifies the instance number (0 to 65535) of falling edge differential contact.

Function

LD– , LDI–

1) The LD– instruction is an ON-to-OFF differentiation NO contact instruction for starting operation. This instruction sets the specified contact ON for one program scan-cycle when an ON-to-OFF transition from the specified device is detected.
The LDI– instruction is an OFF-to-ON differentiation NC contact instruction for starting operation. This instruction sets specified contact OFF for one program scan-cycle when an ON-to-OFF transition from the specified device is detected.

AND– ANI–

1) The AND– instruction is an OFF-to-ON differentiation contact instruction for connecting a NO contact in series. This instruction sets the specified contact ON for one program scan-cycle when an ON-to-OFF transition from the specified device is detected. This instruction then ANDs the result of executing the program and the result of the preceding operation. The result of this AND operation becomes the result of the instruction.
The ANI– instruction is an ON-to-OFF differentiation contact instruction for connecting a NC contact in series. This instruction sets the specified contact OFF for one program scan-cycle when an ON-to-OFF transition from the specified device is detected. This instruction then ANDs the result of executing the program and the result of the preceding operation. The result of this AND operation becomes the result of the instruction.

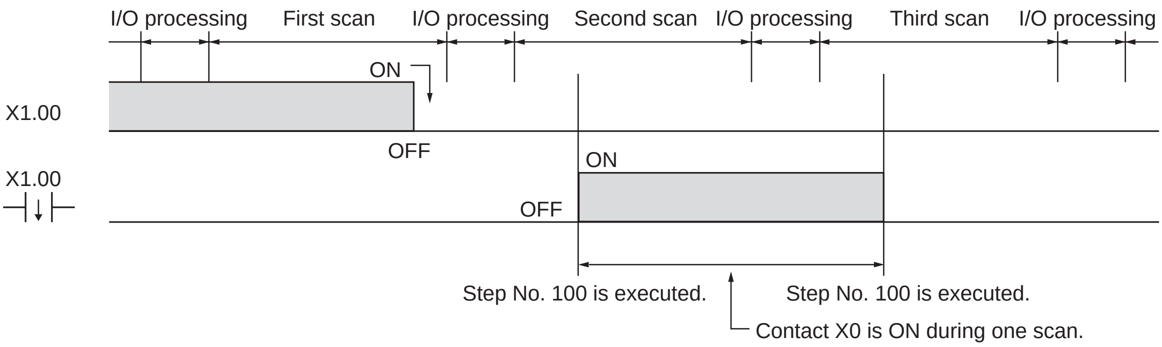
OR– , ORI–

1) The OR– instruction is an ON-to-OFF differentiation contact instruction for connecting a NO contact in parallel. This instruction sets the specified contact ON for one program scan-cycle when an ON-to-OFF transition from the specified device is detected. This instruction then ORs the result of executing the program and the result of the preceding operation. The result of this OR operation becomes the result of the instruction.
The ORI– instruction is an ON-to-OFF differentiation contact instruction for connecting a NC contact in parallel. This instruction sets the specified contact OFF for one program scan-cycle when an ON-to-OFF transition from the specified device is detected. This instruction then ORs the result of executing the program and the result of the preceding operation. The result of this OR operation becomes the result of the instruction.

Program examples

1) When contact X1.00 is set OFF, C0 (counter current value) is transferred just once to D0.

(Ladder diagram representation)

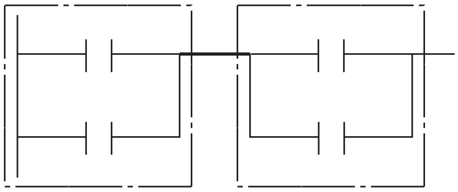


(4) Serial connection and parallel connection of circuit blocks

Symbol

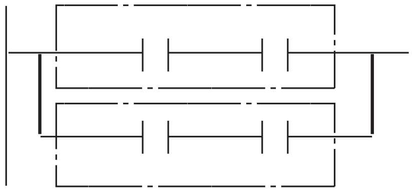
Serial connection of circuit blocks

ANB



Parallel connection of circuit blocks

ORB



Function

ANB

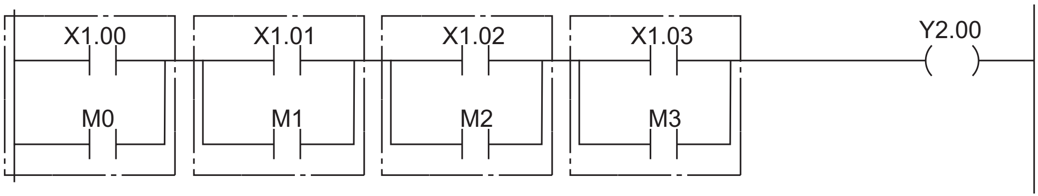
1) The ANB instruction ANDs block A and B. The result of this AND operation becomes the result of the instruction.

ORB

1) The ORB instruction ORs block A and B. The result of this OR operation becomes the result of the instruction.
2) The ORB connects circuit blocks that have two or more contacts each in parallel. Circuit blocks that have only one contact each can be connected in parallel using the OR or ORI instruction. The ORB instruction is not used for this purpose.

Program examples

ANB (Ladder diagram representation)

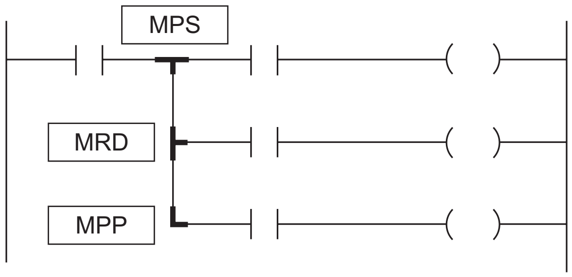


ORB (Ladder diagram representation)

(5) Push, read, and pop of operation results

Symbol

MPS :Push   MRD :Read   MPP :Pop of operation results



* “MPS”, “MRD”, and “MPP” are not shown in ladder diagrams.

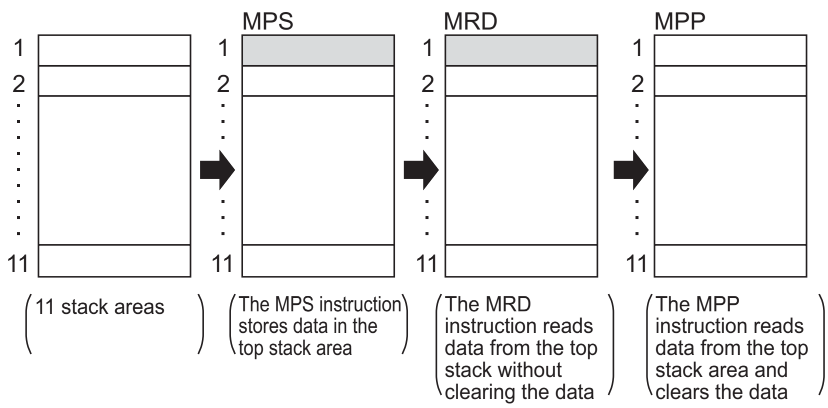
Function

MPS
1) The MPS instruction stores the result (ON or OFF state) of the operation preceding the MPS instruction.
2) Up to 11 MPS instructions can be written consecutively.

MRD
1) The MRD instruction reads the operation result stored by the MPS instruction. It then continues operation from the next step according to the result of the read operation.

MPP
1) The MPP instruction reads the operation result stored by the MPS instruction. It then continues operation from the next step according to the result of the read operation.
2) The MPP instruction clears the result stored by the MPS instruction.

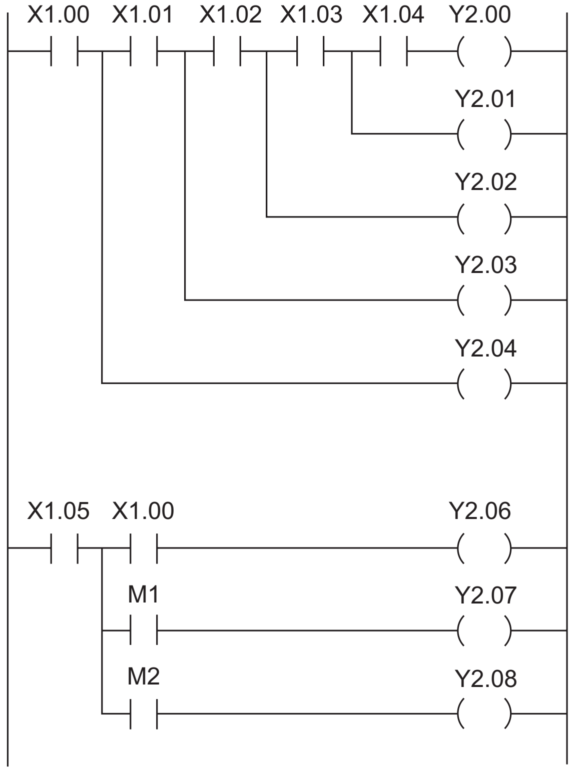
<Strage (stack) area operation>



Program examples

MPS , MRD , MPP

(Ladder diagram representation)

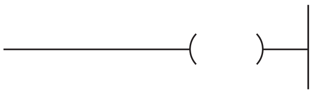


(6) Coil, inverted coil

Symbol

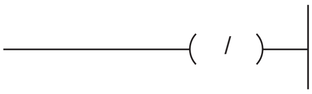
Coil

OUT



Inverted coil

OUTI



Available devices

Bit devices									
X	Y	M	L	SM	T	TR	C	F	V
–	○	○	○	○	–	–	–	○	○

Function

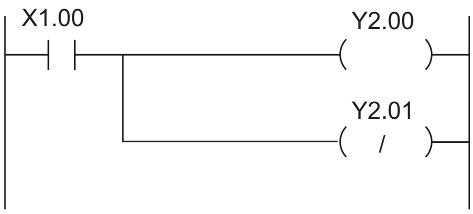
OUT Outputs the result of the operations up to the OUT instruction to the specified device.

OUTI Inverts the result of the operations up to the OUTI instruction and outputs the result to the specified device.

Program examples

OUT , OUTI When X1.00 goes ON, Y2.00 goes ON and Y2.01 goes OFF.

(Ladder diagram representation)

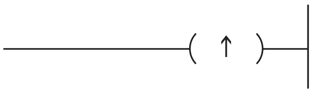


(7) Rising edge differential coil, falling edge differential coil

Symbol

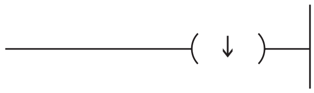
Rising edge differential coil

OUT+



Falling edge differential coil

OUT-



Available devices

Bit devices									
X	Y	M	L	SM	T	TR	C	F	V
–	○	○	○	○	–	–	–	○	○

Specifies the instance number (0 to 65535) of rising/falling edge differential coil.

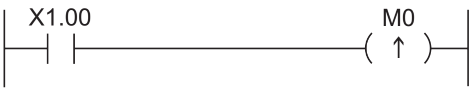
Function

OUT+ The specified device goes ON for one scan when the result of the operations up to the OUT+ instruction changes from OFF to ON.

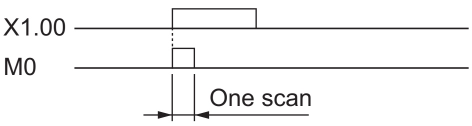
OUTI- The specified device goes ON for one scan when the result of the operations up to the OUT- instruction changes from ON to OFF.

Program examples

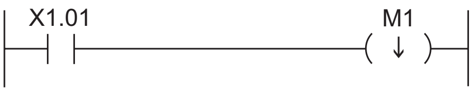
OUT+



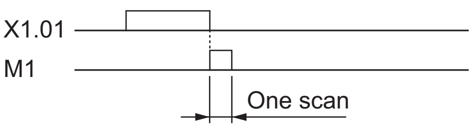
When X1.00 goes ON, this instruction detects the rising edge and sets M0 ON for one scan.



OUT-



When X1.01 goes OFF, this instruction detects the falling edge and sets M1 ON for one scan.

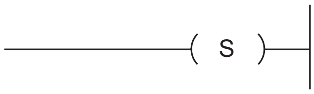


(8) Set, reset

Symbol

Set

SET



Reset

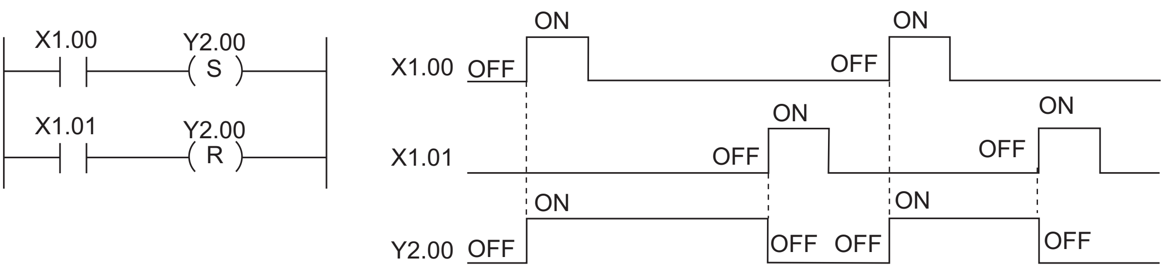
RST

Available devices

Bit devices									
X	Y	M	L	SM	T	TR	C	F	V
—	○	○	○	○	—	—	—	○	○

Function

SET 1) When the SET input goes ON, the specified device goes ON.
2) Devices once set can be set OFF with an RTS instruction.

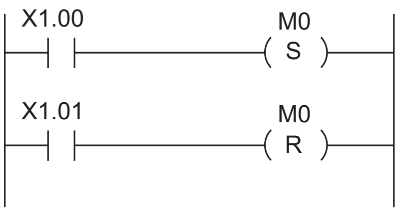


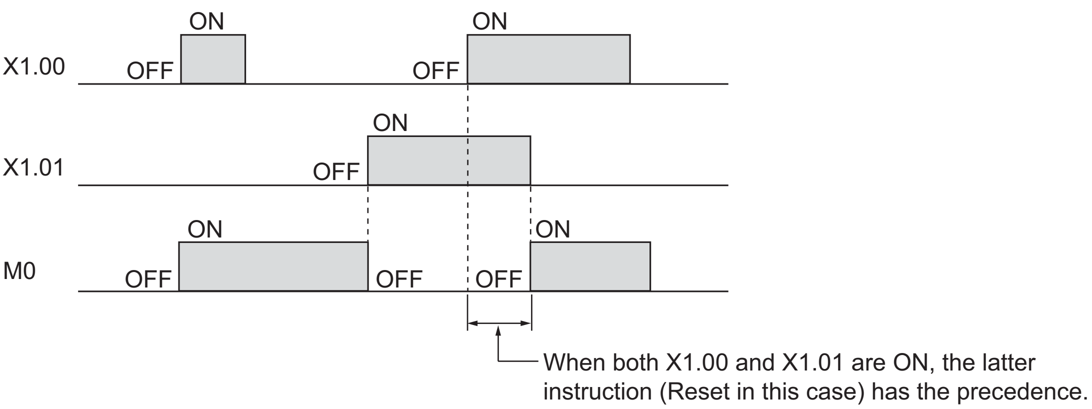
RST 1) When the RST input goes ON, the specified device goes OFF.

Program examples

1) Sets M0 ON when X1.00 goes ON and resets M0 OFF when X1.01 goes ON.

(Ladder diagram representation)

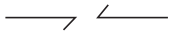




(9) Invert

Symbol

NOT



Function

NOT

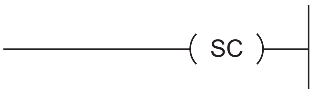
The NOT instruction inverts the value of the logical circuit existing on the left of it and passes the result to the logical circuit existing on the right of it.

(10) Step control

Symbol

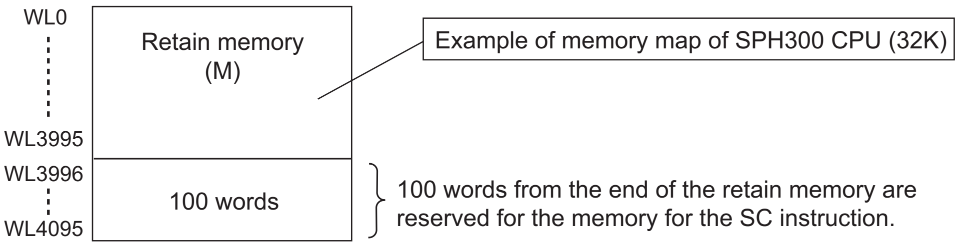
Step control

OUTSC



Available devices

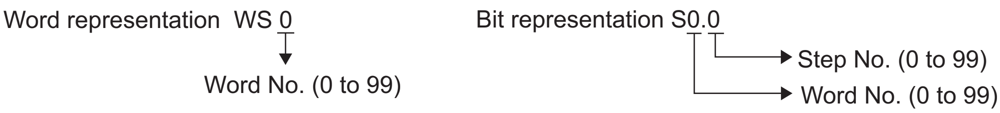
For devices for the step control (S**.**), 100 words are occupied from the end of the retain memory.
Devices other than “S**.**” are not available for the step control.



<Address representation of retain memory when using SC instruction>

The memory for the SC instruction uses the retain memory area. The address representation in a program is as follows:

Word representation	Bit representation
WS0	S00.00→S00.01→S00.02→ ------------------- →S00.99
WS1	S01.00→S01.01→S01.02→ ------------------- →S01.99
WS2	S02.00→S02.01→S02.02→ ------------------- →S02.99
⋮	⋮ ⋮
WS99	S99.00→S99.01→S99.02→ ------------------- →S99.99

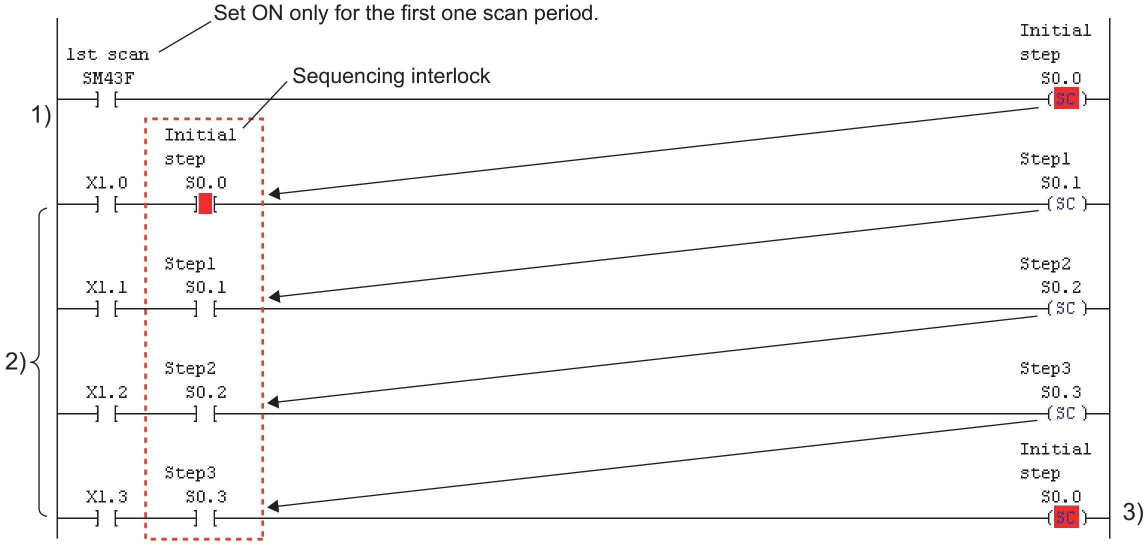


* Word data (WS**) indicates the step No. of an SC coil that is currently ON.
For example, when WS0 is 50 (BCD), it indicates that S0.50 is ON.
On the contrary, if BCD data ranging form 0 to 99 is entered into WS0, the SC coil of the No. is set ON.

Note: WS** is only available in MOVE instructions.

Program examples

<Sequencing circuit>



<Operation>

1) When the system starts operation, the initial step is set ON for the first scan time.
2) X1.0, X1.1, X1.2 and X1.3 are transition instructions to the next steps, however, the step coils of each row are not set on unless the sequencing interlock (S0.0 ro S0.3) of each line set on. Therefore, they are always set ON in the following order: S0.0 → S0.1 → S0.2 → S0.3 → ....
3) If X1.3 is set ON while S0.3 is ON, S0.0 is set ON and then control returns to the initial step.

Functions of SC instruction

The step control instruction (SC instruction) was created for step control. By using the step control instruction, a “sequencing circuit” or a “step process circuit” can be easily created.

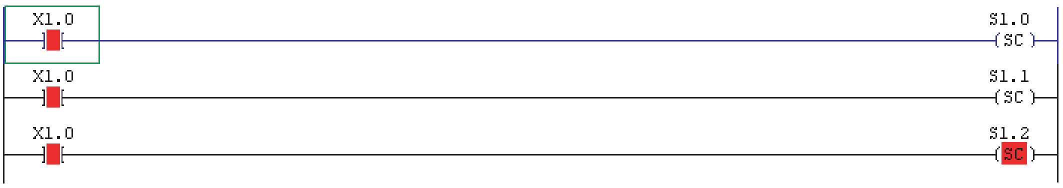
<Functions of SC instruction>

[Self holding function]
: Each SC coil has a built-in self-hold function.

[Interlock function]
: Each SC coil is interlocked and only one step goes on among 100 steps.

[Power-off step retention function]
: SC coils are provided in the retain memory and the memory is retained during a power-off conditions. Therefore, when the power comes back, operations start from the step at power-off.

[Subsequence priority]
: Even if there are two or more inputs in the same group, only one step of SC coil goes ON. (Interlock function)
If different SC coils in the same group are set ON in two or more lines, the later programmed line is given higher priority.
In the sample circuit below, although there are three SC coils that have the same ON condition, only the SC coil that is programmed last is ON.



3-2-2 Program control instructions

(1) Jump/jump end

Symbol

Unconditional jump

JMP — JMP P

Conditional jump

JMPC — JMPC P

Negative conditional jump

JMPCN — JMPCN P

Jump end

JEND — JEND P

P: Jump label number P: P0 to P65535

Function

JMP Jumps to the label position specified in the operand. The jumped instructions are not executed.

JMPC When the result of operations up to the current time is ON, jumps to the label position specified in the operand.
When the result of operations up to the current time is OFF, executes the next instruction.

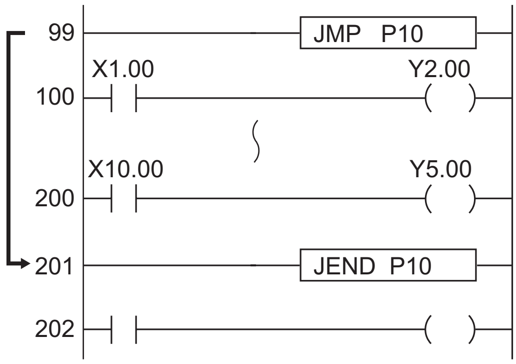
JMPCN When the result of operations up to the current time is OFF, jumps to the label position specified in the operand.
When the result of operations up to the current time is ON, executes the next instruction.

JEND Adds a jump label to an arbitrary position. This is used in pair with various JMPs.

Program examples

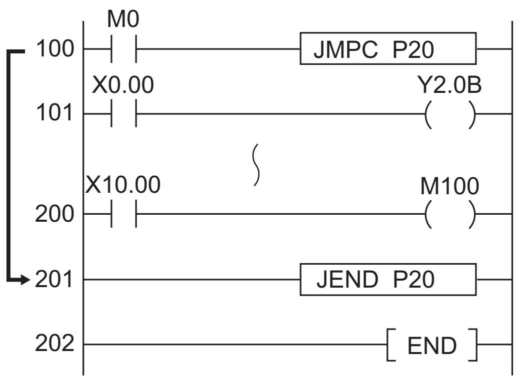
JMP Jumps unconditionally the 100th through 200th circuits in the program.

(Ladder diagram representation)



JMPC

(Ladder diagram representation)



(2) Loop, loop end

Symbol

Loop LOOP

Loop end CONT

S: Loop count (0 to 2147483647)

Available devices

	Word devices (W*)								Double-word devices (D*)								Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	
S	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	○	○

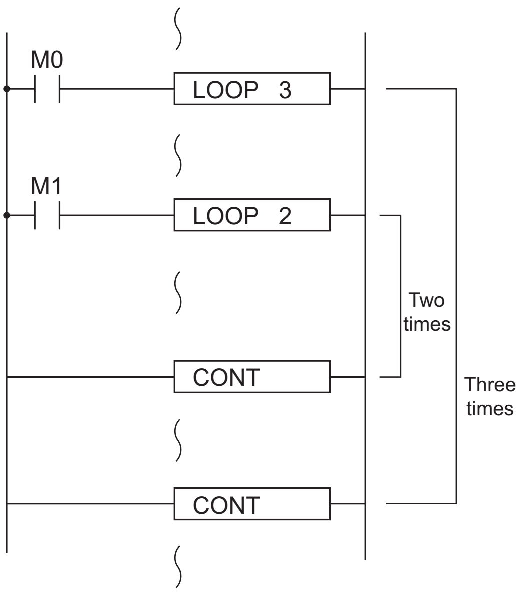
Function

1) After executing the LOOP through CONT instructions for “S” times (number of times specified in S), executes the steps after CONT.
2) Nesting of up to 8 levels is allowed.
3) The allowable range of “S” is 0 to 2147483647. When a minus value is specified in “S,” S=1 is assumed.
4) A program error occurs if the CONT instruction is placed before a LOOP instruction, no CONT instruction exists, the CONT instruction is placed after FEND or END instruction, the number of LOOP instructions does not match that of CONT instructions or a nesting-over occurs.
5) During execution of “LOOP - CONT,” changing the loop count is ineffective.

Note: When the operation takes a long time for a high loop count, a watch-dog timer error may occur.

Program examples

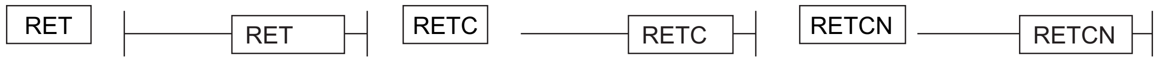
(Ladder diagram representation)



(3) Return instruction

Symbol

Unconditional return | Conditional return | Negative conditional return



Function

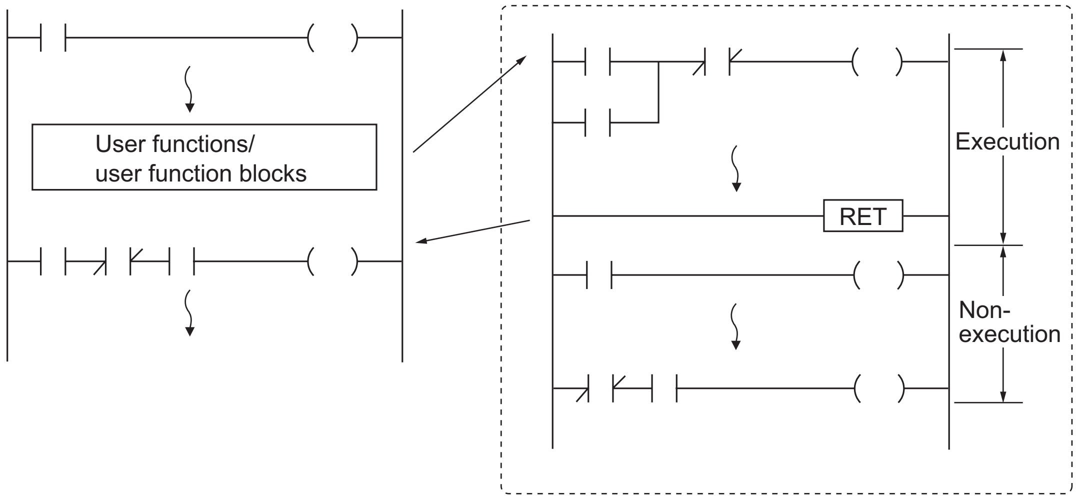
RET
1) When this is used in a user function/user function block, unconditionally returns to the step succeeding the calling position of the function.
2) When this is used in a program, returns unconditionally to the last position of the program.

RETC
1) When the result of the operations up to the current time is ON, executes "- [RET] -."
2) When the result of the operations up to the current time is OFF, executes the next instruction.

RETCN
1) When the result of the operations up to the current time is OFF, executes "- [RET] -."
2) When the result of the operations up to the current time is ON, executes the next instruction.

<Operation>

Execution of RET instruction | User functions/user function blocks



(4) Master control set, reset

Symbol

Master control set

MC

[MC N D]

Master control reset

MCR

[MCR N]

N: Nesting No. (0 to 7), D: Device to be set ON

Available devices

	Bit device										Constants
	X	Y	M	L	SM	T	TR	C	F	V	
N	–	–	–	–	–	–	–	–	–	–	○
D	–	○	○	○	–	–	–	–	○	○	–

Function

MC

1) The MC instruction is the master control start instruction. If the MC condition contact is set ON, operations between the MC and MCR instructions are executed normally. If the MC condition contact is set OFF, operations between the MC and MCR instructions are executed as shown below.

Timer	Counter	OUT instruction	SET, RST
The timer value is reset to zero.	The counter current value is held.	All outputs are set OFF.	The current status is held.

2) Up to eight MC or MCR instructions (N0 to N7) can be nested. Nested MC instructions are executed in ascending order of instruction numbers. (N). Nested MCR instructions are executed in descending order of instruction numbers (N).
3) The instructions between the MC or MCR instructions are executed regardless of the states (ON or OFF) of MC condition contacts.
4) The MC instruction can be used repeatedly by changing the device specification (D).
5) When the MC condition contact is set ON, the device specified by D is set ON. If the same device is used by an OUT or other instruction, duplicate use of a coil is assumed.

MCR

1) The MCR instruction is the master control reset instruction. The MCR instruction is the end of master control.
2) MC instructions having the specified nesting number (N) and subsequent nesting numbers are reset.

[MCR N5] MC instructions of nested number 5, 6, and 7 are reset.

(5) Section instruction

Symbol

SECTION

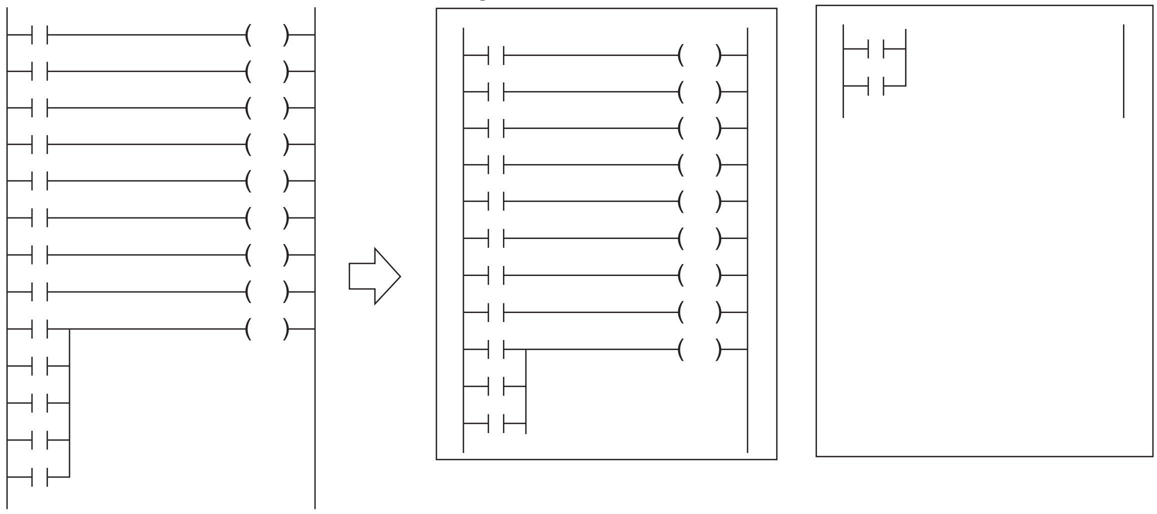
├── SECTION N ──┤ N: 1 to 65535

Function

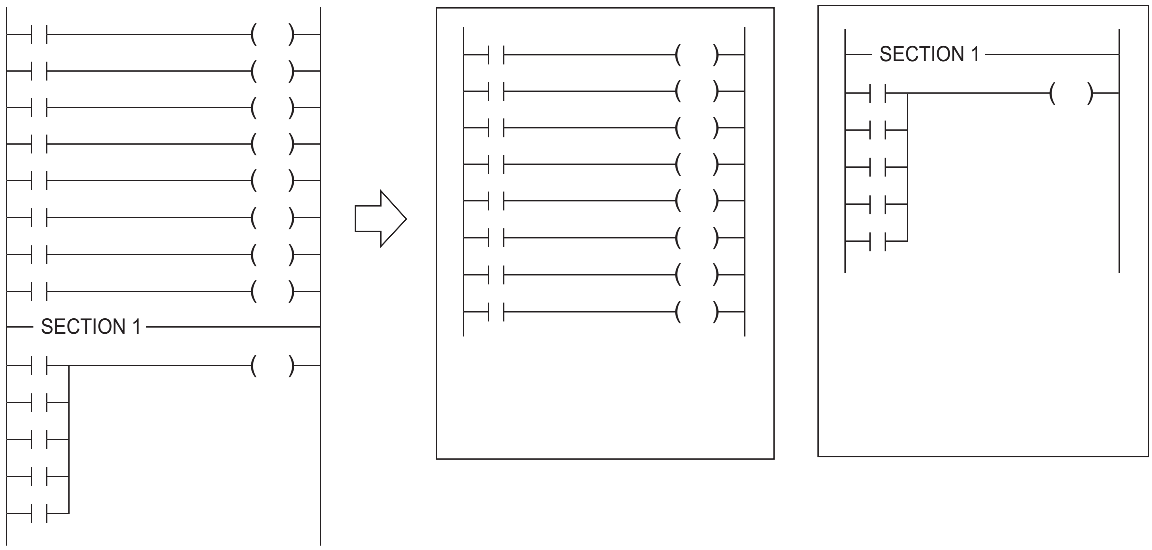
1) When the SECTION instruction is used, a ladder program is divided at arbitrary circuit when printed.
2) The SECTION instruction have no negative effect on the program execution time.

Ladder program Print program

There is no SECTION instruction. Page 1 Page 2



There is a SECTION instruction. Page 1 Page 2



3-2-3 Conversion instructions

(1) BIN to BCD (BCD)

Symbol

BCD

BCD S → D

S: Device where BIN data to be converted is stored
D: Device where converted BCD data is to be stored

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Function

BCD

1) Converts the BIN data (16 or 32 bits) of the device specified in “S” to BCD data and outputs the result to the device specified in “D.”

2147483648 ---------------------- 32768 ------------------ 8421

S	0000	0001	0100	0011	0101	1110	0000	0110	(BIN)

BCD conversion

D	0010 (2)	0001 (1)	0001 (1)	1001 (9)	0010 (2)	0001 (1)	1001 (9)	1000 (8)	(BCD)
	Position of 10000000	Position of 1000000	Position of 100000	Position of 10000	Position of 1000	Position of 100	Position of 10	Position of 1	

2) Values specified in “S” and “D” must have the same bit width.
3) The data range is:
   When word device is specified: 0 to 9999
   When double word device is specified: 0 to 99999999
   “9999” (when a word device is specified) or “99999999” (when a double-word device is specified) is output if BIN data exceeds the specified data range.

(2) BCD to BIN (BIN)

Symbol

BIN

| BIN S → D |

S: Device where BCD data to be converted is stored
D: Device where converted BIN data is to be stored

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Function

BIN

1) Converts the BCD data (0 to 9999 for 16-bit data or 0 to 99999999 for 32-bit data) in the device specified in “S” and outputs the result to the device specified in “D.”

S (bits 31 … 16 | 15 … 0)

31			16	15			0
0010 (2)	0001 (1)	0001 (1)	1001 (9)	0010 (2)	0001 (1)	1001 (9)	1000 (8)
Position of 10000000	Position of 1000000	Position of 100000	Position of 10000	Position of 1000	Position of 100	Position of 10	Position of 1

↓ BIN conversion

D (bits 31 … 16 | 15 … 0)

31			16	15			0
0000	0001	0100	0011	0101	1110	0000	0110

2) Values specified in “S” and “D” must have the same bit width.

(3) BIN to real (□I_TO_R)

Symbol

I_TO_R	S → D	: Signed integer 16 bits
DI_TO_R	S → D	: Signed integer 32 bits
UI_TO_R	S → D	: Unsigned integer 16 bits
UDI_TO_R	S → D	: Unsigned integer 32 bits

S: Device where the BIN data to be converted is stored
D: Device where the conversion result is to be stored

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	○	○	–

Function

1) Converts the BIN integer data in specified device "S" to real-type (REAL) data and outputs the conversion result to the device specified in "D."
2) The number of significant digits after conversion is 6.

<Operation>

I_TO_R , DI_TO_R

S (BIN)	Conversion	D (REAL)
32767	⇨	3.2767E + 04
		ENO = 1
– 32768	⇨	– 3.2768E + 04
		ENO = 1

UI_TO_R , UDI_TO_R

S (BIN)	Conversion	D (REAL)
65535	⇨	6.5535E + 04
		ENO = 1
0	⇨	– 0.0E + 00
		ENO = 1

[References]

The data ranges are as follows:

- BIN integer (signed 16 bits) : –32,768 to 32,767
- BIN integer (unsigned 16 bits) : 0 to 65,535
- BIN integer (signed 32 bits) : –2,147,483,648 to 2,147,483,647
- BIN integer (unsigned 32 bits) : 0 to 4,294,967,295
- Real number : $-2^{128} < N <= -2^{-126}$, 0, $2^{-126} <= N < 2^{128}$

(4) Real to BIN integer (R_TO_☐ I)

Symbol

R_TO_I S → D : Signed integer 16 bits

R_TO_DI S → D : Signed integer 32 bits

R_TO_UI S → D : Unsigned integer 16 bits

R_TO_UDI S → D : Unsigned integer 32 bits

S: Device where the real data to be converted is stored
D: Device where the conversion result is to be stored

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	–	–	–	–	–	–	–	–	○	○	○	○	○	○	○	○	○	○
D	○	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Function

1) Converts the BIN integer data in specified device "S" to real-type (REAL) data and outputs the conversion result to the device specified in "D."
2) When the data converted from real type data exceeds the data range of the device specified in "D," outputs the boundary value of the data range to "D."
3) Rounds off the fraction below decimal point.

<Operation>

R_TO_I

- When the operation result is within the INT type range

S (REAL) 6.789E + 02 ⇨ Conversion ⇨ D (INT) 679
ENO = 1

- When the operation result exceeds the INT type range

S (REAL) 4.500E + 04 ⇨ Conversion ⇨ D (INT) 32767
ENO = 0

S (REAL) –5.000E + 04 ⇨ Conversion ⇨ D (INT) –32768
ENO = 0

R_TO_UI

- When the operation result is within the UINT type range

S (REAL) 6.789E + 02 ⇨ Conversion ⇨ D (UINT) 679
ENO = 1

- When the operation result exceeds the UINT type range

S (REAL) 1.234E + 05 ⇨ Conversion ⇨ D (UINT) 65535
ENO = 0

S (REAL) –2.345E + 02 ⇨ Conversion ⇨ D (UINT) 0
ENO = 0

R_TO_DI

- When the operation result is within the DINT type range

S (REAL) 6.789E + 02 ⇨ Conversion ⇨ D (DINT) 679
ENO = 1

- When the operation result exceeds the DINT type range

S (REAL) 2.147E + 20 ⇨ Conversion ⇨ D (DINT) 2147483647
ENO = 0

S (REAL) –2.147E + 20 ⇨ Conversion ⇨ D (DINT) –2147483648
ENO = 0

R_TO_UDI

- When the operation result is within the UDINT type range

S (REAL) 6.789E + 02 ⇨ Conversion ⇨ D (UDINT) 679
ENO = 1

- When the operation result exceeds the UDINT type range

S (REAL) 4.294E + 10 ⇨ Conversion ⇨ D (UDINT) 4294967295
ENO = 0

S (REAL) –3.456E + 03 ⇨ Conversion ⇨ D (UDINT) 0
ENO = 0

[References]

The data ranges are as follows:

- BIN integer (signed 16 bits) : –32,768 to 32,767
- BIN integer (unsigned 16 bits) : 0 to 65,535
- BIN integer (signed 32 bits) : –2,147,483,648 to 2,147,483,647
- BIN integer (unsigned 32 bits) : 0 to 4,294,967,295
- Real number : $-2^{128} < N <= -2^{-126}$, 0, $2^{-126} <= N < 2^{128}$

(5) Real to integer (TRUNC_☐ I)

Symbol

TRUNC_I	S → D	: Signed integer 16 bits
TRUNC_DI	S → D	: Signed integer 32 bits
TRUNC_UI	S → D	: Unsigned integer 16 bits
TRUNC_UDI	S → D	: Unsigned integer 32 bits

S: Device where the real data to be converted is stored
D: Device where the conversion result is to be stored

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
D	○	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Function

1) Converts the BIN integer data in specified device “S” to real-type (REAL) data and outputs the conversion result to the device specified in “D.”
2) When the data converted from real type data exceeds the data range of the device specified in “D,” outputs the boundary value of the data range to “D.”
3) Rounds off the fraction below decimal point.

<Operation>

TRUNC_I

- When the operation result is within the INT type range

S (REAL)	Conversion	D (INT)
6.789E + 02	⇨	678
		ENO = 1

- When the operation result exceeds the INT type range

S (REAL)	Conversion	D (INT)
4.000E + 04	⇨	32767
		ENO = 0

S (REAL)	Conversion	D (INT)
–5.000E + 04	⇨	–32768
		ENO = 0

TRUNC_UI

- When the operation result is within the UINT type range

S (REAL)	Conversion	D (UINT)
6.789E + 02	⇨	678
		ENO = 1

- When the operation result exceeds the UINT type range

S (REAL)	Conversion	D (UINT)
7.000E + 04	⇨	65535
		ENO = 0

S (REAL)	Conversion	D (UINT)
–5.000E + 04	⇨	0
		ENO = 0

TRUNC_DI

- When the operation result is within the DINT type range

S (REAL)	Conversion	D (DINT)
6.789E + 02	⇨	678
		ENO = 1

- When the operation result exceeds the DINT type range

S (REAL)	Conversion	D (DINT)
2.147E + 10	⇨	2147483647
		ENO = 0

S (REAL)	Conversion	D (DINT)
–2.147E + 100	⇨	–2147483648
		ENO = 0

TRUNC_UDI

- When the operation result is within the UDINT type range

S (REAL)	Conversion	D (UDINT)
6.789E + 02	⇨	678
		ENO = 1

- When the operation result exceeds the UDINT type range

S (REAL)	Conversion	D (UDINT)
4.294E + 10	⇨	4294967295
		ENO = 0

S (REAL)	Conversion	D (UDINT)
–50000.0	⇨	0
		ENO = 0

[References]

The data ranges are as follows:

- BIN integer (signed 16 bits) : –32,768 to 32,767
- BIN integer (unsigned 16 bits) : 0 to 65,535
- BIN integer (signed 32 bits) : –2,147,483,648 to 2,147,483,647
- BIN integer (unsigned 32 bits) : 0 to 4,294,967,295
- Real number : $-2^{128} < N <= -2^{-126}$, 0, $2^{-126} <= N < 2^{128}$

(6) INT to DINT (I_TO_DI)

Symbol

I_TO_DI

—| I_TO_DI S → D |—

S: Word device to be converted
D: Double-word device where the conversion result is to be stored

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	○
D	–	–	–	–	–	–	–	–	–	○	○	○	–	○	○	○	○	–

Function

1) Converts 16-bit signed integer (INT-type data) to 32-bit signed integer (DINT-type data).

<Operation>

S (INT)	Conversion	D (DINT)
32767	⇨	32767
EN = 1		ENO = 1

S (INT)	Conversion	D (DINT)
-32767	⇨	-32768
EN = 1		ENO = 1

(7) DINT to INT (DI_TO_I)

Symbol

DI_TO_I

—| DI_TO_I S → D |—

S: Double-word device to be converted
D: Word device where the conversion result is to be stored

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	–	–	–	–	–	–	–	–	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	–

Function

<Operation>

- When the operation result is within the INT type range

DINT	Conversion	INT
32767	⇨	32767
EN = 1		ENO = 1

- When the operation result exceeds the INT type range

DINT	Conversion	INT
40000	⇨	32767
EN = 1		ENO = 0

DINT	Conversion	INT
-50000	⇨	-32768
EN = 1		ENO = 0

3-2-4 Arithmetic operation instructions

(1) Addition (ADD)

Symbol

ADD

ADD S1, S2 → D

ADD_UAI

ADD_UAI S1, S2 → D

ADD_R

ADD_R S1, S2 → D

S1: Data to be operated or the device storing that data
S2: Data to be operated or the device storing that data
D : Device storing the result

ADD : Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
ADD_UAI : Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
ADD_R : Real type (single-precision floating point type) [REAL]

Available devices

	Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Note: When the data type is real, only double-word devices are available.

Function

1) Adds input data “S1” and “S2” and outputs the result to “D.”
2) Input devices “S1” and “S2” and output device “D” must have the same data type and the same bit width.
3) When the operation result exceeds the range of the data type, ENO goes to 0.
4) When the data type is real and it goes very close to 0 to an extent that cannot be expressed as real data, the output goes to 0 and ENO to 1.

<Operation>

- When the operation result is within the range of the data type

S1 (INT)		S2 (INT)	Operation	D (INT)
1234	+	5678	⇨	6912
				ENO = 1

- When the operation result exceeds the range of the data type

S1 (INT)		S2 (INT)	Operation	D (INT)
32767	+	32767	⇨	–2
				ENO = 0

S1 (INT)		S2 (INT)	Operation	D (INT)
–32768	+	–32768	⇨	0
				ENO = 0

Note: When data not of the real type exceeds the range of the data type, no boundary value processing is performed. Take care so that the operation result does not exceed the data type range. For real-type data, boundary value processing is performed.

(2) Subtraction (SUB)

Symbol

SUB

SUB S1, S2 → D

SUB_R

SUB_R S1, S2 → D

SUB_UAI

SUB_UAI S1, S2 → D

S1: Data to be operated or the device storing that data
S2: Data to be operated or the device storing that data
D : Device storing the result

SUB : Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
SUB_UAI : Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
SUB_R : Real type (single-precision floating point type) [REAL]

Available devices

	Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Note: When the data type is real, only double-word devices are available.

Function

1) Subtracts “S2” from input data “S1” and outputs the result to “D.”
2) Input devices “S1” and “S2” and output device “D” must have the same data type and the same bit width.
3) When the operation result exceeds the range of the data type, ENO goes to 0.
4) When the data type is real and it goes very close to 0 to an extent that cannot be expressed as real data, the output goes to 0 and ENO to 1.

<Operation>

- When the operation result is within the range of the data type

S1 (INT)		S2 (INT)	Operation	D (INT)
5678	–	1234	⇨	4444
				ENO = 1

- When the operation result exceeds the range of the data type

S1 (INT)		S2 (INT)	Operation	D (INT)
–32768	–	50	⇨	32718
				ENO = 0

Note: When data not of the real type exceeds the range of the data type, no boundary value processing is performed. Take care so that the operation result does not exceed the data type range. For real-type data, boundary value processing is performed.

(3) Multiplication (MUL)

Symbol

MUL

MUL S1, S2 → D

MUL_R

MUL_R S1, S2 → D

MUL_UAI

MUL_UAI S1, S2 → D

S1: Data to be operated or the device storing that data
S2: Data to be operated or the device storing that data
D : Device storing the result

MUL : Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
MUL_UAI : Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
MUL_R : Real type (single-precision floating point type) [REAL]

Available devices

	Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Note: When the data type is real, only double-word devices are available.

Function

1) Multiplies input data “S1” by “S2” and outputs the result to “D.”
2) Input devices “S1” and “S2” and output device “D” must have the same data type and the same bit width.
3) When the operation result exceeds the boundary value of the data type, the boundary value of the data type is output and ENO goes to 0.
4) When the data type is real and it goes very close to 0 to an extent that cannot be expressed as real data, the output goes to 0 and ENO to 1.

<Operation>

- When the operation result is within the range of the data type

S1 (INT)		S2 (INT)	Operation	D (INT)
222	×	10	⇨	2220
				ENO = 1

- When the operation result exceeds the range of the data type

S1 (INT)		S2 (INT)	Operation	D (INT)
32767	×	32767	⇨	32767
				ENO = 0

S1 (INT)		S2 (INT)	Operation	D (INT)
–32768	×	32768	⇨	–32768
				ENO = 0

(4) Division (DIV)

Symbol

DIV

DIV S1, S2 → D

DIV_R

DIV_R S1, S2 → D

DIV_UAI

DIV_UAI S1, S2 → D

S1: Data to be operated or the device storing that data
S2: Data to be operated or the device storing that data
D : Device storing the result

DIV : Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
DIV_UAI : Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
DIV_R : Real type (single-precision floating point type) [REAL]

Available devices

	Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Note: When the data type is real, only double-word devices are available.

Function

1) Divides input data "S1" by "S2" and stores the result in "D." For the quotient of the integer type, any digits under the decimal point are truncated.
2) Input devices "S1" and "S2" and output device "D" must have the same data type and the same bit width.
3) When the operation result exceeds the boundary value of the data type, the boundary value of the data type is assumed to be the operation result and ENO goes to 0.
4) When the data type is real and it goes very close to 0 to an extent that cannot be expressed as real data, the output goes to 0 and ENO to 1.
5) When the divisor is 0, the maximum value matching the sign of the dividend is output and ENO goes to 0.

<Operation>

- When the operation result is within the range of the data type

S1 (INT)		S2 (INT)	Operation	D (INT)
10	÷	3	⇨	3
				ENO = 1

- When the divisor is 0

S1 (INT)		S2 (INT)	Operation	D (INT)
3940	÷	0	⇨	0
				ENO = 0

- When the quotient exceeds the range of the data type

S1 (INT)		S2 (INT)	Operation	D (INT)
–32768	÷	–1	⇨	0
				ENO = 0

(5) Division remainder (MOD)

Symbol

MOD

—[MOD S1, S2 → D]—

MOD_UAI

—[MOD_UAI S1, S2 → D]—

S1: Data to be operated or the device storing that data
S2: Data to be operated or the device storing that data
D : Device storing the result

MOD : Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
MOD_UAI : Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
MOD_R : Real type (single-precision floating point type) [REAL]

Available devices

	Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	—	○	○	○	—	○	○	○	—	○	○	○	—	○	○	○	○	—

Note: When the data type is real, only double-word devices are available.

Function

1) Divides input data "S1" by "S2" and outputs the remainder to "D."
2) Input devices "S1" and "S2" and output device "D" must have the same data type and the same bit width.
3) If the quotient exceeds the boundary value of the data type, the output goes to 0 and ENO to 0.
4) When the divisor is 0, the output goes to 0 and ENO to 0.
5) Operation is performed so that (divisor) x (quotient) + (remainder) = (dividend).

<Operation>

- When the operation result is within the range of the data type

S1 (INT)		S2 (INT)	Operation	D (INT)
234	MOD	4	⇨	2
				ENO = 1

- When the divisor is 0

S1 (INT)		S2 (INT)	Operation	D (INT)
3940	MOD	0	⇨	0
				ENO = 0

- When the quotient exceeds the range of the data type

S1 (INT)		S2 (INT)	Operation	D (INT)
–32768	MOD	–1	⇨	0
				ENO = 0

(6) Exponent (EXPT)

Symbol

EXPT

EXPT S1, S2 → D

S1: Data to be operated or the device storing that data
S2: Data to be operated or the device storing that data
D : Device storing the result

Available devices

	Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	–	○	○	○	○	○
S2	○	○	○	○	–	○	○	○	○	○
D	–	○	○	○	–	–	–	○	○	–

Function

1) Exponentiates input data “S1” (base) by “S2” (exponent) and outputs the result to “D.”
2) When the operation result exceeds the boundary value of the real type, the output goes to 0 and ENO to 0.
3) The base is >= 0. When the base is < 0, the output goes to 0 and ENO to 0.
4) The number of significant digits in the output is 4.
5) When the output value exceeds the boundary value of the real type, ENO goes to 0. When the output value goes very close to 0 to an extent that cannot be expressed as real data, the output goes to 0 and ENO to 1.
6) When the base = 0 and exponent = 0 (0), the output goes to 1 and ENO to 1.

<Operation>

- When the operation result is within the range of the data type

S1 (REAL)		S2 (REAL)	Operation	D (REAL)
1.230E + 01	EXPT	2.500E + 00	⇨	5.306E + 02
				ENO = 1

- When the operation result exceeds the range of the data type

S1 (REAL)		S2 (REAL)	Operation	D (REAL)
1.230E + 01	EXPT	2.500E + 02	⇨	3.402 + 38
				ENO = 0

(7) Absolute value (ABS)

Symbol

ABS : Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
ABS_R : Real type (single-precision floating point type) [REAL]

S : Data to be operated or the device storing that data
D : Device storing the result

Available devices

	Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	–	○	○	○	○	○	○	○	–	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Note: When the data type is real, only double-word devices are available.

Function

1) Obtains the absolute value of input data “S” and outputs the result to “D.”
2) When the input value is the maximum minus value, outputs the maximum plus value of the output data type.
3) Input devices “S” and output device “D” must have the same data type and the same bit width.

<Operation>

S (INT)	Operation	D (INT)
–20	⇨	20
		ENO = 1

• When the input value is the maximum minus value (–32768)

S (INT)	Operation	D (INT)
–32768	⇨	32767
		ENO = 0

(8) Square root (SQRT)

Symbol

SQRT

[SQRT S → D]

S : Data to be operated or the device storing that data
D : Device storing the result

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	–	–	–	–	–	–	–	–	○	○	○	○	○	–	○	○	○	○
D	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	○	○	–

Function

1) Obtains the square root of input data "S" (real type, (single-precision floating point type) [REAL]) and outputs the result to "D."
2) When the input value is minus, outputs 0 and ENO goes to 0.
3) The number of significant digits of the output is 5.

<Operation>

S (REAL)	Operation	D (REAL)
1.234E + 07	⇨	3.5128E + 03
		ENO = 1

- When the input value is minus

S (REAL)	Operation	D (REAL)
–1.234E + 07	⇨	0.0E + 00
		ENO = 0

(9) Natural logarithm (LN)

Symbol

LN

LN S → D

S : Data to be operated or the device storing that data
D : Device storing the result

Available devices

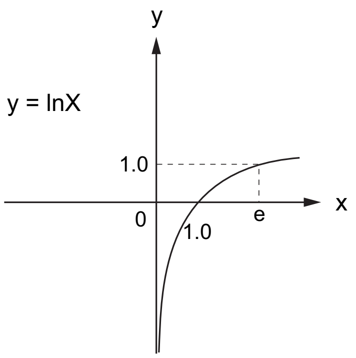
	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
D	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	○	○	–

Function

1) Obtains the square root of input data “S” (real type, (single-precision floating point type) [REAL]) and outputs the result to “D.”
2) When the input value is minus, outputs 0 and ENO goes to 0.
3) When the input value is 0, the output goes to the maximum minus value and ENO goes to 0.
4) The number of significant digits of the output is 4.

<Operation>

S (REAL)	Operation	D (REAL)
1.234E + 03	⇨	7.118E + 00
		ENO = 1



(10) Common logarithm (LOG)

Symbol

LOG

LOG S → D

S : Data to be operated or the device storing that data
D : Device storing the result

Available devices

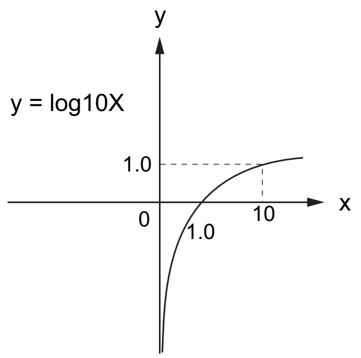
	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	—	—	—	—	—	—	—	—	○	○	○	○	—	○	○	○	○	○
D	—	—	—	—	—	—	—	—	—	○	○	○	—	—	—	○	○	—

Function

1) Obtains the square root of input data “S” (real type, (single-precision floating point type) [REAL]) and outputs the result to “D.”
2) When the input value is minus, outputs 0 and ENO goes to 0.
3) When the input value is 0, the output goes to the maximum minus value and ENO goes to 0.
4) The number of significant digits of the output is 4.

<Operation>

S (REAL)	Operation	D (REAL)
1.234E + 03	⇨	3.091E + 00
		ENO = 1



(11) Exponential (EXP)

Symbol

EXP

EXP S → D

S : Data to be operated or the device storing that data
D : Device storing the result

Available devices

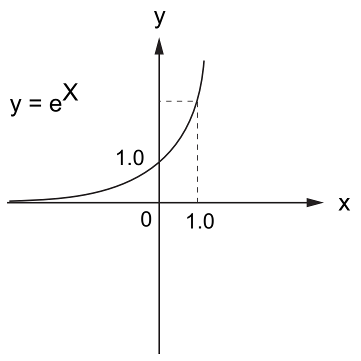
	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	—	—	—	—	—	—	—	—	○	○	○	○	—	○	○	○	○	○
D	—	—	—	—	—	—	—	—	—	○	○	○	—	—	—	○	○	—

Function

1) Performs exponential operation on “S” (real type (single-precision floating point type) [REAL]) using 2.718281 as the base (e) and outputs the result to “D.”
2) When the operation result exceeds the boundary value of real type, the boundary value is output.
3) Number of significant digits of output:
   - When the input/output is within the range of –64 to 64... 4
   - Other than above... Error will increase.
4) When the output value exceeds the boundary value of the real type, ENO goes to 0. When the output value goes very close to 0 to an extent that cannot be expressed as real data, the output goes to 0 and ENO to 1.

<Operation>

S (REAL)	Operation	D (REAL)
2.0E + 00	⇨	7.389E + 00
		ENO = 1



(12) Sine (SIN)

Symbol

SIN

SIN S → D

S : Data to be operated or the device storing that data
D : Device storing the result

Available devices

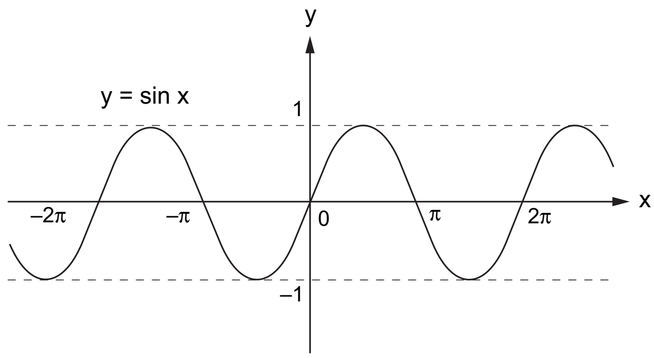
	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
D	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	○	○	–

Function

1) Performs sine operation on “S” (real type (single-precision floating point type) [REAL]) and outputs the result to “D.”
2) The unit of input is radian (angle × π/180).
3) When the input is in the range of –2π to 2π, the number of significant digits of output is 5 (up to four places of decimals). When the absolute value of input is 2π or more, the operation is performed but error will increase.
4) When (input) < -2^{31} or (input value) > $2^{31}-1$, the output value goes to 0 and ENO to 0.

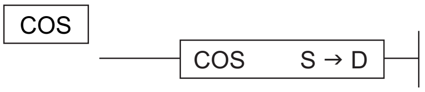
<Operation>

S (REAL)	Operation	D (REAL)
3.141E + 00	⇨	5.926E – 04
		ENO = 1



(13) Cosine (COS)

Symbol



S : Data to be operated or the device storing that data
D : Device storing the result

Available devices

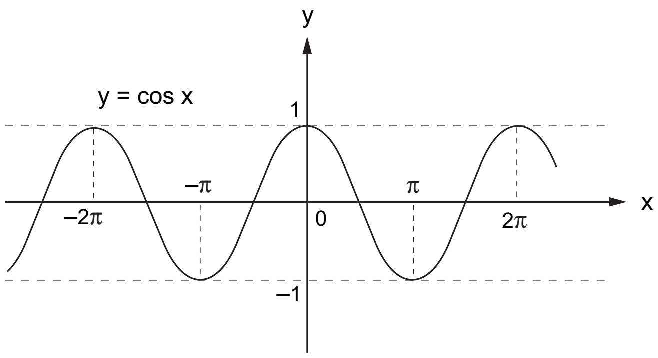
	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	—	—	—	—	—	—	—	—	○	○	○	○	—	○	○	○	○	○
D	—	—	—	—	—	—	—	—	—	○	○	○	—	—	—	○	○	—

Function

1) Performs cosine operation on “S” (real type (single-precision floating point type) [REAL]) and outputs the result to “D.”
2) The unit of input is radian.
3) When the input is in the range of -2π to 2π, the number of significant digits of output is 5 (up to four places of decimals). When the absolute value of input is 2π or more, the operation is performed but error will increase.
4) When (input) < -2^{31} or (input value) > $2^{31}-1$, the output value goes to 0 and ENO to 0.

<Operation>

S (REAL)	Operation	D (REAL)
1.047E + 00	⇨	5.001E – 01
		ENO = 1



(14) Tangent (TAN)

Symbol

TAN

TAN S → D

S : Data to be operated or the device storing that data
D : Device storing the result

Available devices

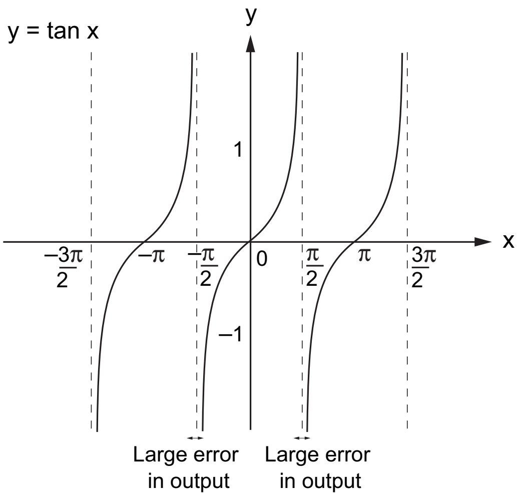
	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
D	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	○	○	–

Function

1) Performs tangent operation on “S” (real type (single-precision floating point type) [REAL]) and outputs the result to “D.”
2) The unit of input is radian.
3) When the input is in the range of -2π to 2π, the number of significant digits of output is 4, but error in the output will increase near an integral multiple of $\pi/2$. When the absolute value of input is 2π or more, the operation is performed but error will increase.
4) When (input) < -2^{31} or (input value) > $2^{31}-1$, the output value goes to 0 and ENO to 0.

<Operation>

S (REAL)	Operation	D (REAL)
7.854E – 01	⇨	1.000E + 00
		ENO = 1



(15) Arcsine (ASIN)

Symbol

ASIN

ASIN S → D

S : Data to be operated or the device storing that data
D : Device storing the result

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
D	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	○	○	–

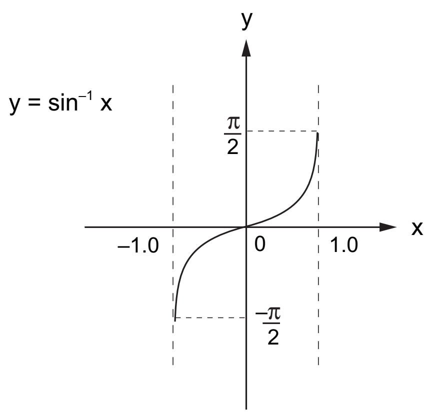
Function

1) Performs arcsine operation on “S” (real type (single-precision floating point type) [REAL]) and outputs the result to “D.”
2) The unit of output is radian.
3) The input range is –1.0 to +1.0 and the output range is $-\pi/2$ to $\pi/2$. When the input exceeds this range, the output goes to 0 and ENO to 0.
4) Number of significant digits of output:
When | input value | = 1.0 or 0.998999 or less ... 4 digits
When | input value | = 0.999 to 0.999999 ... Error will increase.

<Operation>

S (REAL) 7.854E – 01 → Operation → D (REAL) 9.033E – 01

ENO = 1



(16) Arccosine (ACOS)

Symbol

ACOS

—[ACOS S → D]—

S : Data to be operated or the device storing that data
D : Device storing the result

Available devices

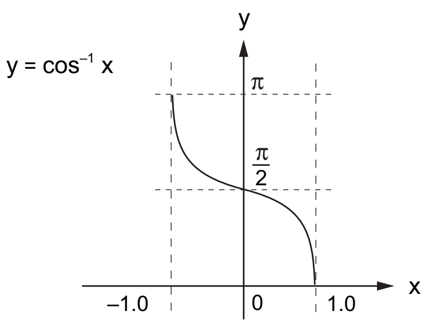
	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	—	—	—	—	—	—	—	—	○	○	○	○	—	○	○	○	○	○
D	—	—	—	—	—	—	—	—	—	○	○	○	—	—	—	○	○	—

Function

1) Performs arccosine operation on “S” (real type (single-precision floating point type) [REAL]) and outputs the result to “D.”
2) The unit of output is radian.
3) The input range is –1.0 to +1.0 and the output range is π to 0. When the input exceeds this range, the output goes to 0 and ENO to 0.
4) Number of significant digits of output:
When | input value | = 1.0 or 0.998999 or less... 4 digits
When | input value | = 0.999 to 0.999999... Error will increase.

<Operation>

S (REAL)	Operation	D (REAL)
7.854E – 01	⇨	6.675E – 01
		ENO = 1



(17) Arctangent (ATAN)

Symbol

ATAN

ATAN S → D

S : Data to be operated or the device storing that data
D : Device storing the result

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
D	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	○	○	–

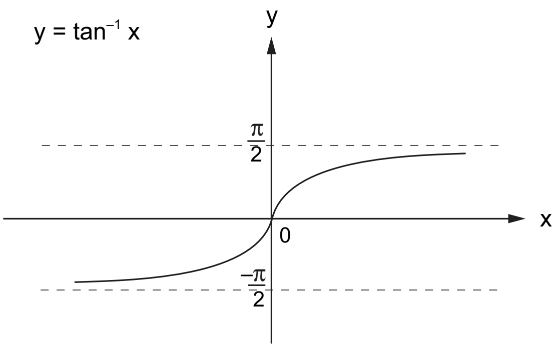
Function

1) Performs arctangent operation on “S” (real type (single-precision floating point type) [REAL]) and outputs the result to “D.”
2) The unit of output is radian.
3) The input range is from the maximum minus value to the maximum plus value and the output range is $-\pi/2$ to $\pi/2$.
4) The number of significant digits of output is 5.

<Operation>

S (REAL)	Operation	D (REAL)
7.854E – 01	⇨	6.6577E – 01

ENO = 1



(18) Move (MOVE)

Symbol

MOVE

MOVE S → D

S : Move source device or data to be moved
D : Move destination device

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Function

MOVE Moves input data “S” directly to “D.”

<Operation>

S (DINT)		D (DINT)
32767	⇨	32767
		ENO = 1

Note: Data in input device “S” and output device “D” must have the same bit length.

(19) Negation (NEG)

Symbol

NEG

NEG S → D

NEG_R

NEG_R S → D

NEG : Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
NEG_R : Real type (single-precision floating point type) [REAL]

S : Input data or the device where input data is stored
D : Device storing the result

Available devices

	Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Note: When the data type is real, only double-word devices are available.

Function

1) Inverts the sign of input data “S” and outputs the result to “D.”
2) The available data types are integer type, double integer type and real type.
3) When the data type is integer type or double integer type and the operation result exceeds the range of the data type, ENO goes to 0.

<Operation>

S (DINT)	Output	D (DINT)
123	⇨	–123
		ENO = 1

- For a minus boundary value of DINT or INT type

S (DINT)	Output	D (DINT)
–2147483648	⇨	2147483647
		ENO = 0

S (INT)	Output	D (INT)
–32768	⇨	32767
		ENO = 0

(20) Block move (BMOV)

Symbol

BMOV

BMOV S, D, N

S: Heading device of data to be moved from
D: Heading device of data to be moved to
N: Number of words to be moved or the device where the number of words is stored (range: 1 to 32767)

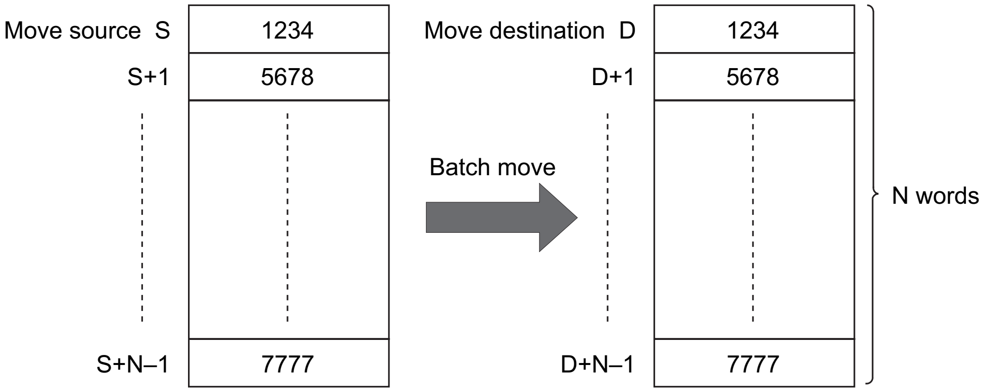
Available devices

	Bit devices										Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	○	–	–	–	○	○	○	○	○	○	○	–	○	○	○	○	○	○	○	–	–	○	○	–
D	–	○	○	○	–	–	–	–	○	○	–	○	○	○	–	–	○	○	–	○	○	○	–	–	–	○	○	–
N	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	○	○	○	–	–	○	○	–	–	–	○	○	○

Function

BMOV

1) Moves data of “N” words (number of words to be moved) at a time from the device specified in “S” to the area of N words beginning at the device specified in “D.”



2) Data in “S” (move source data) and “D” (move destination data) must have the same bit width.

Note: Specify the number of words to be moved so that devices used for other purposes are not rewritten. Especially take care when indirectly specifying the number of words to be moved.

(21) Fill move (FMOV)

Symbol

FMOV

FMOV S, D, N

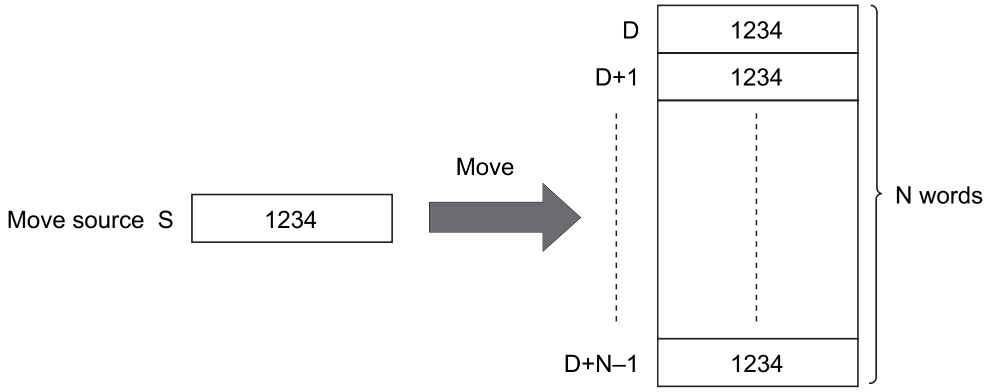
S: Data to be moved or the device where the data is stored
D: Heading device of move destination
N: Number of words to be moved or the device where the number of words is stored (range: 1 to 32767)

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
N	○	○	○	○	○	○	○	○	–	–	○	○	–	–	–	○	○	○

Function

FMOV Moves the contents of the device specified in “D” to the area of N words beginning at the device specified in “D.”



Note: Specify the number of words to be moved so that devices used for other purposes are not rewritten. Especially take care when indirectly specifying the number of words to be moved.

(22) Exchange (XCH)

Symbol

XCH

XCH D1, D2

D1, D2: Devices where data to be exchanged is stored

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
D1	○	○	○	○	–	○	○	○	○	○	○	○	–	○	○	○	○	○
D2	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Function

XCH Exchanges the data in “D1” with that in “D2.”

Before exchange D1

0	0	1	1	0	0	1	1	1	0	1	0	0	1	1	1

D2

0	0	1	1	0	0	1	1	0	0	1	1	0	0	0	0

After exchange D1

0	0	1	1	0	0	1	1	0	0	1	1	0	0	0	0

D2

0	0	1	1	0	0	1	1	1	0	1	0	0	1	1	1

Data in “D1” and “D2” must have the same bit width.

(23) Indirect put (BDMPX)

Symbol

BDMPX

```
──[ BDMPX    S1, D, S2, N ]─┤
```

S1 : Move source device
D : Move destination device (reference address)
S2 : Data pointer
N : Number of words to be moved (range: 1 to 32767)

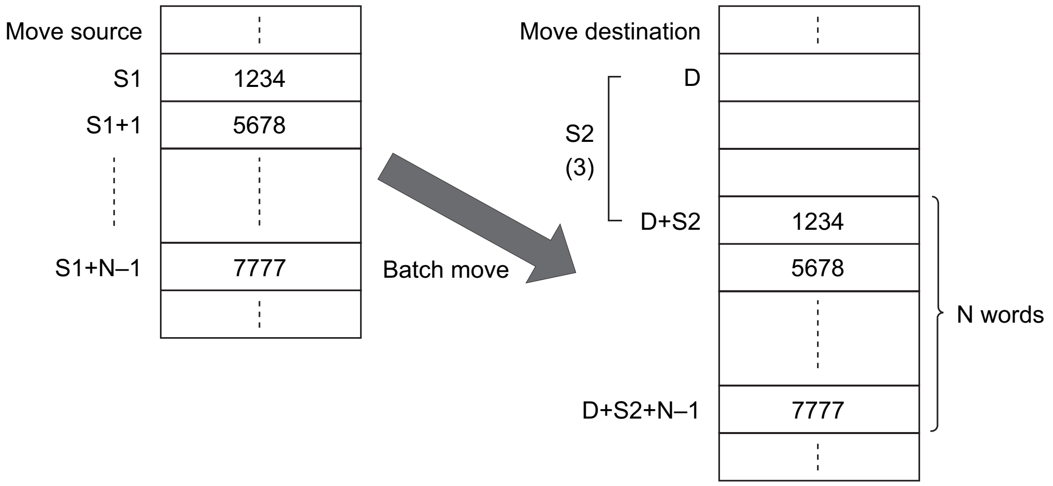
Available devices

	Bit devices										Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	–	–	–	○	○	○	○	○	○	○	–	○	○	○	○	○	○	○	–	–	○	○	–
D	–	○	○	○	–	–	–	–	○	○	–	○	○	○	–	–	○	○	–	○	○	○	–	–	–	○	○	–
S2	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	○	○	○	○	–	○	○	–	–	–	○	○	○
N	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	○	○	○	–	–	○	○	–	–	–	○	○	○

Function

BDMPX

1) Moves data of “N” words (number of words to be moved) from the device specified in “S1” to the device displaced by the data (data pointer) in the device specified in “S2” from the device (reference address) specified in “D.”



2) Data in “S1” (move source data) and “D” (move destination data) must have the same bit width.

Note: Specify the number of words to be moved so that devices used for other purposes are not rewritten. Especially take care when indirectly specifying the number of words to be moved.

(24) Indirect get (BMPX)

Symbol

BMPX

BMPX S1, S2, D, N

S1 : Move source device (reference address)
S2 : Data pointer
D : Move destination device
N : Number of words to be moved (range: 1 to 32767)

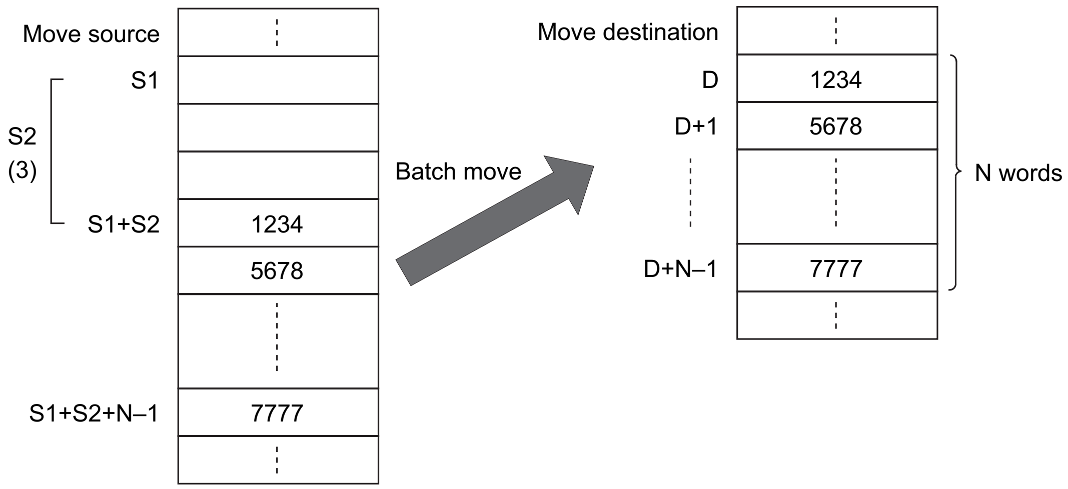
Available devices

	Bit devices										Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	–	–	–	○	○	○	○	○	○	○	–	○	○	○	○	○	○	○	–	–	○	○	–
S2	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	○	○	○	–	–	○	○	–	–	–	○	○	○
D	–	○	○	○	–	–	–	–	–	–	–	○	○	○	–	–	○	○	–	○	○	○	–	–	–	○	○	–
N	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	○	○	○	–	–	○	○	–	–	–	○	○	○

Function

BDMPX

1) Moves data of "N" words from the device displaced by the data (data pointer) in the device specified in "S2" using "S1" as the reference address to the area starting from "D".



2) Data in "S" (move source data) and "D" (move destination data) must have the same bit width.

Note: Specify the number of words to be moved so that devices used for other purposes are not rewritten. Especially take care when indirectly specifying the number of words to be moved.

3-2-5 Bit string operation instructions

(1) Logical AND (AND_AW)

Symbol

```
AND_AW
   ──[ AND_AW     S1, S2 → D ]──┤
```

S1 : Data to be logical ANDed or device where the data is stored
S2 : Data to be logical ANDed or device where the data is stored
D : Device storing the result

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Function

1) Performs logical AND bit by bit on the data of the device specified in “S1” and the data (or constant) of the device specified in “S2” and outputs the result to the device specified in “D.”

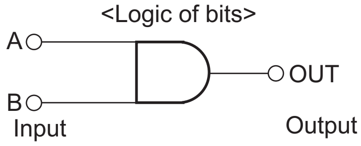
Example

S1	1	1	1	1	0	0	0	0	1	1	0	0	1	1	0	0
S2	1	1	0	0	1	1	0	0	1	1	0	0	0	1	1	1

⬇

D	1	1	0	0	0	0	0	0	1	1	0	0	0	1	0	0

<Logic of bits>



A	B	OUT
0	0	0
1	0	0
0	1	0
1	1	1

2) Data in “S1” and “S2” must have the same bit width.

(2) Logical OR (OR_AW)

Symbol

OR_AW

OR_AW S1, S2 → D

S1 : Data to be logical ORed or device where the data is stored
S2 : Data to be logical ORed or device where the data is stored
D : Device storing the result

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Function

1) Performs logical OR bit by bit on the data of the device specified in “S1” and the data (or constant) of the device specified in “S2” and outputs the result to the device specified in “D.”

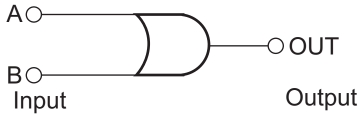
Example

S1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1
S2	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0

⬇

D	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

<Logic of bits>



A	B	OUT
0	0	0
1	0	1
0	1	1
1	1	1

2) Data in “S1” and “S2” must have the same bit width.

(3) Logical exclusive OR (XOR)

Symbol

Bit type:

XOR_B

XOR_B S1, S2 → D

Word/double-word type:

XOR_AW

XOR_AW S1, S2 → D

S1 : Data to be logical exclusive ORed or device where the data is stored
S2 : Data to be logical exclusive ORed or device where the data is stored
D : Device storing the result

Available devices

	Bit devices										Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	○	–	–	–	○	○	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

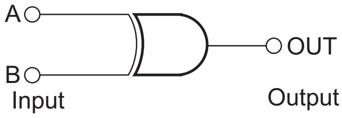
Function

1) Performs logical exclusive OR bit by bit on the data of the device specified in “S1” and the data (or constant) of the device specified in “S2” and outputs the result to the device specified in “D.”

Example

S1	1	0	1	1	0	1	0	0	0	0	1	1	0	1	1	0
S2	1	0	0	0	1	0	1	0	0	1	0	1	1	1	1	0
D	0	0	1	1	1	1	1	0	0	1	1	0	1	0	0	0

<Logic of bits>



A	B	OUT
0	0	0
1	0	1
0	1	1
1	1	0

2) Data in “S1” and “S2” must have the same bit width.

(4) Logical exclusive NOR (XORN)

Symbol

Bit type:

XORN_B

XORN_B S1, S2 → D

Word/double-word type:

XORN_AW

XORN_AW S1, S2 → D

S1 : Data to be logical exclusive ORed or device where the data is stored
S2 : Data to be logical exclusive ORed or device where the data is stored
D : Device storing the result

Available devices

	Bit devices										Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	○	–	–	–	○	○	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

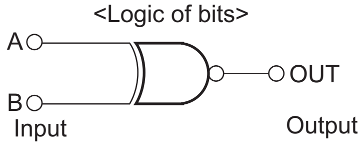
Function

1) Performs logical exclusive NOR bit by bit on the data of the device specified in “S1” and the data (or constant) of the device specified in “S2” and outputs the result to the device specified in “D.”

Example

S1	1	0	1	1	0	0	1	0	0	0	0	1	1	0	1	0
S2	1	0	0	0	1	1	1	0	1	1	0	0	1	1	1	0
D	1	1	0	0	0	0	1	1	0	0	1	0	1	0	1	1

<Logic of bits>



A	B	OUT
0	0	1
1	0	0
0	1	0
1	1	1

2) Data in “S1” and “S2” must have the same bit width.

(5) Logical NOT (NOT)

Symbol

NOT_AW

NOT_AW S → D

S : Move source device or data
D : Move destination device

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	—	○	○	○	—	○	○	○	—	○	○	○	—	○	○	○	○	—

Function

1) Inverts each bit of data in the device specified in “S” and outputs the result to the device specified in “D.”

Move source: S

0	0	1	1	0	1	0	0	1	1	1	0	0	0	1	1

⬇ Invert and move

Move destination: D

1	1	0	0	1	0	1	1	0	0	0	1	1	1	0	0

2) Data in “S” and “D” must have the same bit width.

(6) Rotation (ROR, ROL)

Symbol

Rotation right:

ROR

ROR S, N → D

Rotation left:

ROL

ROL S, N → D

S: Device to be rotated
N: Number of bits to be rotated (N = 0 to 31)
D: Device storing rotated data

Available devices

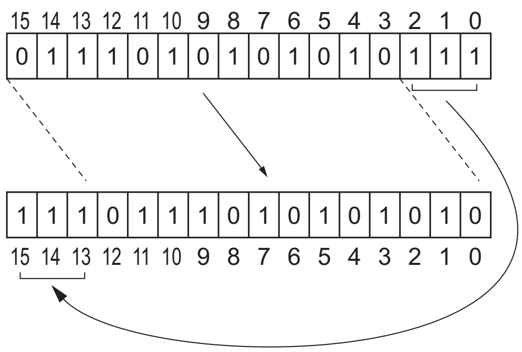
	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
N	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	○
D	–	○	○	○	–	–	○	○	–	○	○	○	–	–	–	○	○	–

Function

ROR

1) Rotates input data “S” (device to be rotated) rightward by the numeric value specified in “N” (number of bits to be rotated) and outputs the result to “D.”

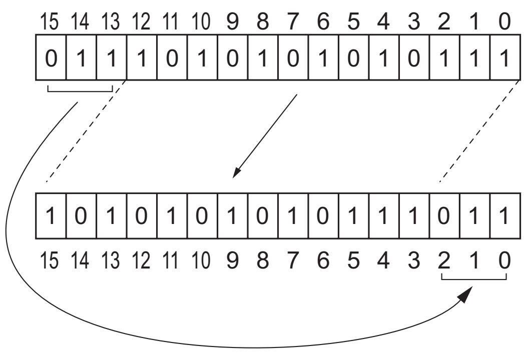
Example: Rotation right 3 bits (N = 3)



ROL

1) Rotates input data “S” (device to be rotated) leftward by the numeric value specified in “N” (number of bits to be rotated) and outputs the result to “D.”

Example: Rotation left 3 bits (N = 3)



<Description about number of bits to be rotated “N”>

When the bit width is 16 bits, the low-order four bits are valid for “N.”
For example, when N = 16, the rotation size is 0 bit; when N = 17, the rotation size is 1 bit.
When the bit width is 32 bits, the low-order five bits are valid for “N.”
For example, when N = 32, the rotation size is 0 bit; when N = 33, the rotation size is 1 bit.

* Data in “S” and “D” must have the same bit width.

(7) Shift (SHR, SHL)

Symbol

SHR — [SHR S, N → D]

SHL — [SHL S, N → D]

S: Device to be shifted
N: Number of bits to be shifted (N = 0 to 31)
D: Device storing shifted data

Available devices

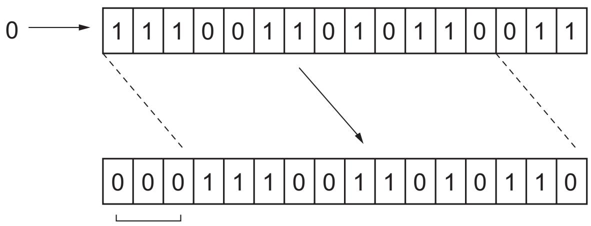
	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
N	○	○	○	○	○	○	○	○	—	—	—	—	—	—	—	—	—	○
D	—	○	○	○	—	—	○	○	—	○	○	○	—	—	—	○	○	—

Function

SHR

1) Shifts input data “S” rightward by “N” bits, puts 0s in the bits emptied by shifting and outputs the result to the device specified in “D.”

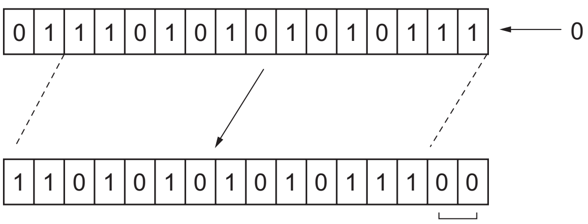
Example: Shift right 3 bits (N = 3)



SHL

1) Shifts input data “S” leftward by “N” bits, puts 0s in the bits emptied by shifting and outputs the result to the device specified in “D.”

Example: Rotation left 3 bits (N = 3)



<Description about number of bits to be rotated "N">

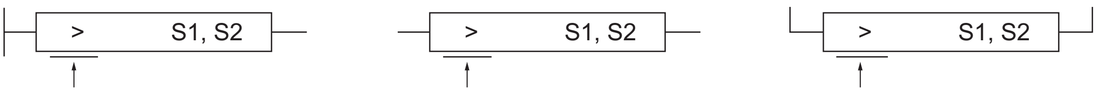
When the bit width is 16 bits, the low-order four bits are valid for “N.”
For example, when N = 16, the rotation size is 0 bit; when N = 17, the rotation size is 1 bit.
When the bit width is 32 bits, the low-order five bits are valid for “N.”
For example, when N = 32, the rotation size is 0 bit; when N = 33, the rotation size is 1 bit.

* Data in “S” and “D” must have the same bit width.

3-2-6 Selection and comparison instructions

(1) Comparison (>)

Symbol



Operation start (LD)	Connection in series (AND)	Connection in parallel (OR)
>	>	>
UAI >	UAI >	UAI >
R >	R >	R >
B >	B >	B >
AW >	AW >	AW >
D >	D >	D >
TOD >	TOD >	TOD >
T >	T >	T >
DT >	DT >	DT >

S1, S2: Comparison data or device

: Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
UAI : Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
R : Real type (binary floating point type) [REAL]
B : Bit type [BOOL]
AW : Word type [WORD]/double-word type [DWORD]
D : Date type [DATE]
TOD : Duration type [TOD]
T : Continuous duration type [TIME]
DT : Date + duration type [DT]

Available devices

	Bit devices (Note)										Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Note: When the data type is real, only double-word devices are available.

Function

1) When the condition of input data “S1” > input data “S2” is met, the operation (comparison) result goes ON (conductive).
2) Input data “S1” and “S2” must have the same data type and same bit width.

<Operation>

S1 (INT)		S2 (INT)		Output
5678	>	1234	⇨	ON

S1 (INT)		S2 (INT)		Output
1234	>	5678	⇨	OFF

Note: Generally, real-type data contains an error. As operations are performed repeatedly, the error increases. Thus, comparison of real-type data may produce operation result which is different from the actual value. When making comparisons, consider errors.

(2) Comparison (>=)

Symbol

>= S1, S2	>= S1, S2	>= S1, S2
Operation start (LD)	Connection in series (AND)	Connection in parallel (OR)
> =	> =	> =
UAI > =	UAI > =	UAI > =
R > =	R > =	R > =
B > =	B > =	B > =
AW > =	AW > =	AW > =
D > =	D > =	D > =
TOD > =	TOD > =	TOD > =
T > =	T > =	T > =
DT > =	DT > =	DT > =

S1, S2: Comparison data or device

	: Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
UAI	: Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
R	: Real type (binary floating point type) [REAL]
B	: Bit type [BOOL]
AW	: Word type [WORD]/double-word type [DWORD]
D	: Date type [DATE]
TOD	: Duration type [TOD]
T	: Continuous duration type [TIME]
DT	: Date + duration type [DT]

Available devices

	Bit devices (Note)										**Word devices (W*)** (Note)								**Double-word devices (D*)**									**Constants**
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Note: When the data type is real, only double-word devices are available.

Function

1) When the condition of input data “S1” >= input data “S2” is met, the operation (comparison) result goes ON (conductive).
2) Input data “S1” and “S2” must have the same data type and same bit width.

<Operation>

S1 (INT)		S2 (INT)		Output
1234	>=	1234	⇨	ON

S1 (INT)		S2 (INT)		Output
1234	>=	3456	⇨	OFF

Note: Generally, real-type data contains an error. As operations are performed repeatedly, the error increases. Thus, comparison of real-type data may produce operation result which is different from the actual value. When making comparisons, consider errors.

(3) Comparison (=)

Symbol

= S1, S2		= S1, S2		= S1, S2

Operation start (LD)
=
UAI =
R =
B =
AW =
D =
TOD =
T =
DT =

Connection in series (AND)
=
UAI =
R =
B =
AW =
D =
TOD =
T =
DT =

Connection in parallel (OR)
=
UAI =
R =
B =
AW =
D =
TOD =
T =
DT =

S1, S2: Comparison data or device

	: Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
UAI	: Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
R	: Real type (binary floating point type) [REAL]
B	: Bit type [BOOL]
AW	: Word type [WORD]/double-word type [DWORD]
D	: Date type [DATE]
TOD	: Duration type [TOD]
T	: Continuous duration type [TIME]
DT	: Date + duration type [DT]

Available devices

	Bit devices (Note)										Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Note: When the data type is real, only double-word devices are available.

Function

1) When the condition of input data “S1” = input data “S2” is met, the operation (comparison) result goes ON (conductive).
2) Input data “S1” and “S2” must have the same data type and same bit width.

<Operation>

S1 (INT)		S2 (INT)		Output
1234	=	1234	⇨	ON

S1 (INT)		S2 (INT)		Output
1234	=	2345	⇨	OFF

Note: Generally, real-type data contains an error. As operations are performed repeatedly, the error increases. Thus, comparison of real-type data may produce operation result which is different from the actual value. When making comparisons, consider errors.

(4) Comparison (<=)

Symbol

< = S1, S2	< = S1, S2	< = S1, S2
Operation start (LD)	Connection in series (AND)	Connection in parallel (OR)
< =	< =	< =
UAI < =	UAI < =	UAI < =
R < =	R < =	R < =
B < =	B < =	B < =
AW < =	AW < =	AW < =
D < =	D < =	D < =
TOD < =	TOD < =	TOD < =
T < =	T < =	T < =
DT < =	DT < =	DT < =

S1, S2: Comparison data or device

	: Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
UAI	: Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
R	: Real type (binary floating point type) [REAL]
B	: Bit type [BOOL]
AW	: Word type [WORD]/double-word type [DWORD]
D	: Date type [DATE]
TOD	: Duration type [TOD]
T	: Continuous duration type [TIME]
DT	: Date + duration type [DT]

Available devices

	Bit devices (Note)										Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Note: When the data type is real, only double-word devices are available.

Function

1) When the condition of input data “S1” <= input data “S2” is met, the operation (comparison) result goes ON (conductive).
2) Input data “S1” and “S2” must have the same data type and same bit width.

<Operation>

S1 (INT)		S2 (INT)	Output	
1234	<=	5678	⇨	ON

S1 (INT)		S2 (INT)	Output	
1234	<=	1234	⇨	ON

S1 (INT)		S2 (INT)	Output	
5678	<=	1234	⇨	OFF

Note: Generally, real-type data contains an error. As operations are performed repeatedly, the error increases. Thus, comparison of real-type data may produce operation result which is different from the actual value. When making comparisons, consider errors.

(5) Comparison (<)

Symbol

< S1, S2	< S1, S2	< S1, S2
Operation start (LD)	Connection in series (AND)	Connection in parallel (OR)
<	<	<
UAI <	UAI <	UAI <
R <	R <	R <
B <	B <	B <
AW <	AW <	AW <
D <	D <	D <
TOD <	TOD <	TOD <
T <	T <	T <
DT <	DT <	DT <

S1, S2: Comparison data or device

	: Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
UAI	: Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
R	: Real type (binary floating point type) [REAL]
B	: Bit type [BOOL]
AW	: Word type [WORD]/double-word type [DWORD]
D	: Date type [DATE]
TOD	: Duration type [TOD]
T	: Continuous duration type [TIME]
DT	: Date + duration type [DT]

Available devices

	Bit devices (Note)										Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Note: When the data type is real, only double-word devices are available.

Function

1) When the condition of input data "S1" < input data "S2" is met, the operation (comparison) result goes ON (conductive).
2) Input data "S1" and "S2" must have the same data type and same bit width.

<Operation>

S1 (INT)		S2 (INT)		Output
1234	<	5678	⇨	ON

S1 (INT)		S2 (INT)		Output
5678	<	1234	⇨	OFF

Note: Generally, real-type data contains an error. As operations are performed repeatedly, the error increases. Thus, comparison of real-type data may produce operation result which is different from the actual value. When making comparisons, consider errors.

(6) Comparison (≠)

Symbol

< > S1, S2	< > S1, S2	< > S1, S2
Operation start (LD)	Connection in series (AND)	Connection in parallel (OR)
< >	< >	< >
UAI < >	UAI < >	UAI < >
R < >	R < >	R < >
B < >	B < >	B < >
AW < >	AW < >	AW < >
D < >	D < >	D < >
TOD < >	TOD < >	TOD < >
T < >	T < >	T < >
DT < >	DT < >	DT < >

S1, S2: Comparison data or device

	: Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
UAI	: Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
R	: Real type (binary floating point type) [REAL]
B	: Bit type [BOOL]
AW	: Word type [WORD]/double-word type [DWORD]
D	: Date type [DATE]
TOD	: Duration type [TOD]
T	: Continuous duration type [TIME]
DT	: Date + duration type [DT]

Available devices

	Bit devices (Note)										Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Note: When the data type is real, only double-word devices are available.

Function

1) When the condition of input data “S1” ≠ input data “S2” is met, the operation (comparison) result goes ON (conductive).
2) Input data “S1” and “S2” must have the same data type and same bit width.

<Operation>

S1 (INT)		S2 (INT)		Output
1234	≠	5678	⇨	ON
5678	≠	1234	⇨	ON
1234	≠	1234	⇨	OFF

Note: Generally, real-type data contains an error. As operations are performed repeatedly, the error increases. Thus, comparison of real-type data may produce operation result which is different from the actual value. When making comparisons, consider errors.

(7) Select (SEL)

Symbol

```
──┬─────────────────────────┬──
  │ SEL      S1, S2, S3 → D │
  └─────────────────────────┘
    ───
     ↑
```

SEL	: Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
SEL_UAI	: Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
SEL_R	: Real type (binary floating point type) [REAL]
SEL_B	: Bit type [BOOL]
SEL_AW	: Word type [WORD]/double-word type [DWORD]
SEL_T	: Continuous duration type [TIME]
SEL_STR	: Character string type [STRING]

S1	: Data store destination selection device (bit)
S2, S3	: Move source devices
D	: Move (store) destination device

Available devices

	Bit devices (Note)										Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S3	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	○	–	–	–	○	○	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Note: When the data type is real, only double-word devices are available.

Function

1) When the value of input data “S1” is OFF, outputs the value of “S2” to “D.” When the value of “S1” is ON, outputs the value of “S3” to “D.”
2) Data in “S2,” “S3” and “D” must have the same bit width.

<Operation>

- When S1 is OFF

```
S2
[     12345]   Output    D
                ⇨       [     12345]
S3
[      5000]
```

- When S1 is ON

```
S2
[     12345]   Output    D
                ⇨       [      5000]
S3
[      5000]
```

[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)

(8) Maximum value (MAX)

Symbol

MAX S1, S2 → D

MAX : Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
MAX_UAI : Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
MAX_R : Real type (binary floating point type) [REAL]

S1, S2 : Comparison data or device
D : Storing device

Available devices

	Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Note: When the data type is real, only double-word devices are available.

Function

1) Compares input data “S1” and input data “S2” and outputs the maximum value to “D.”
2) Data in “S1,” “S2” and “D” must have the same bit width.

<Operation>

S1 (INT): 1234
S2 (INT): 5678

Output ⇨ D (INT): 5678

(9) Minimum value (MIN)

Symbol

MIN S1, S2 → D

MIN : Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
MIN_UAI : Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
MIN_R : Real type (binary floating point type) [REAL]

S1, S2 : Comparison data or device
D : Storing device

Available devices

	Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Note: When the data type is real, only double-word devices are available.

Function

1) Compares input data “S1” and input data “S2” and outputs the maximum value to “D.”
2) Data in “S1,” “S2” and “D” must have the same bit width.

<Operation>

S1 (INT)
1234

S2 (INT)
5678

Output ⇨

D (INT)
1234

(10) Limit (LIMIT)

Symbol

LIMIT S1, S2, S3 → D

LIMIT : Integer type (signed) [INT]/double-precision integer type (signed) [DINT]
LIMIT_UAI : Integer type (unsigned) [UINT]/double-precision integer type (unsigned) [UDINT]
LIMIT_R : Real type (binary floating point type) [REAL]

S1, S2, S3 : Comparison data or device
D : Storing device

Available devices

	Word devices (W*) (Note)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
S3	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Note: When the data type is real, only double-word devices are available.

Function

1) When "S2" is less than "S1," outputs "S1."
When "S2" is greater than "S3," outputs "S3." In other cases, outputs "S2."
2) When "S1" is greater than "S3," outputs "S3."
3) Input data "S1," "S2," "S3" and output data "D" must have the same type and the same bit width.

<Operation>

- When "S2" <= "S1"

S1	1000	Output	D
S2	0	⇨	1000
S3	5000		

- When "S1" < "S2" < "S3"

S1	1000	Output	D
S2	3000	⇨	3000
S3	5000		

- When "S2" >= "S3"

S1	1000	Output	D
S2	5500	⇨	5000
S3	5000		

3-2-7 Character string instructions

(1) Get length (LEN)

Symbol

LEN	S → D

S : Device where string type data is stored
D : Device storing character string data

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	—	—	○	○	—	—	—	—	—	—	—	—	—	—
D	—	○	○	○	—	○	○	○	—	—	—	—	—	—	—	—	—	—

Function

Counts the number of characters in the string data stored beginning at the address position of the specified device “S” and stores the result in the device specified in “S.”

<Operation>

S	'A'
S+1	'B'
S+2	'C'
	'D'
	'E'
	'F'
	'G'
	'H'
	'I'
S+9	'J'
S+10	H0000

⇨ D (INT) 10

[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character (1- or 2-byte character) is handled as 16 bits.

(2) Get left sub-string (LEFT)

Symbol

LEFT S, N → D

S : Device where string type data is stored
N : Number of characters to be extracted
D : Device storing character string data extracted

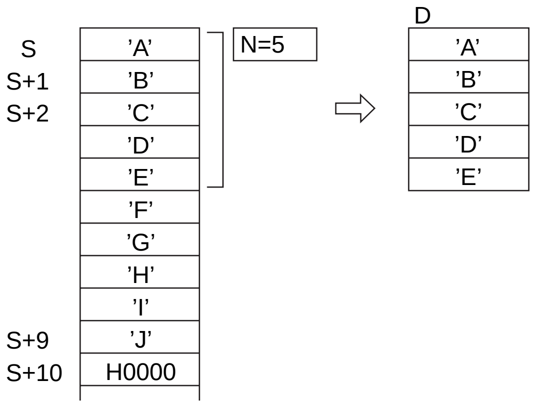
Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	—	—	○	○	—	—	—	—	—	—	—	—	—	—
N	○	○	○	○	—	○	○	○	—	—	—	—	—	—	—	—	—	○
D	—	○	○	○	—	—	○	○	—	—	—	—	—	—	—	—	—	—

Function

1) Extracts characters of the number specified in “N” from the left end of the string data stored beginning at the address of specified device “S.”
2) When “N” equals or greater than the number of characters in the input character string, the input character string is directly output. ENO goes to 1.
3) When N = 0, outputs only NULL and ENO goes to 1.

<Operation>



[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(3) Get right sub-string (RIGHT)

Symbol

RIGHT S, N → D

S : Device where string type data is stored
N : Number of characters to be extracted
D : Device storing character string data extracted

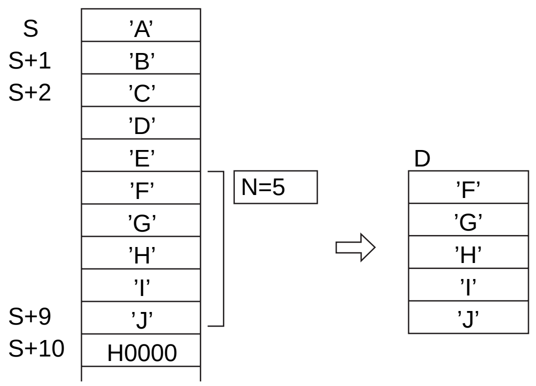
Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
N	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	○
D	–	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–

Function

1) Extracts characters of the number specified in “N” from the right end of the string data stored beginning at the address of specified device “S.”
2) When “N” equals or greater than the number of characters in the input character string, the input character string is directly output. ENO goes to 1.
3) When N = 0, outputs only NULL and ENO goes to 1

<Operation>.



[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(4) Get middle sub-string (MID)

Symbol

MID S, N1, N2 → D

S : Device where string type data is stored
N1 : Number of characters to be extracted
N2 : Heading position of string data to be extracted
D : Device storing character string data extracted

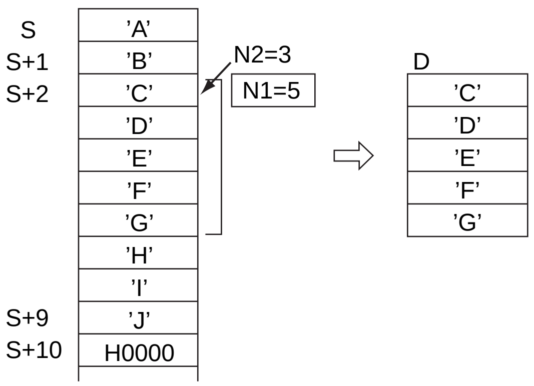
Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	—	—	○	○	—	—	—	—	—	—	—	—	—	—
N1	○	○	○	○	—	○	○	○	—	—	—	—	—	—	—	—	—	○
N2	○	○	○	○	—	○	○	○	—	—	—	—	—	—	—	—	—	○
D	—	○	○	○	—	—	○	○	—	—	—	—	—	—	—	—	—	—

Function

1) Extracts characters of the number specified in “N1” from the “N2”th character of the string type data stored beginning at the address of specified device “S.”
2) For SPH, when N2 >= 65 or N2 = 0, the output goes to NULL and EN0 to 0. When the number of input characters is < N2 and N1≠0, the output goes to NULL and ENO to 0. When starting position “N2” is less than the number of characters in the character string and the sum of starting position “N2” and number of characters “N1” is greater than the number of characters of the input character string, outputs the character string ranging from the starting position to the end of the input character string. ENO goes to 1.
3) When N1 = 0, outputs only NULL and ENO goes to 1.

<Operation>



[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(5) Concatenate (CONCAT)

Symbol

CONCAT S1, S2 → D

S1 : Device where string type data is stored
S2 : Device where string type data is stored
D : Device storing the string data concatenated

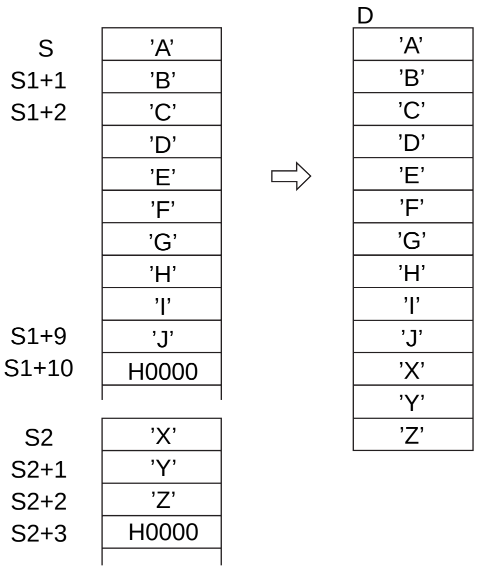
Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
S2	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
D	–	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–

Function

1) Concatenates the string data stored beginning at the address of specified device “S2” to the string type data stored beginning at the address of specified device “S1.”
2) For SPH, when the number of characters to be concatenated is greater than 64, outputs the first 64 characters and ENO goes to 0.
3) When the number of concatenated characters is “0,” outputs only NULL and ENO goes to 1.

<Operation>



[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(6) Insert string (INSERT)

Symbol

INSERT S1, S2, N → D

S1 : Device where string type data of insert destination is stored
S2 : String type data to be inserted
N : Insert destination position
D : Device storing insertion result

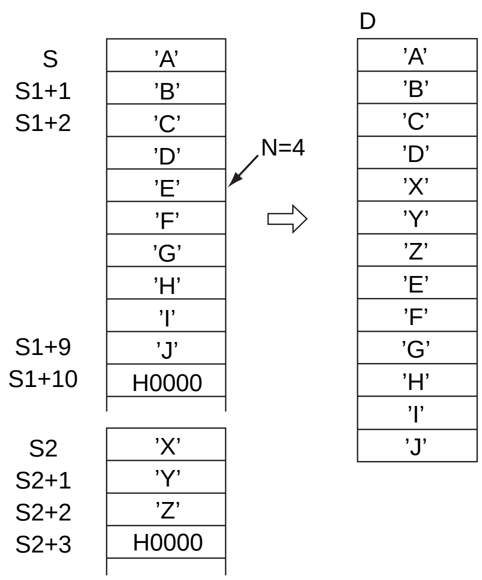
Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
S2	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
N	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	○
D	–	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–

Function

1) Inserts string type data stored beginning at the address specified in “S2” to the positions beginning at the “N”th character of the string data stored beginning at the address specified in “S1” and outputs the result to “D.”
2) When the number of concatenated characters is greater than 64, outputs only the first 64 characters of those inserted and ENO goes to 0.
3) When the number of characters in “S1” is < N, outputs the input characters of “S1” and ENO goes to 0.
4) When N = 0, inserts the character string data of “S2” to the position preceding the character string data of “S1” and ENO goes to 1.

<Operation>



[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(7) Delete string (DELETE)

Symbol

DELETE S, N1, N2 → D

S : Device where string type data is stored
N1 : Number of pieces of string type data to be deleted
N2 : Heading position of string type data to be deleted
D : Device storing deletion result

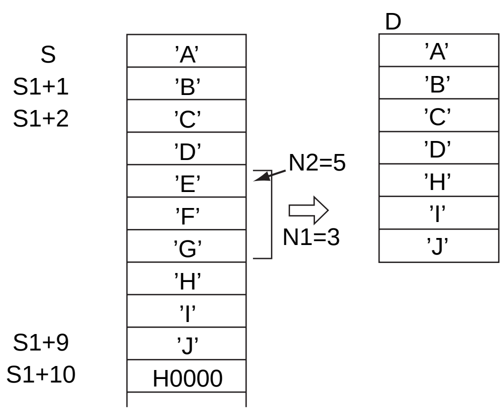
Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	—	—	○	○	—	—	—	—	—	—	—	—	—	—
N1	○	○	○	○	—	○	○	○	—	—	—	—	—	—	—	—	—	○
N2	○	○	○	○	—	○	○	○	—	—	—	—	—	—	—	—	—	○
D	—	○	○	○	—	—	○	○	—	—	—	—	—	—	—	—	—	—

Function

1) Deletes characters of the number specified in “N1” from the “N2”th position of the string data stored beginning at the address specified in “S” and outputs the result to “D.”
2) When N2 > 65 or N2 = 0, outputs NULL and ENO goes to 0.
3) When the number of characters in “S” is < N2, outputs the string data of “S” and ENO goes to 0.
4) When the result contains 65 or more characters, outputs only the first 64 characters and ENO goes to 0.
5) When N1 = 0, outputs directly the string data of “S” and ENO goes to 1.

<Operation>



[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(8) Replace string (REPLACE)

Symbol

REPLACE S1, S2, N1, N2 → D

S1 : Device where string type data is stored (replace source)
S2 : Device where string type data is stored (replace destination)
N1 : Number of pieces of string type data to be replaced
N2 : Heading position of string type data to be replaced

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
S2	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
N1	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	○
N2	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	○
D	–	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–

Function

1) Deletes characters of the number specified in “N1” from the “N2”th position of the string data stored beginning at the address specified in “S1,” inserts string data stored beginning at the address of “S2” and outputs the result to “D.”
2) When the number of characters replaced is greater than 64, stores only the first 64 characters of those replaced and ENO goes to 0.
3) When N2 >= 65 or N2 = 0, outputs NULL and ENO goes to 0.
4) When the number of characters in “S1” is < N2, outputs the string data of “S1” and ENO goes to 0.
5) When N1 = 0, inserts characters of “S2” from the “N2”th position and ENO goes to 1.

<Operation>

N2=3
N1=5 ⇨

S	’A’
S1+1	’B’
S1+2	’C’
	’D’
	’E’
	’F’
	’G’
	’H’
	’I’
S1+9	’J’
S1+10	H0000

S2	’X’
S2+1	’Y’
S2+2	’Z’
S2+3	H0000

D
’A’
’B’
’X’
’Y’
’Z’
’H’
’I’
’J’

[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(9) Find string (FIND)

Symbol

REPLACE S1, S2 → D

S1 : Device where string type data is stored (replace source)
S2 : Device where string type data is stored (replace destination)
D : Heading position of searched result

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
S2	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
D	–	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	–

Function

1) Searches the character string stored beginning at the address of “S1” for the character string same as that stored beginning at the address of “S2” and outputs the first position found to “D.”
2) When the character string searched for is not found, outputs “0.”

<Operation>

S1	'A'
S1+1	'B'
S1+2	'C'
	'D'
	'E'
	'F'
	'G'
	'H'
	'I'
S1+9	'J'
S1+10	H0000

⇨ D (INT): 3

S2	'C'
S2+1	'D'
S2+2	'E'
S2+3	'F'
S2+4	H0000

[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(10) Compare string (STR>)

Symbol

STR > S1, S2		STR > S1, S2		STR > S1, S2

Instruction symbol: LD_STR > AND_STR > OR_STR >

S1, S2: Comparison data or device

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
S2	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–

Function

1) When the condition of S1 > S2 is met, the output goes ON.
   When the condition above is not met, the output goes OFF.
2) The operation result of a comparison instruction is of BOOL type (ON or OFF).
3) The data type available is string type (STRING).
4) Converts a character string to character code and compares the character code as numeric data.

<Program example>

Execution condition X1.00 — [STR > S1 M0 S2 M10] — FLAG M200 ( )

<Operation>

	1 byte		Character code
S1	お a b	⇨	82A8, 0061, 0062
S2	え	⇨	82A6, [0000], [0000]

S1	'お'	⇨	H82A8
S1+1	'a'		H0061
S1+2	'b'		H0062
S2	'え'	⇨	H82A6
			H0000
			H0000

Therefore,
S1 > S2
Thus, the output to go ON.

Notes: 1) A 1-byte character is added with “00” in its high-order 8 bits and treated as 16-bit code.
2) When character strings have different lengths, the shorter string is added with “0000” to its right end for operation.

[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(11) Compare string (STR >=)

Symbol

STR > = S1, S2 STR > = S1, S2 STR > = S1, S2

Instruction symbol: LD_STR > = AND_STR > = OR_STR > =

S1, S2: Comparison data or device

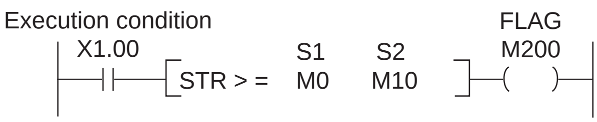
Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
S2	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–

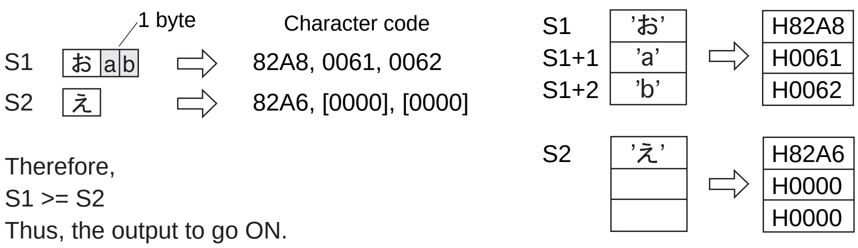
Function

1) When the condition of S1 >= S2 is met, the output goes ON.
   When the condition above is not met, the output goes OFF.
2) The data type available is string type (STRING).
3) Converts a character string to character code and compares the character code as numeric data.

<Program example>



<Operation>



Therefore,
S1 >= S2
Thus, the output to go ON.

[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(12) Compare string (STR =)

Symbol

Instruction symbol: LD_STR = AND_STR = OR_STR =

S1, S2: Comparison data or device

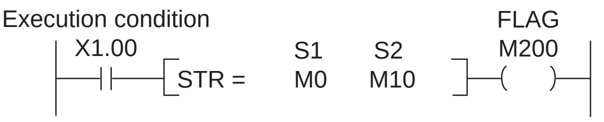
Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
S2	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–

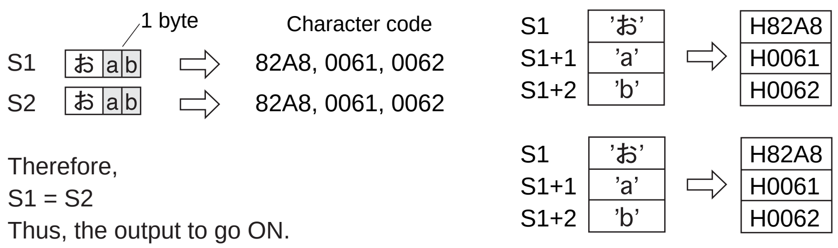
Function

1) When the condition of S1 = S2 is met, the output goes ON.
   When the condition above is not met, the output goes OFF.
2) The data type available is string type (STRING).
3) Converts a character string to character code and compares the character code as numeric data.

<Program example>



<Operation>



Therefore,
S1 = S2
Thus, the output to go ON.

[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(13) Compare string (STR <=)

Symbol

STR < = S1, S2 — STR < = S1, S2 — STR < = S1, S2

Instruction symbol: LD_STR < = AND_STR < = OR_STR < =

S1, S2: Comparison data or device

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	—	—	○	○	—	—	—	—	—	—	—	—	—	—
S2	○	○	○	○	—	—	○	○	—	—	—	—	—	—	—	—	—	—

Function

1) When the condition of S1 <= S2 is met, the output goes ON.
   When the condition above is not met, the output goes OFF.
2) The data type available is string type (STRING).
3) Converts a character string to character code and compares the character code as numeric data.

<Program example>

Execution condition X1.00 — STR < = S1 M0 S2 M10 — FLAG M200 ( )

<Operation>

Character code

S1 あ お ⇨ 82A0, 82A8

S2 あ か ⇨ 82A0, 82A9

Therefore,
S1 <= S2
Thus, the output to go ON.

S1	'あ'	⇨	H82A0
S1+1	'お'		H82A8
S2	'あ'	⇨	H82A0
S2+1	'か'		H82A9

[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(14) Compare string (STR <)

Symbol

Instruction symbol: LD_STR <   AND_STR <   OR_STR <

S1, S2: Comparison data or device

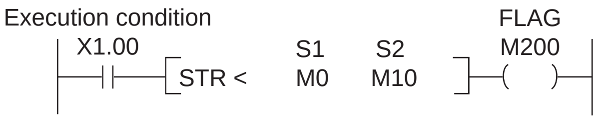
Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	—	—	○	○	—	—	—	—	—	—	—	—	—	—
S2	○	○	○	○	—	—	○	○	—	—	—	—	—	—	—	—	—	—

Function

1) When the condition of S1 < S2 is met, the output goes ON.
   When the condition above is not met, the output goes OFF.
2) The data type available is string type (STRING).
3) Converts a character string to character code and compares the character code as numeric data.

<Program example>



<Operation>

			Character code
S1	あ お	⇨	82A0, H82A8
S2	あ か	⇨	82A0, H82A9

S1	'あ'	⇨	H82A0
S1+1	'お'		H82A8
S2	'あ'	⇨	H82A0
S2+1	'か'		H82A9

Therefore,
S1 < S2
Thus, the output to go ON.

[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(15) Compare string (STR ≠)

Symbol

STR < >   S1, S2 | STR < >   S1, S2 | STR < >   S1, S2

Instruction symbol: LD_STR < > AND_STR < > OR_STR < >

S1, S2: Comparison data or device

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
S2	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–

Function

1) When the condition of S1 ≠ S2 is met, the output goes ON.
   When the condition above is not met, the output goes OFF.
2) The data type available is string type (STRING).
3) Converts a character string to character code and compares the character code as numeric data.

<Program example>

Execution condition
X1.00 ─┤├─ [STR < > S1 M0 S2 M10] ─( )─ FLAG M200

<Operation>

	1 byte		Character code
S1	お a b	⇨	82A8, 0061, 0062
S2	え	⇨	82A6, [0000], [0000]

Therefore,
S1 ≠ S2
Thus, the output to go ON.

S1	'お'	⇨	H82A8
S1+1	'a'		H0061
S1+2	'b'		H0062
S2	'え'	⇨	H82A6
			H0000
			H0000

[Reference]
Number of characters in SPH of character string type: 0 to 64 (1- or 2-byte characters)
For SPH, one character is handled as 16 bits.

(16) Move string (MOVE_STR)

Symbol

MOVE_STR S → D

S : Move source device
D : Move destination device

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–
D	–	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–

Function

Moves the string type data stored beginning at the address of specified device “S” directly to the positions beginning with the address of the device specified in “D.”

<Operation>

	S			D	
S	'あ'	⇨	D	'あ'
S+1	'い'		D+1	'い'
S+2	'う'			'う'
·	'え'			'え'
·	'お'			'お'
·	H0000			H0000

3-2-8 Time instructions

(1) Add time (ADD_)

Symbol

ADD_T_T S1, S2 → D	: Continuous duration type (S1) + continuous duration type (S2)
ADD_TD_T S1, S2 → D	: Duration type (S1) + continuous duration type (S2)
ADD_DT_T S1, S2 → D	: Date and duration type (S1) + continuous duration type (S2)

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
S2	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
D	–	–	–	–	–	–	–	–	–	○	○	○	–	△	△	○	○	–

△ : ADD_T_T Available only for (continuous duration type + continuous duration type) instruction.

Function

ADD_T_T

1) Adds continuous duration type data “S1” and “S2” and stores the result in “D.”
2) When the operation result exceeds the boundary value of the data type, outputs the operation result ignoring the overflow and ENO goes to 0.

ADD_TD_T

1) Adds duration type data “S1” and continuous duration type data “S2” and stores the result as duration type data in “D.”
2) The value in ms unit of the continuous duration type is truncated.
3) When the operation result exceeds the boundary value of the data type, outputs the value converted to 24-hour system and ENO goes to 0.
   (Boundary value: 0, 23:59:59)

ADD_DT_T

1) Adds date and duration type data “S1” and continuous duration type data “S2” and stores the result as date and duration type data in “D.”
2) The value in ms unit of the continuous duration type is truncated.
3) When the operation result exceeds the boundary value of the data type, outputs the operation result ignoring the overflow and ENO goes to 0.

<Operation>

ADD_T_T

- When the operation result is within the range of the data type

S1		S2	Operation	D
2h10m20s123ms	+	1h09m50s456ms	⇨	3h20m10s579ms
				ENO = 1

- When the operation result exceeds the range of the data type

S1		S2	Operation	D
30d2h10m20s123ms	+	25d1h09m50s456ms	⇨	5d10h17m23s283ms
				ENO = 0

[Reference]

Data range of continuous duration type SPH: 0 to 4,294,967,295 (ms)

ADD_TD_T

- When the operation result is within the range of the data type

S1		S2	Operation	D
17:54:30	+	5m55s123ms	⇨	18:00:25
				ENO = 1

- When the operation result exceeds the range of the data type

S1		S2	Operation	TOD
17:54:30	+	10h5m55s123ms	⇨	4:00:25
				ENO = 0

ADD_DT_T

S1		S2	Operation	D
1997-06-25-10:10:00	+	7d20h30m40s567ms	⇨	1997-07-03-06:40:40
				ENO = 1

(2) Subtract time (SUB_)

Symbol

SUB_T_T	S1, S2 → D	: Continuous duration type (S1) – continuous duration type (S2)
SUB_D_D	S1, S2 → D	: Date type (S1) – date type (S2)
SUB_TD_T	S1, S2 → D	: Duration type (S1) – continuous duration type (S2)
SUB_TD_TD	S1, S2 → D	: Duration type (S1) – duration type (S2)
SUB_DT_T	S1, S2 → D	: Date and duration type (S1) – continuous duration type (S2)
SUB_DT_DT	S1, S2 → D	: Date and duration type (S1) – date and duration type (S2)

Function

SUB_T_T

1) Subtracts continuous duration type data “S2” from “S1” and stores the result in “D.”
2) When the operation result exceeds the boundary value of the data type, outputs the operation result ignoring the underflow and ENO goes to 0.
(Boundary values: 0, 49d17h02m47s295ms)

SUB_D_D

1) Subtracts date type data “S2” from “S1” and stores the result as continuous duration type data in “D.”
2) When the operation result exceeds the plus boundary value of the data type, outputs the boundary value of the data type and ENO goes to 0.
3) When the operation result is minus, the output value is not guaranteed and ENO goes to 0.

SUB_TD_T

1) Subtracts continuous duration type data “S2” from duration type data “S1” and stores the result as duration type data in “D.”
2) The value in ms unit of the continuous duration type is truncated.
3) When the operation result exceeds the boundary value of the data type, outputs the value converted to 24-hour system and ENO goes to 0.
(Boundary values: 00:00:00, 23:59:59)

SUB_TD_TD

1) Subtracts duration type data “S2” from “S1” and stores the result as continuous duration type data in “D.”
2) When the operation result is minus, the output value is not guaranteed and ENO goes to 0.

SUB_DT_T

1) Subtracts continuous duration type data “S2” from date and duration type data “S1” and stores the result as date and duration type data in “D.”
2) The value in ms unit of the continuous duration type is truncated.
3) When the operation result is minus, the output value is not guaranteed and ENO goes to 0.

SUB_DT_DT

1) Subtracts date and duration type data “S2” from “S1” and stores the result as continuous duration type data in “D.”
2) When the operation result exceeds the plus boundary value of the data type, outputs the boundary value of the data type and ENO goes to 0.
3) When the operation result is minus, the output value is not guaranteed and ENO goes to 0.

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
S2	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
D	–	–	–	–	–	–	–	–	–	○	○	○	–	△	△	○	○	–

△ : SUB_T_T (Continuous duration type data - continuous duration type data) Available only for SUB_D_D, SUB_TD_TD and SUB_DT_DT instructions.

<Operation>

SUB_T_T

- When the operation result is within the range of the data type

S1		S2	Operation	D
2h10m20s123ms	–	1h09m50s456ms	⇨	1h29s667ms
				ENO = 1

- When the operation result exceeds the range of the data type

S1		S2	Operation	D
1h09m50s456ms	–	2h10m20s123ms	⇨	49d16h02m17s629ms
				ENO = 0

[Reference]
Data range of continuous duration type SPH: 0 to 4,294,967,295 (ms)

SUB_D_D

- When the operation result is within the range of the data type

S1		S2	Operation	D
1997-06-25	–	1997-06-20	⇨	5d
				ENO = 1

- When the operation result exceeds the range of the data type

S1		S2	Operation	D
1997-06-25	–	1997-03-20	⇨	49d17h02m47s295ms
				ENO = 0

SUB_TD_T

- When the operation result is within the range of the data type

S1		S2	Operation	D
19:30:00	–	1h30m123ms	⇨	18:00:00

- When the operation result exceeds the range of the data type

S1		S2	Operation	D
17:54:30	–	20h5m55s123ms	⇨	21:48:35
				EN = 0

SUB_TD_TD

S1		S2	Operation	D
15:30:15	–	08:15:08	⇨	7h15m7s0ms
				ENO = 1

SUB_DT_T

S1		S2	Operation	D
1997-06-25-10:10:00	–	7d20h30m40s567ms	⇨	1997-06-17-13:39:20
				ENO = 1

SUB_DT_DT

S1		S2	Operation	D
1998-04-02-15:30:15	–	1998-04-01-08:15:08	⇨	1d7h15m7s0ms
				ENO = 1

(3) Multiply time (MUL_)

Symbol

MUL_T_UDI   S1, S2 → D : Continuous duration type (S1) × unsigned double-precision integer type (S2)

MUL_T_R   S1, S2 → D : Continuous duration type (S1) × real type (S2)

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
S2	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
D	–	–	–	–	–	–	–	–	–	○	○	○	–	○	○	○	○	–

Function

MUL_T_UDI

1) Multiplies continuous duration type data “S1” by unsigned double-precision integer type data (S2) and stores the result as continuous duration type data in “D.”
2) When the operation result exceeds the boundary value of the data type, outputs the boundary value of the data type and ENO goes to 0.

MUL_T_R

1) Multiplies continuous duration type data “S1” by real type data (S2) and stores the result as continuous duration type data in “D.”
2) When the operation result exceeds the boundary value of the data type, outputs the boundary value of the data type and ENO goes to 0.
3) When the operation result is minus, the output value is not guaranteed and ENO goes to 0.

<Operation>

MUL_T_UDI

- When the operation result is within the range of the data type

S1		S2	Operation	D
5h30m	×	2	⇨	11h
				ENO = 1

- When the operation result exceeds the range of the data type

S1		S2	Operation	D
7d20h30m40s567ms	×	20	⇨	49d17h02m47s295ms
				ENO = 0

[Reference]
Data range of continuous duration type SPH: 0 to 4,294,967,295 (ms)

MUL_T_R

- When the operation result is within the range of the data type

S1		S2	Operation	D
5h30m	×	2.5E + 0	⇨	13h45m
				ENO = 1

- When the operation result exceeds the range of the data type

S1		S2	Operation	D
5h30m	×	2.5E + 3	⇨	49d17h02m47s295ms
				ENO = 0

Note: This instruction converts continuous duration type data to real type, performs multiplication and converts the result to continuous duration type. Thus, the precision of operation is the same as that of real type operation.

[Reference]
Data range of continuous duration type SPH: 0 to 4,294,967,295 (ms)

(4) Divide time (DIV_)

Symbol

DIV_T_UDI S1, S2 → D : Continuous duration type (S1) ÷ unsigned double-precision integer type (S2)

DIV_T_R S1, S2 → D : Continuous duration type (S1) ÷ real type (S2)

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	—	—	—	—	—	—	—	—	○	○	○	○	—	○	○	○	○	○
S2	—	—	—	—	—	—	—	—	○	○	○	○	—	○	○	○	○	○
D	—	—	—	—	—	—	—	—	—	○	○	○	—	○	○	○	○	—

Function

DIV_T_UDI

1) Divides continuous duration type data “S1” by unsigned double-precision integer type data (S2) and stores the result as continuous duration type data in “D.”
2) When “S2” is of unsigned double-precision integer type and equals or less than the value of ms unit in the operation result, it is <u>truncated</u>.
3) When the divisor is “0,” outputs the boundary value of continuous duration type and ENO goes to 0.

DIV_T_R

1) Divides continuous duration type data “S1” by real type data (S2) and stores the result as continuous duration type data in “D.”
2) When the operation result exceeds the boundary value of the data type, outputs the boundary value of the data type and ENO goes to 0.
   (Boundary values: 0, 49d17h02m47s295ms)
3) When the operation result is minus, the output value is not guaranteed and ENO goes to 0.
4) When “S2” is of real type and equals or less than the value of ms unit in the operation result, it <u>is rounded off</u>.
5) When the divisor is “0,” outputs the boundary value of continuous duration type and ENO goes to 0.

<Operation>

DIV_T_UDI

- When the operation result is within the range of the data type

S1		S2	Operation	D
5h30m	÷	2	⇨	2h45m
				ENO = 1

- When the divisor is 0

S1		S2	Operation	D
5h30m	÷	0	⇨	49d17h02m47s295ms
				ENO = 0

[Reference]
Data range of continuous duration type SPH: 0 to 4,294,967,295 (ms)

DIV_T_R

- When the operation result is within the range of the data type

S1		S2	Operation	D
5h30m	÷	2.5E + 0	⇨	2h12m
				ENO = 1

- When the operation result exceeds the range of the data type

S1		S2	Operation	D
5h30m	÷	1.0E – 4	⇨	49d17h2m47s295ms
				ENO = 0

Note: This instruction converts continuous duration type data to real type, performs multiplication and converts the result to continuous duration type. Thus, the precision of operation is the same as that of real type operation.

- When the divisor is 0

S1		S2	Operation	D
5h30m	÷	0.0E + 0	⇨	49d17h2m47s295ms

[Reference]

Data range of continuous duration type SPH: 0 to 4,294,967,295 (ms)

(5) Concatenate time (CONCAT_D_D)

Symbol

CONCAT_D_D S1, S2 → D

S1 : Date type data
S2 : Duration type data
D : Date + duration type data

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	—	—	—	—	—	—	—	—	○	○	○	○	—	○	○	○	○	○
S2	—	—	—	—	—	—	—	—	○	○	○	○	—	○	○	○	○	○
D	—	—	—	—	—	—	—	—	—	○	○	○	—	—	—	○	○	—

Function

1) Concatenates duration type data “S2” to date type data “S1” and stores the result as date and duration type data in “D.”
2) When the operation result exceeds the boundary value of the data type, the output value is not guaranteed and ENO goes to 0.

<Operation>

DIV_T_UDI

- When the operation result is within the range of the data type

S1		S2		D
1997-06-25	Concate-nation	10:10:00	Operation ⇨	1997-06-25-10:10:00
EN = 1				ENO = 1

- When the operation result exceeds the range of the data type

S1		S2		D
2106-02-07	Concate-nation	10:10:00	Operation ⇨	Undefined
				(1970-01-01-03 : 41 : 44)

Value actually output

(6) Date and time to duration (DT_TO_TOD)

Symbol

DT_TO_TOD S → D

S : Date + duration type data
D : Duration type data

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	—	—	—	—	—	—	—	—	○	○	○	○	—	○	○	○	○	○
D	—	—	—	—	—	—	—	—	—	○	○	○	—	—	—	○	○	—

Function

Extracts the data of duration part from date and duration type data "S" and stores the result as duration type data in "D."

<Operation>

S: 1997-06-25-10:10:00 ⇨ Convert ⇨ D: 10:10:00.00

ENO = 1

(7) Date and time to date (DT_TO_D)

Symbol

DT_TO_D S → D

S : Date + duration type data
D: Date type data

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	—	—	—	—	—	—	—	—	○	○	○	○	—	○	○	○	○	○
D	—	—	—	—	—	—	—	—	—	○	○	○	—	—	—	○	○	—

Function

Extracts the data of date part from date and duration type data "S" and stores the result as date type data in "D."

<Operation>

S: 1997-06-25-10:10:00 ⇨ Convert ⇨ D: 1997-06-25

ENO = 1

3-2-9 Original functions

(1) Set bit (SBIT)

Symbol

SBIT S, N → D

S : Device to be set
N : Bit No.
D : Store destination device

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	–	○	○	○	○	○	○	○	–	○	○	○	○	○
N	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Function

1) Sets the Nth bit of specified device “S” and stores the result in specified device “D.”
2) Data in “S” and “D” must have the same bit width.

<Operation>

- When data of “N” is normally read

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S (WORD)	0	1	1	1	0	1	0	1	0	1	0	1	0	1	1	1

N (UINT) 5

⇩ Set

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D (WORD)	0	1	1	1	0	1	0	1	0	1	1	1	0	1	1	1

ENO = 1

- When data of “N” is normally read

EN = 1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S (DWORD)	0	0	0	1	1	1	0	1	0	1	0	1	1	0	1	0	0	0	0	1	0	1	1	0	0	1	1	0	0	0	1	1

N (UINT) 18

⇩ Set

ENO = 1

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D (DWORD)	0	0	0	1	1	1	0	1	0	1	0	1	1	1	1	0	0	0	0	1	0	1	1	0	0	1	1	0	0	0	1	1

(2) Reset bit (RBIT)

Symbol

RBIT S, N → D

S : Device to be set
N : Bit No.
D : Store destination device

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	–	○	○	○	○	○	○	○	–	○	○	○	○	○
N	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Function

1) Sets the Nth bit of specified device “S” and stores the result in specified device “D.”
2) Data in “S” and “D” must have the same bit width.

<Operation>

• When data of “N” is normally read

S (WORD) EN = 1

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

N (UINT) 5

⇩ Reset

D (WORD) ENO = 1

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

• When data of “N” is normally read

S (DWORD)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	1	0	1	0	1	0	1	1	1	1	0	0	0	0	1	0	1	1	0	0	1	1	0	0	0	1	1

N (UINT) 18

⇩ Reset

ENO = 1

D (DWORD)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	1	0	1	0	1	0	1	1	0	1	0	0	0	0	1	0	1	1	0	0	1	1	0	0	0	1	1

(3) Test bit (TBIT)

Symbol

TBIT S, N → D

S : Device to be tested
N : Bit No.
D : Test result (ON/OFF)

Available devices

	Bit devices										Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	–	–	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○	○	○	–	○	○	○	○	○
N	–	–	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	○
D	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Function

Tests (judges whether 0 or 1) the "N"th bit of specified device "S" and stores the result in "D" (bit output).

<Operation>

- When data of "N" is normally read

S (WORD) 0 1 1 1 | 0 1 0 1 | 0 1 0 1 | 0 1 1 1
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

N (UINT) 5

⇩ Test

D (BOOL) 0

ENO = 1

- When data of "N" is normally read

EN = 1

S (DWORD) 0 0 0 1 | 1 1 0 1 | 0 1 0 1 | 1 0 1 0 | 0 0 0 1 | 0 1 1 0 | 0 1 1 0 | 0 0 1 1
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

N (UINT) 18

⇩ Test

D (BOOL) 0

ENO = 1

(4) Decode (DECODE)

Symbol

DECODE S → D

S : Device to be decoded
D : Device storing decoded result

Available devices

	Word devices (W*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Function

1) Using specified device “S” as the bit number, sets ON only the specified bit and outputs the result to specified device “D.”
2) When 16-bit data is specified in “D,” the low-order four bits are valid. For example, when the input value is “16,” the 0th bit is set ON; when the input value is “17,” the first bit is set ON.
3) When 32-bit data is specified in “D,” the low-order five bits are valid. For example, when the input value is “32,” the 0th bit is set ON; when the input value is “33,” the first bit is set ON.

<Operation>

S (UINT) 5

⇩ Decode

D (WORD) ENO = 1

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

D (UINT) 18 EN = 1

⇩ Decode

D (DWORD) ENO = 1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(5) Encode (ENCODE)

Symbol

ENCODE S → D

S: Device to be encoded
D: Device storing encoded result

Available devices

	Word devices (W*)								Double word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	–

Function

1) Outputs the least significant bit that is ON in specified device “S” to specified device “D.”
2) When “S” is 16-bit data and there are no bits that are ON, outputs “16.”
3) When “S” is 32-bit data and there are no bits that are ON, outputs “32.”

<Operation>

S (WORD)

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

EN = 1

⇩ Encode

D: 5

ENO = 1

S (DWORD)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

⇩ Encode

D: 18

ENO = 1

(6) Bit count (BITCOUNT)

Symbol

—| BITCOUNT S → D |—

S: Device to be counted
D: Device storing counted result

Available devices

	Word devices (W*)								Double word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	–

Function

Counts the number of bits that are ON in specified device “S” and outputs the result to specified device “D.”

<Operation>

S (WORD)	0	0	0	1	0	0	1	1	1	0	1	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

⇩ Count

D (UINT) 5 ENO = 1

EN = 1

S (DWORD)	0	0	0	1	0	1	0	1	1	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

⇩ Count

D (UINT) 7 ENO = 1

(7) Convert string to number (STR_TO_UI)

Symbol

STR_TO_UI S → D

S: Device to be converted (character string data)
D: Device storing converted data

Available devices

	Word devices (W*)								Double word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	–
D	–	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	–

Function

1) Converts string type data in the specified device to unsigned integer type data.
2) String type data must consist of 1-byte digits ranging from “0” to “9.” When other characters are used, the output goes to 0 and ENO to 0.
3) When the converted result exceeds the range of the unsigned integer type, outputs the upper limit of the unsigned integer type and ENO goes to 0.
(Upper limit value: 65535)
4) Two-byte digits are not recognized as digits. When there are 2-byte digits, the output goes to 0 and ENO to 0.

<Operation>

S	‘1’
S+1	‘2’
S+2	‘3’
S+3	‘4’
	H0000

⇨ D 1234

ENO = 1

(8) Convert number to string (UI_TO_STR)

Symbol

UI_TO_STR S → D

S: Device to be converted (unsigned integer data)
D: Device storing converted data

Available devices

	Word devices (W*)								Double word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	–
D	–	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–

Function

Converts unsigned integer type data in specified device “S” to string type data (1-byte digits).

<Operation>

S 1234 ⇨

D	‘1’
D+1	‘2’
D+2	‘3’
D+3	‘4’
	H0000

ENO = 1

(9) Convert shift-JIS to string (SJ_TO_STR)

Symbol

SJ_TO_STR S → D

S: Device to be converted (word type data)
D: Device storing converted data

Available devices

	Word devices (W*)								Double word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	–
D	–	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–

Function

1) Converts shift-JIS code defined as word type data array in the specified device to string type data.
2) The shift-JIS code must end with NULL code (00 or 00 00).
3) When shift-JIS code of 65 or more characters is input, converts 64 characters to string type and ENO goes to 0.

<Operation>

S

A0	82
A2	82
A4	82
42	41
00	00

CONVERT ⇨

D

'あ'
'い'
'う'
'A'
'B'
H0000

ENO = 1

Note: Whether or not the input code actually exist in shift-JIS code is not checked. ENO remains 1 and the output is undefined.

(10) Convert string to shift-JIS (STR_TO_SJ)

Symbol

STR_TO_SJ S → D

S: Device to be converted (string type data)
D: Device storing converted data

Available devices

	Word devices (W*)								Double word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	–
D	–	○	○	○	–	–	○	○	–	–	–	–	–	–	–	–	–	–

Function

1) Converts string type data in specified device “S” to shift-JIS code defined as word type data array and outputs the result to “D.”
   NULL code is added to the end.
2) When the output array capacity is larger than the input, undefined data may be inserted in the positions succeeding the NULL code.
3) When the output array capacity is smaller than the input, the array is filled with shift-JIS code and NULL code is added to the end of the array. ENO goes to 0.

<Operation>

S
’あ’
’い’
’う’
‘A’
‘B’
H0000

CONVERT ⇨

D	
A0	82
A2	82
A4	83
42	41
XX	00
XX	XX
XX	XX
XX	XX

ENO = 1

Note: Because NULL code is added to the end, the output array must be larger than the number of input characters by one word.

(11) Byte length (BYTE_LEN)

Symbol

BYTE_LEN S → D

S: Device to be converted (string type data)
D: Device storing converted data

Available devices

	Word devices (W*)								Double word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	–
D	–	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	–

Function

Outputs the number of bytes, as integer type data, of the shift-JIS code converted from the string type data in specified device “S” to “D.”

<Operation>

S
'あ'
'い'
'う'
'a'
'b'
'え'
'お'
H0000

CONVERT ⇨

D (INT)

12

ENO = 1

Note: A 1-byte character occupies a 1-byte position and a 2-byte character occupies two 1-byte positions.

(12) Dead band (DBAND, DBAND_R)

Symbol

DBAND S1, S2 → D : Integer type (signed) [INT] /double-precision integer type [DINT]

DBAND_R S1, S2 → D : Real type (binary floating-point type) [REAL]

S1: Input device or data
S2: Device where dead band width (value) is stored or the data
D: Device storing result

Available devices

	Word devices (W*)								Double word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	–	○	○	○	○	○	○	○	–	○	○	○	○	○
S2	○	○	○	○	–	○	○	○	○	○	○	○	–	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Function

1) Treats data in specified device "S2" as the dead band width.
   When S1 > |S2|, outputs S1 - |S2| to the device specified in "D."
   When S1 < -|S2|, outputs S1 + |S2| to the device specified in "D."
   In other cases, outputs "0" to the device specified in "D."
2) Data in "S1," "S2" and "D" must have the same bit width.

<Operation>

DBAND

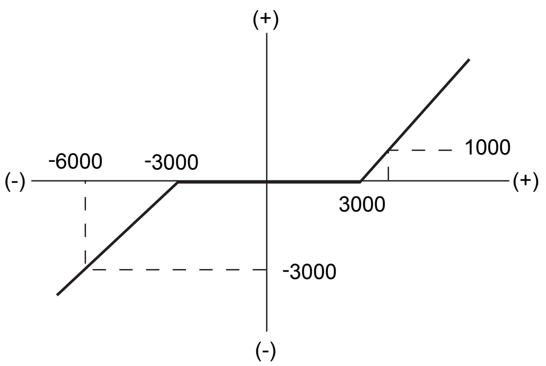
When INT "S2" 3000

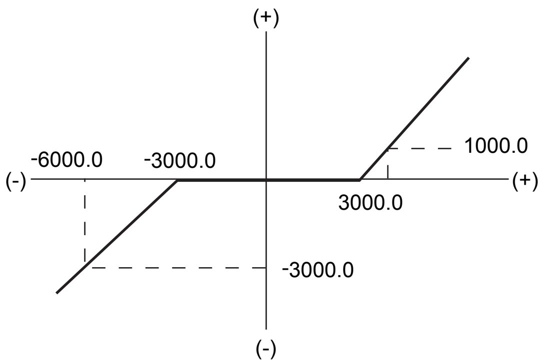
	"S1"		(Output)
INT	4000	→	1000 (ENO = 1)
INT	2000	→	0 (ENO = 1)
INT	-6000	→	-3000 (ENO = 1)

DBAND_R

When REAL "S2" 3.0E + 3

	"S1"		(Output)
REAL	4.0E + 3	→	1.0E + 3 (ENO = 1)
REAL	2.0E + 3	→	0.0E + 0 (ENO = 1)
REAL	-6.0E + 3	→	-3.0E + 3 (ENO = 1)





(13) Bias (BIAS, BIAS_R)

Symbol

BIAS	S1, S2 → D	: Integer type (signed) [INT] /double-precision integer type (signed) [DINT]
BIAS_R	S1, S2 → D	: Real type (binary floating-point type) [REAL]

S1: Input device or data
S2: Device where bias value is stored or the data
D: Device storing result

Available devices

	Word devices (W*)								Double word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	–	○	○	○	○	○	○	○	–	○	○	○	○	○
S2	○	○	○	○	–	○	○	○	○	○	○	○	–	○	○	○	○	○
D	–	○	○	○	–	○	○	○	–	○	○	○	–	○	○	○	○	–

Function

1) Treats specified device “S2” as bias.
   When S1 > 0, outputs S1 + |S2| to the device specified in “D.”
   When S1 < 0, outputs S1 - |S2| to the device specified in “D.”
   When S1 = 0, outputs “0” to the device specified in “D.”
2) Data in “S1,” “S2” and “D” must have the same bit width.

<Operation>

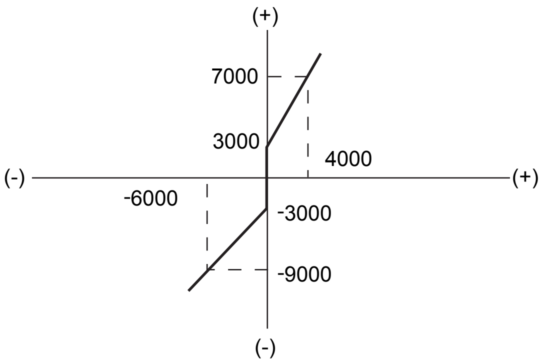
BIAS

When INT “S2” 3000

	“S1”		(Output)	
INT	4000	→	7000	ENO = 1

	“S2”		(Output)	
INT	0	→	0	ENO = 1

	“S2”		(Output)	
INT	-6000	→	-9000	ENO = 1



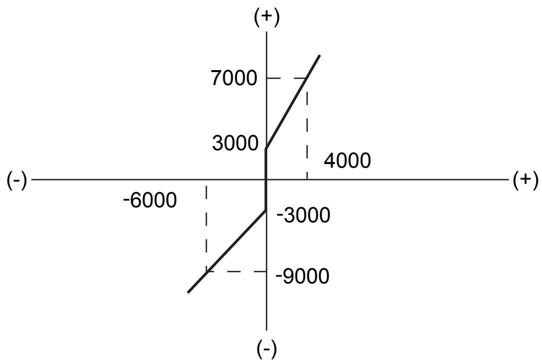
BIAS_R

When REAL “S2” 3.0E + 3

	“S1”		(Output)	
REAL	4.0E + 3	→	7.0E + 3	ENO = 1

	“S1”		(Output)	
REAL	0.0E + 3	→	0.0E + 0	ENO = 1

	“S1”		(Output)	
REAL	-6.0E + 3	→	-9.0E + 3	ENO = 1



(14) Step sequence (SC_COIL/SC)

Symbol

SC_COIL C, N1, S2 → D : Step sequence coil

SC C, N1, S2 → D : Step sequence bit

	SC_COIL	SC
C	Start input	
N1	Step input 1	
N2	Step input 2	
D	Step output	Match output

Available devices

1) SC_COIL

	Bit devices										Word devices (W*)								Double word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
C	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
N1	–	–	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	○
N2	–	–	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	○
D	–	–	–	–	–	–	–	–	–	–	–	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	–

2) SC

	Bit devices										Word devices (W*)								Double word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
C	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
N1	–	–	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	○
N2	–	–	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	–	–	–	–	–	–	–	–	–	○
D	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Function

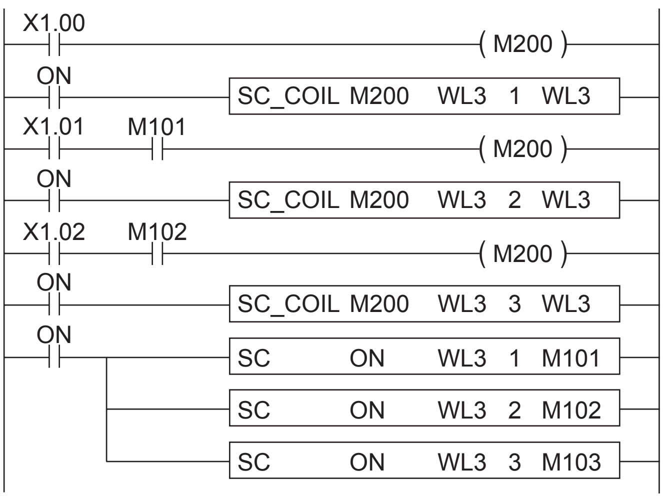
Step control has four functions: self-holding, interlock, step hold at power failure and trailer priority.

<Step sequence coil>
1) When "C" is "0," outputs the value of "N1" to "D." When "C" is "1," outputs the value of "N2" to "D."

<Step sequence bit>
1) When "C" is 1, compares "N1" and "N2"; when they match, outputs "1" to "D"; when not, outputs "0" to "D."

Note: To realize the function for step hold at power failure, the address of N1 must be allocated to the retain memory.

<Program example>
This is an example of the sequential operation circuit. Stepping occurs like M101 → M102 → M103.



(15) Shift 32 bits with carry (SLC, SRC)

Symbol

SLC: Shift left 32 bits with carry:

SLC S1, S2 → D

SRC: Shift right 32 bits with carry:

SRC S1, S2 → D

S1 : Device of data to be shifted and data
S2 : Carry
D : Device storing shifted data

Available devices

	Bit devices										Word devices (w*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	○	○	○	○	○	○	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
D	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	–	○	○	○	○	–

Function

SLC 1) Shifts data in specified device “S1” with carry “S2” leftward by 1 bit and outputs the result to the device specified in “D.”

S1

0	1	1	1	0	1	0	1	0	1	0	1	0	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	1	0	1	0

Shift 1 bit left

D / S2

1	1	1	0	1	0	1	0	1	0	1	0	1	1	1	1	0	1	0	1	0	1	0	1	0	1	1	1	0	1	0	1	←	1

Carry

SRC 1) Shifts data in specified device “S1” with carry “S2” rightward by 1 bit and outputs the result to the device specified in “D.”

S1

1	1	1	0	0	1	1	0	1	0	1	1	0	0	1	1	0	0	0	1	1	1	0	0	1	1	0	1	0	1	1	0

Shift 1 bit right

S2 / D

1	→	1	1	1	1	0	0	1	1	0	1	0	1	1	0	0	1	1	0	0	0	1	1	1	0	0	1	1	0	1	0	1	1

Carry

(16) Shift 32 bits carry (SLCO, SRCO)

Symbol

SLCO: Shift left 32 bits carry

SLCO S → D

SRCO: Shift right 32 bits carry

SRCO S → D

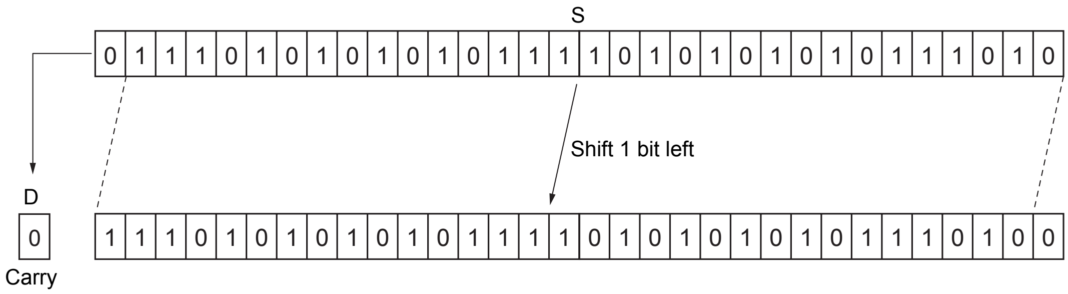
S: Device of data to be shifted and data
D: Device storing carry

Available devices

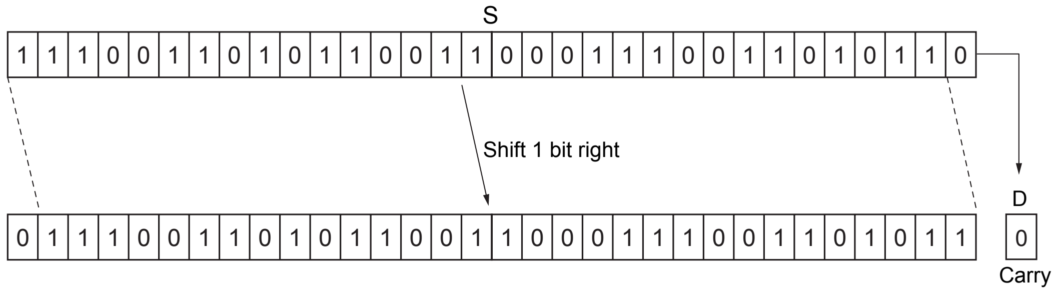
	Bit devices										Word devices (w*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	○	○	○	○	○	○	○
D	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Function

SLCO 1) Shifts data in specified device “S” leftward by 1 bit and outputs the carry to the device specified in “D.”



SRCO 1) Shifts data in specified device “S” rightward by 1 bit and outputs the carry to the device specified in “D.”



(17) 32 bits addition with carry (ADC, ADCO)

Symbol

32 bits addition with carry:

ADC S1, S2, S3 → D

32 bits addition carry:

ADCO S1, S2, S3 → D

S1: Data to be operated or device where data is stored
S2: Data to be operated or device where data is stored
S3: Carry
D : Device storing result

Available devices

ADC:

	Bit devices										Word devices (w*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	○	○	○	○	○	○	○
S2	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	○	○	○	○	○	○	○
S3	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
D	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	–	○	○	○	○	–

ADCO:

	Bit devices										Word devices (w*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	○	○	○	○	○	○	○
S2	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	○	○	○	○	○	○	○
S3	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
D	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

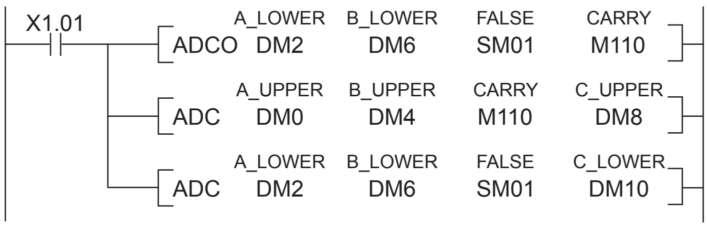
Function

When adding data of 64 or more bits, use ADC and ADCO instructions in combination. These instructions cannot be used individually.

ADC 1) Adds the signed double-precision integer type data (double-word data) in specified devices “S1” and “S2” with a carry in “S3” and outputs the result to the device specified in “D.”

ADCO 1) Adds the signed double-precision integer type data (double-word data) in specified devices “S1” and “S2” with a carry in “S3” and outputs the result to the device specified in “D.”

<Program example>
This is an example of program for adding 64-bit data.



When adding two pieces of 64-bit data, they must be assumed as one piece of 64-bit data because the maximum data length allowed by the MICREX-SX system is 32 bits. In the diagram above, 64-bit data A and B are added and the result is output to C. In this case, the high-order 32 bits of A, B and C are assumed as A_UPPER, B_UPPER and C_UPPER and the low-order 32 bits are assumed as A_LOWER, B_ LOWER and C_ LOWER for calculation. Here, the most significant bit in the low-order 32 bits is treated as a numeric value, not a sign.

[Reference]
When monitoring 64-bit data, use of hexadecimal notations is recommended.

(18) 32 bits subtraction with borrow (SBB, SBBO)

Symbol

32 bits subtraction with borrow:

SBB S1, S2, S3 → D

32 bits subtraction with borrow:

SBBO S1, S2, S3 → D

S1: Data to be operated or device where data is stored
S2: Data to be operated or device where data is stored
S3: Borrow
D : Device storing result

Available devices

SBB:

	Bit devices										Word devices (w*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	○	○	○	○	○	○	○
S2	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	○	○	○	○	○	○	○
S3	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
D	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	–	○	○	○	○	–

SBBO:

	Bit devices										Word devices (w*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	○	○	○	○	○	○	○
S2	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	○	○	○	○	○	○	○
S3	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
D	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

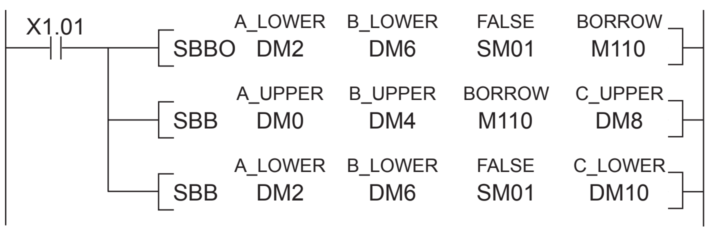
Function

When subtracting data of 64 or more bits, use SBB and SBBO instructions in combination. These instructions cannot be used individually.

SBB 1) Subtracts the signed double-precision integer type data (double-word data) in specified device “S2” from “S1” with a borrow in “S3” and outputs the result to the device specified in “D.”

SBBO 1) Subtracts the signed double-precision integer type data (double-word data) in specified device “S2" from “S1” with a borrow in “S3” and outputs the result to the device specified in “D.”

<Program example>
This is an example of program for subtracting 64-bit data.



When subtracting two pieces of 64-bit data, they must be assumed as one piece of 64-bit data because the maximum data length allowed by the MICREX-SX system is 32 bits. In the diagram above, 64-bit data B is subtracted from A and the result is output to C. In this case, the high-order 32 bits of A, B and C are assumed as A_UPPER, B_UPPER and C_UPPER and the low-order 32 bits are assumed as A_LOWER, B_ LOWER and C_ LOWER for calculation. Here, the most significant bit in the low-order 32 bits is treated as a numeric value, not a sign.

[Reference]
When monitoring 64-bit data, use of hexadecimal notations is recommended.

(19) 64 bits multiplication (low-order digits) (MULL)/64 bits multiplication (high-order digits) (MULU)

Symbol

64 bits multiplication (low-order digits):

```
──[ MULL    S1, S2 → D ]─┤
```

64 bits multiplication (high-order digits):

```
──[ MULU    S1, S2 → D ]─┤
```

S1: Data to be operated or device where data is stored
S2: Data to be operated or device where data is stored
D : Device storing result

Available devices

	Word devices (w*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
S2	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
D	–	–	–	–	–	–	–	–	–	○	○	○	–	○	○	○	○	–

Function

When the operation result becomes 64-bit data, use MULL and MULU instructions in combination. These instructions cannot be used individually.

MULL 1) Multiplies the signed double-precision integer type data (double-word data) in specified device "S1" by "S2" and outputs the low-order 32 bits of the 64-bit result data to the device specified in "D."

MULU 1) Multiplies the signed double-precision integer type data (double-word data) in specified device "S1" by "S2" and outputs the high-order 32 bits of the 64-bit result data to the device specified in "D."

<Program example>

```
     X1.00              DATA_A   DATA_B   C_UPPER
 ├───┤ ├────┬──[ MULU   DM0      DM2      DM4     ]─┤
 │          │           DATA_B   DATA_B   C_LOWER
 │          └──[ MULL   DM0      DM2      DM6     ]─┤
```

* 4 words of the device storing result "D" are required. (DM4 and DM6 in the left exapmle)

In the program above, 32-bit data A and B are multiplied and the result is output to C. In this case, the high-order 32 bits of C is assumed as C_UPPER and the low-order 32 bits is assumed as C_ LOWER for storing. Here, the most significant bit in the low-order 32 bits is treated as a numeric value, not a sign.

[Reference]
When monitoring 64-bit data, use of hexadecimal notations is recommended.

(20) 64 bits division (low-order digits) (DIVL)/64 bits division (high-order digits) (DIVU)

Symbol

64 bits division (low-order digits):

DIVL S1, S2, S3 → D

64 bits division (high-order digits):

DIVU S1, S2, S3 → D

S1: Data of dividend (high-order bits) or device where data is stored
S2: Data of dividend (low-order bits) or device where data is stored
S3: Data of divisor or device where data is stored
D : Device storing result

Available devices

	Word devices (w*)								Double-word devices (D*)									Constants
	X	Y	M	L	SM	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
S2	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
S3	–	–	–	–	–	–	–	–	○	○	○	○	–	○	○	○	○	○
D	–	–	–	–	–	–	–	–	–	○	○	○	–	○	○	○	○	–

Function

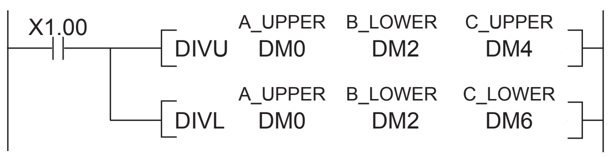
When dividing 64-bit data, use DIVL and DIVU instructions in combination. These instructions cannot be used individually.

DIVL 1) Divides 64-bit data made by combining dividend high-order bits “S1” and dividend “S2” by divisor “S3” and outputs the low-order 32 bits of the result data to the device specified in “D.”

DIVU 1) Divides 64-bit data made by combining dividend low-order bits “S1” and dividend “S2” by divisor “S3” and outputs the low-order 32 bits of the result data to the device specified in “D.”

<Common> 1) When the divisor is “0,” the maximum value matching the dividend sign is output and ENO goes to 0.

<Program example>



* 4 words of the device storing result “D” are required. (DM4 and DM6 in the left exapmle)

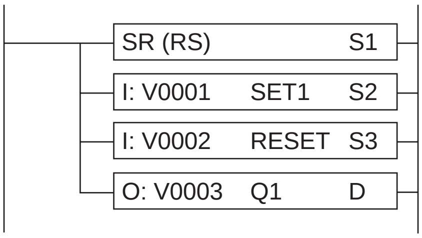
In the program above, 64-bit data A is divided by B and the result is output to C. In this case, the high-order 32 bits of C is assumed as C_UPPER and the low-order 32 bits is assumed as C_ LOWER for storing. Here, the most significant bit in the low-order 32 bits is treated as a numeric value, not a sign.

[Reference]
When monitoring 64-bit data, use of hexadecimal notations is recommended.

3-2-10 IEC standard function blocks

(1) Set reset flip-flop (SR)/Reset set flip-flop (RS)

Symbol



S1: Instance No.
S2: Set
S3: Reset
D : Output

Available devices

	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
D	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–

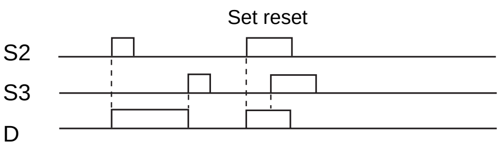
Function

<Parameters>

Parameters	Name	Description
S1	Instance No.	Specifies the instance No. The allowable range is 0 to 65535.
S2	Set request	When this signal goes ON, output "D" is set.
S3	Reset request	When this signal goes ON, output "D" is reset.
D	Output	Specifies the flip-flop output device.

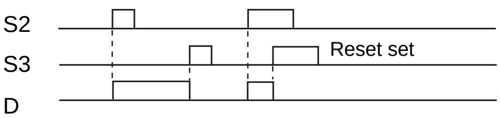
<Operation>

Set reset flip-flop (SR)
When set request “S2” goes ON, output “D” is set. When reset request “S3” goes ON, “D” is reset. When “S2” and “S3” go ON at the same time, “D” is set.



S2	S3	D
1	0	1
1	1	0
0	1	0
0	0	Previous value

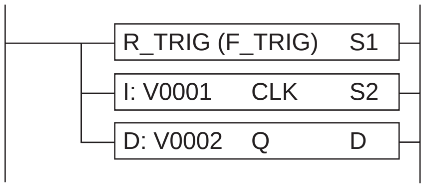
Reset set flip-flop (RS)
When set request “S2” goes ON, output “D” is set. When reset request “S3” goes ON, “D” is reset. When “S2” and “S3” go ON at the same time, “D” is reset.



S2	S3	D
1	0	1
1	1	0
0	1	0
0	0	Previous value

(2) Rising edge trigger (R_TRIG)/Falling edge trigger (F_TRIG)

Symbol



S1: Instance No.
S2: Input
D: Output

Available devices

	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
D	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–

Function

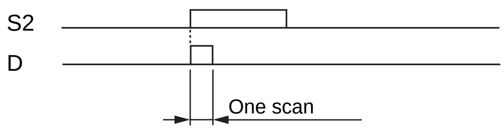
<Parameters>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Input	Inputs a signal whose edge is to be detected.
D	Output	When the rising edge (R_TRIG) or falling edge (F_TRIG) of the signal input to "S2" is detected, this signal goes and remains ON for one scan.

<Operation>

Rising edge trigger (R_TRIG)

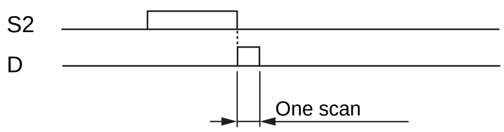
When input “S2” goes ON, its rising edge is detected and output “D”goes and remains ON for one scan.



Falling edge trigger (F_TRIG)

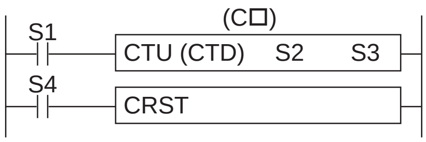
When input “S2” goes OFF, its falling edge is detected and output “D” goes and remains ON for one scan.

Note: The previous value for input “S2” has a retain attribute. That means, the retained previous value is used when starting (warm start). At initial start-up (cold start) or when clearing the retain memory has been specified in option settings for move, it is cleared to 0.



(3) Up counter (CTU)/Down counter (CTD)

Symbol



S1: Count input
S2: Counter No. (C☐) (Count-up output) C0 to C8191: Local counter
C8192 to C16383: Global counter
Note: Loader version V2.2.2.0 or later
S3: Count set value (0 to 32767)
S4: Reset input

Available devices

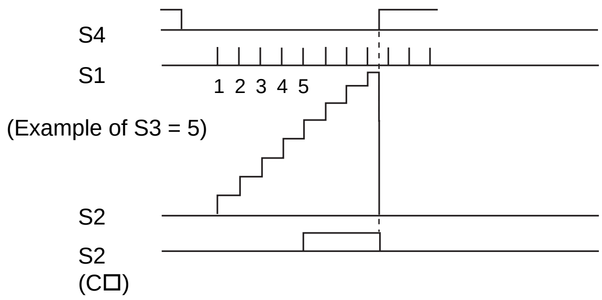
	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S2	–	–	–	–	–	–	–	○	–	–	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	○	○	○	○
S4	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–

Function

<Operation>

Up counter (CTU)

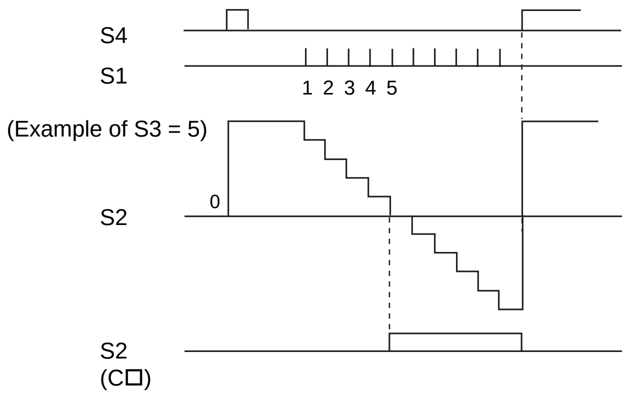
1) CTU increments the current value by 1 on the rising edge of the count input "S1." When the current value exceeds the set value "S3," the count-up output "S2" (counter No.) goes ON.
2) The count set value "S3" can take on a value from 0 to 32767.
3) When using CTU, initially set the reset input "S4" ON to reset the current value.



Note: The reset input resets the count current value but does not set the count-up bit to OFF. Therefore, if the reset input is set to ON when the count set value is "0", the count current value becomes "0" and the count-up bit goes ON.

Down counter (CTD)

1) CTD decrements the current value by 1 on the rising edge of the count input "S1." When the current value reaches the set value "0," the count-up output "S2" (counter No.) goes ON. Countdown continues down to -32768.
2) The count set value "S3" can take on a value from 0 to 32767.
3) When using CTD, initially set the reset input "S4" ON to initialize the current value.

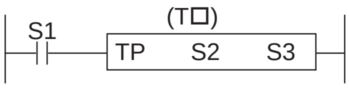


Note: The reset input resets the count current value but does not set the count-up bit to OFF. Therefore, if the reset input is set to ON when the count set value is "0", the count current value becomes "0" and the count-up bit goes ON.

Note: The current counter value has a retain attribute. That means, the retained previous value is used when starting (warm start). At initial start-up (cold start) or when clearing the retain memory has been specified in option settings for move, it is cleared to 0.

(4) Pulse (TP)

Symbol



S1: Pulse start input
S2: Pulse timer No.
(T□) (Pulse output)
S3: Timer set value
(0 to 4294967295 ms)

T0 to T8191: Local counter
T8192 to T16383: Global counter
Note: Loader version V2.2.0.0 or later

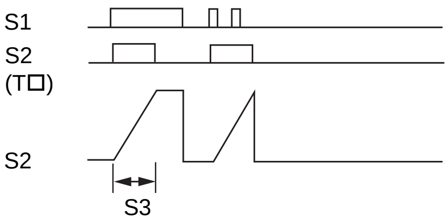
Available devices

	Bit devices										Word devices (W*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–
S2	–	–	–	–	–	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	○	○	○

Function

<Operation>

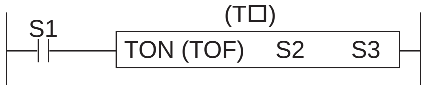
1) When pulse start input “S1”goes ON, pulse output “S2 (T□)” goes ON for the time specified in timer set value “S3.”
2) The allowable range for the timer set value is 0 to 4294967295 ms. (The time base is 1 ms.)
3) When “S3” is 0, “S2” does not go ON.



Notes: 1) Precision of timer instructions
When a timer instruction is executed, a +0 to +2 scan time error occurs to update the elapsed time.
2) The timer compares between “S3” and current time and outputs the result. When the value is modified (increased) by the set value after time-up, the output “S2 (T□)” goes ON and the timer continues its operation from the current value.

(5) On-delay timer (TON)/Off-delay timer (TOF)

Symbol



S1: Timer start input
S2: Timer No.
(T0 to T8191: local timer
T8192 to T16383: global timer)
Note: Loader version: V2.2.0.0 or later
S3: Timer set value
(0 to 4294967295 ms)

Available devices

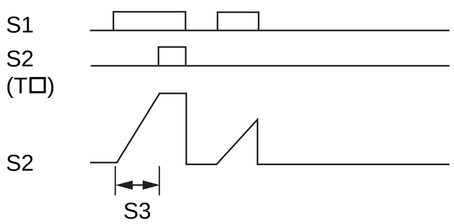
	Bit devices										Double word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–
S2	–	–	–	–	–	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	○	○	○

Function

<Operation>

On-delay timer (TON)

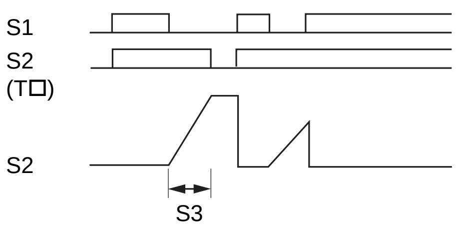
1) When set value “S3”goes ON at the rising edge of realtime input “S1,” the timer starts. When the current value reaches the set time “S3,” time-up signal “S2” (T□) goes ON. When the timer start input goes OFF, the timer current value goes to 0 and also time-up signal “S2” (T□) goes OFF.
2) The allowable range for the timer set value is 0 to 4294967295 ms.
3) When the timer set value is 0, time-up signal “S2” (T□) goes ON immediately after “S1” goes ON.



Note: When on-delay timer is started, it once resets the current value to 0 and starts.

Off-delay timer (TOF)

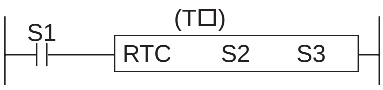
1) When timer start input “S1” goes ON, time-up signal “S2” (T□) goes ON, the timer starts and the current value goes to 0. After “S1” goes OFF, the timer starts and, when the current value reaches the set value, “S2” (T□) goes OFF.
2) The allowable range for the timer set value is 0 to 4294967295 ms.
3) When the timer set value is 0, “S2” (T□) goes OFF immediately after “S1” goes OFF.



Notes: 1) About accuracy of timer instructions
Because the elapsed time is updated during execution of a timer instruction, an error of +0 to +2 scans will occur.
2) When “S1” is ON, the on-delay timer always compares the set value with the current value and outputs the result to “S2.” Therefore, when the set value is changed (increased) after time-up, “S2” goes OFF and the timer starts from the current value.
3) In the case of off-delay timer, when “S1” is 0 and the set value is made greater than the current value, “S2” goes ON and the timer starts. When the current value reaches the set value, “S2” goes OFF.

(6) Real-time clock (RTC)

Symbol



S1: Realtime clock input
S2: Realtime clock output
S3: Realtime clock

T0 to T8191: Local counter
T8192 to T16383: Global counter
Note: Loader version V2.2.0.0 or later

Available devices

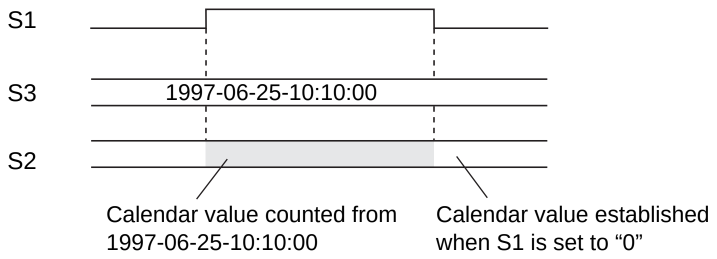
	Bit devices										Word devices (W*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–
S2	–	–	–	–	–	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	○	○	○

Function

<Operation>

1) Fetches the preset value "S3" at the rising edge on an "S1." The current date and time relative to the current value "S3" is output into the current value (T☐).

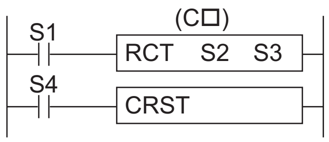
Note: This FB does not set the real-time clock of the system.



3-2-11 Original function blocks

(1) Ring counter (RCT)

Symbol



S1: Count input
S2: Counter No. (C□)
(Count-up output)
S3: Count set value (0 to 32767)
S4: Reset input

C0 to C8191: Local counter
C8192 to C16383: Global counter
Note: Loader version V2.2.2.0 or later

Available devices

	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S2	–	–	–	–	–	–	–	○	–	–	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	○	○	○	○
S4	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–

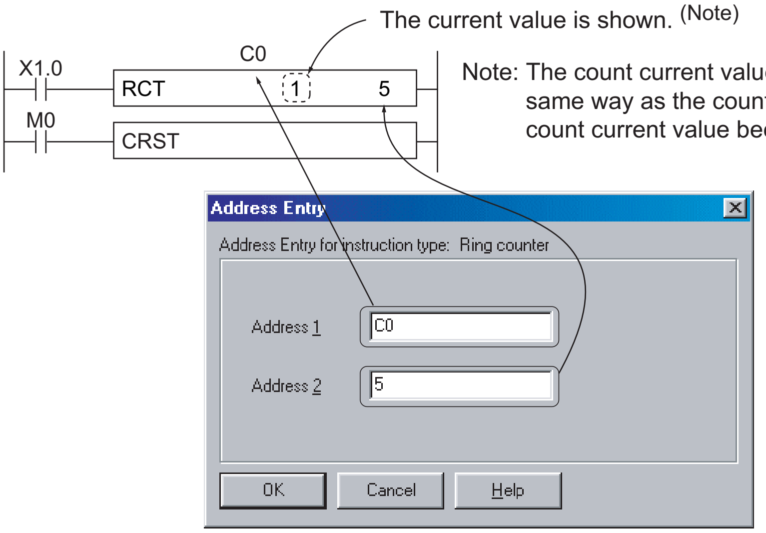
Function

1) Increments the current value by +1 at the rising edge of count input “S1.” When the current value reaches the set value, count-up output “S2” (counter No.) goes ON. At the rising edge of the next count input, the current value and output signal go OFF.
2) The allowable range of counter set value “S3” is 0 to 32767. When “S3” goes to a minus value, this FB does not function.
3) When using the ring counter, first set ON reset input “S4” to reset the current value.

Notes: 1) The count current value has a retain attribute (holding at power failure). That means, when starting (warm start), counting starts from the retained count value. At initial start-up (cold start) or when clearing the retain memory has been specified in option settings for move, it is cleared to 0.
2) When count set value “S3” is changed during operation, the result of comparison between changed “S3” and the current value is stored in count-up output “S2.” When the value of changed “S3” becomes equal to the current value or less, “S2” goes ON and the current value becomes the value of “S3.” In this case, the current value goes to 0 when the next count input is ON.

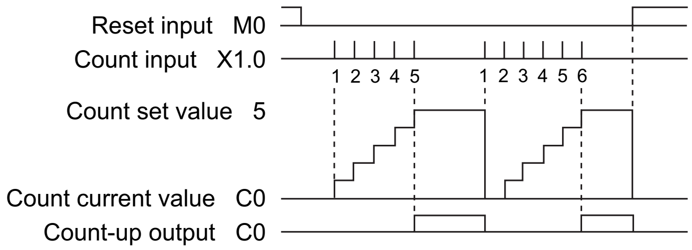
Program examples

<Program examples>



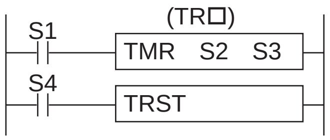
Note: The count current value is treated at “C□” in the same way as the counter No. In this example, the count current value becomes “C0.”

<Operation>



(2) Integrating timer (TMR)

Symbol



S1: Integrating timer start input
S2: Integrating timer No. (TR□)
(Time-up output)
S3: Timer set value
(0 to 4294967295 ms)
S4: Reset input

TR0 to TR8191: Local counter
TR8192 to TR16383: Global counter
Note: Loader version V2.2.2.0 or later

Available devices

	Bit devices										Double word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–
S2	–	–	–	–	–	–	–	○	–	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	○	○	○
S4	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–

Function

1) The timer starts when integrating timer start input “S1” goes ON while reset input “S4” is OFF and pauses when “S1” goes OFF. The timer restarts when “S1” goes ON again.
2) When the current value reaches the set value, time-up signal “S2” (integrating timer No.) goes ON.
3) The allowable range of set time is 0 to 4294967295 ms. The time base is 1 ms.

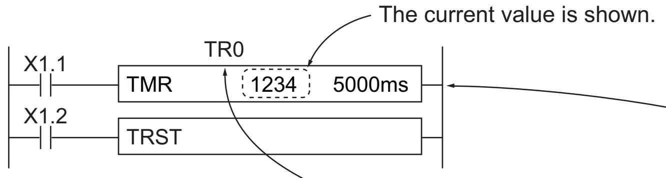
Notes: 1) The timer current value has a retain attribute (holding at power failure). That means, when starting (warm start), counting starts from the retained count value. At initial start-up (cold start) or when clearing the retain memory has been specified in option settings for move, it is cleared to 0.

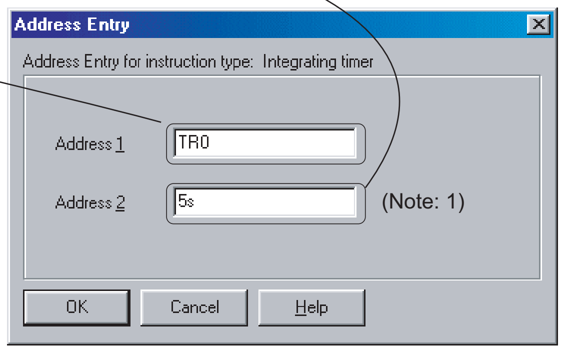
2) When timer set value “S3” is changed while integrating timer start input “S1” is ON, the result of comparison between changed “S3” and the current value is stored in time-up output “S2” regardless of the current output. When a value less than the current value is set in “S3,” “S2” goes ON and the current value becomes the same value as “S3.”

3) About accuracy of timer
Because the elapsed time is updated during execution of a timer instruction, an error of +0 to +2 scans will occur.

Program examples

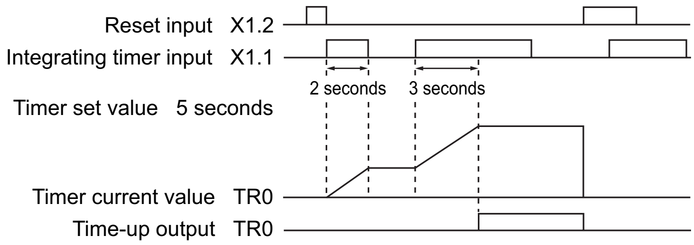
<Program examples>





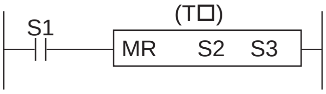
Notes: 1) The time base for this instruction is 1 ms, but the set value may be input as time + time symbol.
Second (s), minute (M), hour (H), day (D)
2) The timer current value is treated as "TR□" in the same way as the integrating timer No. In this example, the integrating timer current value becomes “TR0.”

<Operation>



(3) Retriggerable timer (MR)

Symbol



S1: Timer start input
S2: Timer No. (T☐)
(Time-up signal)
S3: Timer set value
(0 to 4294967295 ms)

T0 to C8191: Local counter
T8192 to T16383: Global counter
Note: Loader version V2.2.0.0 or later

Available devices

	Bit devices										Double word devices (D*)									Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	F	V	
S1	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–
S2	–	–	–	–	–	○	–	–	–	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	○	○	○

Function

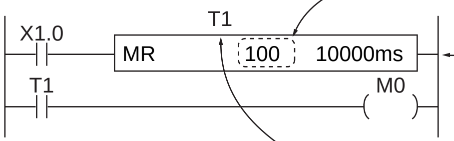
1) The timer starts at the rising edge of timer start input “S1” and time-up signal remains ON until the set time is reached.
2) The timer restarts when the input signal rises again.
3) When the current value reaches the timer set value “S3,” the output becomes undefined (0 or holds the set value).
4) The allowable range of set time is 0 to 4294967295 ms. The time base is 1 ms.

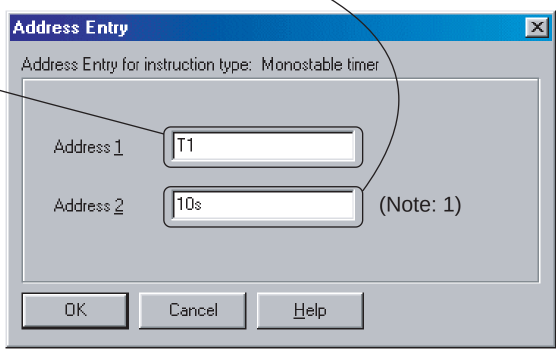
Note: When timer set value “S3” is changed, the result of comparison between changed “S3” and the current value is stored in time-up output “S2.” When a value less than the current value is set in “S3,” “S2” goes OFF. The current value becomes the same value as “S3” if “S1” is ON and goes to 0 if “S1” is OFF. Because the elapsed time is updated during execution of a timer instruction, an error of +0 to +2 scans will occur.

Program examples

<Program examples>

The current value is shown.

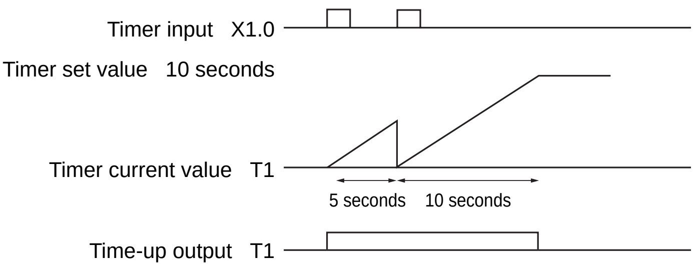




(Note: 1)

Notes: 1) The time base for this instruction is 1 ms, but the set value may be input as time + time symbol.
Second (s), minute (M), hour (H), day (D)
2) The timer current value is treated as “T☐” in the same way as the integrating timer No. In this example, the integrating timer current value becomes “T1.”

<Operation>



(4) Open channel (M_OPEN)

Symbol

M_OPEN		S1
I: V0001	NE_C	S2
I: WV0002	MODULE_NO	S3
I: WV0003	CHANNEL_NO	S4
I: DV0004	STATION_NO	S5
I: WV0005	MODULE_TYPE	S6
I: WV0006	MODE	S7
I: WV0007	SUB_MODE	S8
I: WV0008	P_PORT_NO	S9
I: WV0009	S_PORT_NO	S10
O: V0010	VALID	D1
O: V0011	ERROR	D2
O: WV0012	STATUS	D3
O: WV0013	CON_NO	D4

S1: Instance No.
S2: Open request
S3: Module No.
S4: Channel No.
S5: Station No.
S6: Module type
S7: Communication mode
S8: Communication sub-mode
S9: Communication partner port No.
S10: Local port No. (1 to 127)
D1: Open enable flag
D2: Open error flag
D3: Error status
D4: Connection No.

Available devices

	Bit devices										(Double) Word devices (W*, D*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S5	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S6	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S7	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S8	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S9	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S10	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
D1	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
D4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–

* Only for station No. "S5," a double-word device is used.

Function

This FB is used for setting a communication port for message communication. Use this FB in combination with M_SEND (send message) or M_RECEIVE (receive message).

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Open request	When this signal goes ON, open processing starts. When this signal goes OFF, close processing starts.
S3	Module No.	Specify the SX station No. of the communication module used for communications outside the configuration or that of the communication partner CPU for communications inside the configuration.
S4	Channel No.	Specify the channel No. in the communication module. The module with only one communication port is fixed to 0.
S5	Station No.*	Specify the station No. of the communication partner on the network. For example, it is an IP address for Ethernet and P/PE link station No. for P/PE link. It is meaningless for communications inside the configuration.
S6	Module type	Specify whether the communication partner is inside or outside the configuration. 0: Communications inside configuration 1: Communications outside configuration
S7	Communication mode	Settings for the communication partner module. For details, refer to the communication module manual. For example, this is fixed to 0 for a P/PE link.
S8	Communication sub-mode	
S9	Communication partner port No.	Specify the communication partner port No.
S10	Local port No.	Set port No. 1 to 127 to be set on the SX bus for communications inside the configuration. For communications outside the configuration, it also means the receive port No. to be set on the network used.
D1	Open enable flag	Goes ON when an open request is issued and the port can be opened normally. When this is ON, it indicates that the port is opened normally.
D2	Open error flag	Remains ON for one scan when the port could not be opened.
D3	Error status	Indicates error code.
D4	Connection No.	Management No. of the port that has completed opening.

1) When open request “S2” is set ON, open processing starts. When the processing has completed normally, open enable “D1” goes ON and the connection No. is output to connection number “D4.”
When open request is set OFF, close processing is performed.
2) Passes connection No. “D4” to connection No. “S3” of M_SEND FB or M_RECEIVE FB to create communication program.

* The IP address is set as follows. Example: When IP address is 172.16.0.1

172	16	0	1
ACh	10h	00h	01h

⇨ UDINT#16#AC100001

Note: Open or close processing does not complete in one scan.

<Open status>

Status	Description
177 (B1h)	Parameter error - Input value exceeding the allowable range. - Module No. "S3" exceeding the allowable SX bus station No. range (1 to FE). - Input value to module type "S6" is other than the defined values. - Module No. "S3" indicates the local station No.
193 (C1h)	Channel open error - An illegal value is set in station No. "S5." (Normal values depend on communication modules.) - An illegal value is set for communication mode. (Normal values depend on communication modules.) - Communication mode is set to active (send side), but the partner station No. (IP address) and communication partner port No. "S9" are not on the network (only when the Ethernet module is used).
200 (C8h)	Port specification error - Code specified in local port No. "S10" is not in the range of 1 to 127. - Same local port No. "S10" is already used in the resource.
201 (C9h)	Connection No. full - An attempt was made to open 57 or more ports in the resource. - An attempt was made to open ports more than those defined in one communication module (the definition depends on communication modules).

<About communication ports S9 (communication partner port No.) and S10 (local port No.)>

A communication port is used to indicate the slot used for communications in the communication partner module specified in S5 (station No.) and/or S3 (module No.).
MICREX-SX allows assigning numbers 1 to 127 to local ports.

1) Communication port for communications in the configuration

For communications in the configuration, the value specified in parameter “S10” of M_OPEN becomes the local port No. and it is used as a communication slot for receiving messages from the communication partner. The value in parameter “S9” becomes the destination port No. of the communication partner.

2) Communication port for communications outside the configuration

For communications outside the configuration, the value specified in parameter “S10” of M_OPEN becomes the local port No. and a communication slot for receiving messages from the communication partner is prepared on the communication module network. The value in parameter “S9” becomes the destination port No. of the communication partner (communication module).
Note that for some network (TCP/IP, etc.), allowable port Nos. are limited and port Nos. 1-127 may not be available. In such a case, use the sum of the value specified in parameter “S10” and a certain offset value as the port No. to be set on the network. In the same way, the sum of the value specified in parameter “S9” and an offset should be used. The offset value can be set using parameters for communication module system definitions.

(5) Send message (M_SEND)

Symbol

M_SEND		S1
I: V0001	REQ	S2
I: WV0002	CON_NO	S3
IO: _V0003	SD	S4
I: WV0004	SIZE	S5
O: V0005	DONE	D1
O: V0006	ERROR	D2
O: WV0007	STATUS	D3

S1: Instance No.
S2: Send request
S3: Connection No.
S4: Send data
S5: Send data word count
D1: Normal completion flag
D2: Error flag
D3: Status

Available devices

	Bit devices										(Double) Word devices (W*, D*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	○	○	○	–
S4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	○	○	–
S5	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	○	○	○	○
D1	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D3	–	–	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	–	○	○	–

* Only for send data “S4,” a double-word device can be specified.

Function

This FB is used for sending messages to a destination module specified in M_OPEN (open channel). Use this FB in combination with M_OPEN.

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Send request	Message send processing starts at the rising edge of this signal. Though this signal is effective at its rising edge, it is ineffective at its rising edge during data send (until "D1" or "D2" goes ON after this signal goes ON).
S3	Connection No.	Specify the same device as that specified in parameter "D4" of M_OPEN used in pair with this FB.
S4	Send data heading address	Specify the heading address of the data to be sent. The specified heading address must be an even address.
S5	Send data word count	Specify the send data word count.
D1	Normal completion flag	When data send has completed normally, this flag remains ON for one scan.
D2	Error flag	When data send has been disabled, this flag remains ON for one scan.
D3	Status	When there is an error in message sending, the error cause is set. It goes to "0" at normal completion.

<Operations>

1) At the rising edge of send request “S2,” the message of the number of words specified in “S5” beginning at the address specified in “S4” (heading address) is sent.
2) Sending is not completed in one scan. When sending has completed normally, normal completion flag “D1” remains ON for one scan. When sending has terminated abnormally, error flag “D2” remains ON for one scan and the error cause is stored in status “D3.”
3) Specify the communication partner in M_OPEN used in pair. It is made pair by specifying the address of connection number “D4” of M_OPEN in connection number “S3” of this FB.

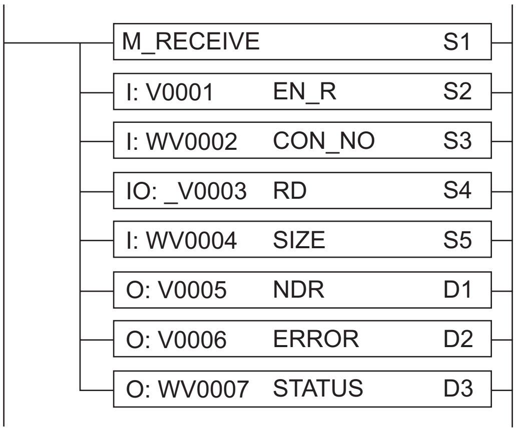
Notes: 1) The size of data that can be sent as one message is 2048 words for communications inside the configuration. For communications outside the configuration, it depends on the communication module used.
2) During message processing, do not change send data “S4.” If changed, send data is not guaranteed.
3) When the number of data pieces specified in send data word count "S5” exceeds the size of the variable specified in S4, excessive data may be undefined. In “S5,” be sure to input a size not exceeding the specified variable area.
4) Program send request “S2” so that it goes ON after open enable “D1” of M_OPEN goes ON.
5) When the communication partner is outside the configuration, M_SEND ends regardless of whether the M_RECEIVE reception status of the communication partner.
In addition, when the communication partner is inside the configuration, it ends upon normal completion of M_RECEIVE by the communication partner.

<Status>

Status	Description
66 (42h)	Memory access error Accessing to nonexistent P/PE link memory or FL-net common memory
162 (A2h)	No command response Receiving no response to command after a predetermined period of time
164 (A4h)	Message send error The communication partner is missing or no module exists for the specified SX station No.
165 (A5h)	Message receive busy The message communication partner is busy on the SX bus and message sending is disabled.
170 (AAh)	Message send busy The resource in the CPU for sending messages is busy and message send is disabled.
177 (B1h)	Parameter error - The specified send data exceeds the memory area. - Send data word count "S5" is 0.
195 (C3h)	Message send error - No messages can be sent to the communication module of the communication partner. - No response returns from the communication module of the communication partner. (Sending has completed, but no send ACK returned.)
197 (C5h)	Network send busy During message communications through the network between communication modules, the communication module of the communication partner is busy and message send is disabled.
200 (C8h)	Port specification error The communication port of the communication partner is not open.
206 (CEh)	Buffer over - The send data count has exceeded 2048 words. - For communications through communication modules, the send data limit for the communication module has been exceeded.
207 (CFh)	Connection No. error A connection No. not open is used. An attempt was made to send using a connection No. being used for sending (this occurs when two M_SEND instructions are used in parallel).

(6) Receive message (M_RECEIVE)

Symbol



S1: Instance No.
S2: Receive enable
S3: Connection No.
S4: Receive data
S5: Receive data area word count
D1: Normal completion flag
D2: Error flag
D3: Status

Available devices

	Bit devices										(Double) Word devices (W*, D*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	○	○	○	○	○	–	–	○	○	○	○
S4	–	–	–	–	–	–	–	–	–	–	–	○	○	○	○	–	–	–	○	○	○
S5	–	–	–	–	–	–	–	–	–	–	○	○	○	○	○	–	–	○	○	○	○
D1	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D3	–	–	–	–	–	–	–	–	–	–	–	○	○	○	○	–	–	–	○	○	○

* Only for receive data “S4,” a double-word device can be specified.

Function

This FB is used for receiving messages from a source module specified in M_OPNE (open channel). (Message receive)
Use this FB in combination with M_OPEN.

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Receive enable	Message receive processing starts when this signal goes ON. During message receive processing (until "D1" or ""D2" goes ON), keep this signal ON.
S3	Connection No.	Specify the same device as that specified in parameter "D4" of M_OPEN used in pair with this FB.
S4	Receive data heading address	Specify the heading address of the device that stores the receive data. The heading address must be an even address.
S5	Receive data area word count	Specify the number of words in the receive data area.
D1	Normal completion flag	When data receive has completed normally, this flag remains ON for one scan.
D2	Error flag	When data receive has been disabled, this flag remains ON for one scan.
D3	Status	When there is an error in message receiving, the error cause is set. It goes to "0" at normal completion.

<Operations>

1) When receive enable “S2” goes ON, message data received from the partner is stored in the receive data area specified in “S4” and “S5.”
2) Receive processing is not completed in one scan. When receive has completed normally, normal completion flag “D1” remains ON for one scan. When receive has terminated abnormally, “D2” remains ON for one scan and the error cause is stored in status “D3.”
3) Specify the communication partner in M_OPEN used in pair. It is made pair by specifying the address of M_OPEN connection number “D4” in connection number “S3” of this FB.

Notes:
1) The size of data that can be received as one message is 2048 words for communications inside the configuration. For communications outside the configuration, it depends on the communication module used.
2) When receive has paused, it is restarted by setting “S2” ON. Even when “S3,” “S4” and “S5” have been changed at this time, receive will restarts with the input value used before the pause. The changes are not reflected to the message receive processing.
3) After completing message receive processing, new message receive processing will start if “S2” is ON in the next scan.
4) During receive processing, keep “S4” ON. When “S4” is rewritten, the receive data is not guaranteed.
5) Be careful so that the receive data area specified in “S5” and “S4” is not overlapped with other data areas. If they overlap, other data areas may be overwritten.
6) Program “S2” so that it goes ON after open enable “D1” of M_OPEN goes ON.

<Status>

Status	Description
66 (42h)	Memory access error Accessing to nonexistent P/PE link memory or FL-net common memory
162 (A2h)	No command response Receiving no response to command after a predetermined period of time
164 (A4h)	Message send error The communication partner is missing or no module exists for the specified SX station No.
170 (AAh)	Message send busy The resource in the CPU for sending messages is busy and message send is disabled.
177 (B1h)	Parameter error - The specified send data exceeds the memory area. - Receive data area word count "S5" is 0.
196 (C4h)	Abnormal termination of RECEIVE
199 (C7h)	Channel close The port of the communication partner was closed during communication outside the configuration.
200 (C8h)	Port specification error The communication port of the communication partner is not open.
206 (CEh)	Buffer over - The receive data count has exceeded 2048 words. - For communications through communication modules, the send data limit for the communication module has been exceeded.
207 (CFh)	Connection No. error A connection No. not open is used. An attempt was made to send using a connection No. being used for receiving (this occurs when two M_SEND instructions are used in parallel).

(7) Direct read (READ_W, READ_B)

Symbol

READ_W(B)		S1
I: V0001	REQ	S2
I: WV0002	SIZE	S3
IO: _V0003	GLOBAL_VAR	S4
IO: _V0004	RD	S5
O: V0005	DONE	D1
O: V0006	ERROR	D2
O: WV0007	STATUS	D3

S1: Instance No.
S2: Read request
S3: Read data size
S4: Partner read device
S5: Read data storage device
D1: Normal completion flag
D2: Error flag
D3: Status

Available devices

	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	○	○	○	○
S4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S5	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
D1	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–

Function

Use this FB to read devices in other resources (CPU modules, P/PE link modules, etc.) in the same configuration. When reading devices in resources on the same processor bus, Move instruction, etc. may be used.

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Read request	Read processing will start at the rising edge of this signal. Though this signal is effective at its rising edge, it is ineffective at its rising edge during data receive (until "D1" or "D2" goes ON after this signal goes ON).
S3	Read data size	Specify the size of the device to read data. READ_W: Word count, READ_B: Bit count
S4	Partner read device	Specify the device (heading address) of the partner resource (CPU modules, P/PE links, etc. in the same configuration) to be read. Note: The specified address must be an even address.
S5	Read data storage device	Specify the storage destination (heading address) of the read data. Note: The specified address must be an even address.
D1	Normal completion flag	When data read has completed normally, this flag remains ON for one scan.
D2	Error flag	When data read has terminated abnormally, this flag remains ON for one scan.
D3	Status	Indicates the read result. It goes to "0" at normal completion.

<Operations>

1) At the rising edge of read request “S2,” the device in the partner resource specified in “S4” is read to read data storage destination device “S5” (heading address).
2) Reading is not completed in one scan. When reading has completed normally, normal completion flag “D1” remains ON for one scan. When reading has terminated abnormally, error flag “D2” remains ON for one scan and the error cause is stored in status “D3.”

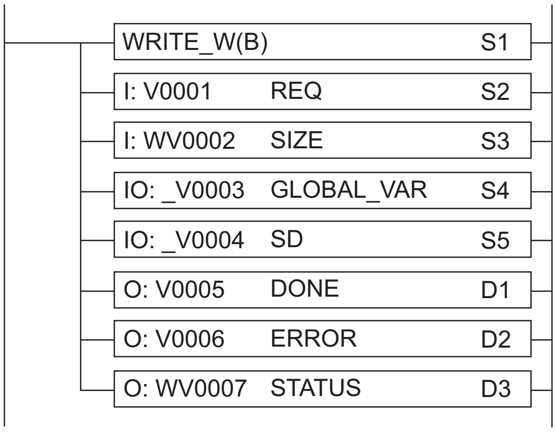
Notes: 1) During read processing, do not change “S5.” If changed, the read data is not guaranteed.
2) The read data of the size specified in “S3” is stored beginning with the device specified in “S5” (heading address). Be careful so that it is not overlapped with other data areas. If they overlap, other data areas will be overwritten.

<Status>

Status	Description
66 (42h)	Memory access error Accessing to nonexistent P/PE link memory or FL-net common memory
162 (A2h)	No command response Receiving no response to command after a predetermined period of time
164 (A4h)	Message send error The communication partner is missing or no module exists for the specified SX station No.
165 (A5h)	Message receive busy The message communication partner is busy on the SX bus and message sending is disabled.
170 (AAh)	Message send busy The resource in the CPU for sending messages is busy and message send is disabled.
176 (B0h)	Global device specification error A device of the local CPU has been specified in partner read device "S4."
177 (B1h)	Parameter error - An area for storing read data is out of the memory area. - Read data sizes "S3" is 0.
201 (C9h)	No empty ports An attempt was made to open ports exceeding the defined number in one communication module. READ_W/READ_B also use communication ports. For example, READ_W used for a device in the P link uses one communication port of the P link. There are five communication ports available for the P/PE link.

(8) Direct write (WRITE_W, WRITE_B)

Symbol



S1: Instance No.
S2: Write request
S3: Write data size
S4: Partner writing device
S5: Write data
D1: Normal completion flag
D2: Error flag
D3: Status

Available devices

	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	○	○	○	○
S4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S5	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
D1	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–

Function

Use this FB to write data in devices in other resources (CPU modules, P/PE link modules, etc.) in the same configuration. When writing data in devices in resources on the same processor bus, Move instruction, etc. may be used.

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Write request	Write processing will start at the rising edge of this signal. Though this signal is effective at its rising edge, it is ineffective at its rising edge during data write (until "D1" or "D2" goes ON after this signal goes ON).
S3	Write data size	Specify the size of the data to be written. WRITE_W: Word count, WRITE_B: Bit count
S4	Partner writing device	Specify the device (heading address) of the write destination resource (CPU modules, P/PE links, etc.). Note: The specified address must be an even address.
S5	Write data	Specify the device (heading address) where the write data is stored. Note: The specified address must be an even address.
D1	Normal completion flag	When data write has completed normally, this flag remains ON for one scan.
D2	Error flag	When data write has terminated abnormally, this flag remains ON for one scan.
D3	Status	Indicates the write result. It goes to "0" at normal completion.

<Operations>

1) At the rising edge of write request “S2,” the write data prepared in “S5” (heading address) is written to the device (write destination) in the partner resource of “S4.”
2) Writing is not completed in one scan. When writing has completed normally, normal completion flag “D1” remains ON for one scan. When writing has terminated abnormally, error flag “D2” remains ON for one scan and the error cause is stored in status “D3.”

Key point

WRITE_W can write data in the waiting CPU in the 1-to-1 redundant system warm standby configuration.
This function allows data equalization with the application program.

Notes: 1) During write processing, do not change “S5.” If changed, the write data is not guaranteed.
2) Specify “S3” so that it is not overlapped with other data areas (to prevent other devices from being overwritten). If they overlap, other data areas of the partner will be overwritten.
3) WRITE_B cannot be used for devices in P/PE links or FL-net modules.
4) When writing data in the waiting CPU (data equalization with application), “S3” guarantees uniqueness as long as S3 <= 240.

<Status>

Status	Description
35 (23h)	Transmission interlock Write processing was performed while the communication partner was being interlocked (program transfer, etc.).
66 (42h)	Memory access error Accessing to nonexistent P/PE link memory or FL-net common memory
162 (A2h)	No command response Receiving no response to command after a predetermined period of time
164 (A4h)	Message send error The communication partner is missing or no module exists for the specified SX station No.
165 (A5h)	Message receive busy The message communication partner is busy on the SX bus and message sending is disabled.
170 (AAh)	Message send busy The resource in the CPU for sending messages is busy and message send is disabled.
176 (B0h)	Global device specification error - When not in the 1-to-1 redundant system warm standby configuration, the device specified in "S4" is the device of the local CPU. - In the 1-to-1 redundant system warm standby configuration, the device specifie in "S4" indicates the high-speed memory area of the local CPU.
177 (B1h)	Parameter error The write data size "S3" is 0.
201 (C9h)	No empty ports An attempt was made to open ports exceeding the defined number in one communication module. READ_W/READ_B also use communication ports. For example, WRITE_W used for a device in the P link uses one communication port of the P link. There are five communication ports available for the P/PE link.

(9) Remote data read (R_READ)

Symbol

R_READ		S1
I: V0001	REQ	S2
I: WV0002	MODULE_NO	S3
I: WV0003	CHANNEL_NO	S4
I: DV0004	STATION_NO	S5
I: WV0005	VAR_TYPE	S6
IO: _V0006	REMOTE_VAR	S7
I: WV0007	SIZE	S8
IO: _V0008	RD	S9
O: V0009	DONE	D1
O: V0010	ERROR	D2
O: WV0011	STATUS	D3

S1: Instance No.
S2: Read request
S3: Communication module SX bus station No.
S4: Communication module channel No.
S5: Station No.
S6: Device specification method
S7: Read address
S8: Read data size
S9: Read data storage device
D1: Normal completion flag
D2: Error flag
D3: Status

Available devices

	Bit devices										(Double) Word devices (W*, D*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S5	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S6	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S7	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S8	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S9	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
D1	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–

* Only for station No. "S5," a double-word device is used.

Function

This FB reads data about devices on the network through the communication modules connected to the same SX bus.
It can read data about the following devices:
- Memory in SPH-series CPU modules through network (independent of types)
- Memory in MICREX-F/FLEX-PC CPU through T/P/PE links
- Memory of devices connected to the network of OPEN standard such as OPCN-1, FL-net

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Read request	Read processing will start at the rising edge of this signal. Though this signal is effective at its rising edge,it is ineffective at its rising edge during data read (until "D1" or "D2" goes ON after this signal goes ON).
S3	Communication module SX bus station No.	Specify the SX bus station No. of the communication module connected to the SX bus same as the CPU module.
S4	Communication module channel No.	Specify the channel No. in the communication module. For modules such as P/PE link modules with only one communication port, this is fixed to 0.
S5	Station No.	Specify the station No. of the communication partner on the network. For example, it is an IP address for Ethernet or P/PE link station No. for P/PE link.
S6	Device specification method	The method for specifying the read memory (device) depends on the access target of the communication partner. For details, see <Device specification method>.
S7	Read address	The read address specification depends on the device specification method. For details, see <Address specification format>.
S8	Read data size	Specify the number of words to be read, beginning at the address specified in "read address" (in word units).
S9	Read data storage device	Specify the storage destination (heading address) of the read data.
D1	Normal completion flag	When data read has completed normally, this flag remains ON for one scan.
D2	Error flag	When data read has terminated abnormally, this flag remains ON for on scan.
D3	Status	Indicates the read result. It goes to "0" at normal completion.

<Operations>

1) At the rising edge of READ request “S2,” memory of network station “S5” is read through the communication port of the communication module specified in “S3” and “S4.”
2) The network station memory specification is determined by read address “S7” that matches the format specified in device specification method “S6.” For details, see page 3-147.
3) Reading is not completed in one scan. When reading has completed normally, normal completion flag “D1” remains ON for one scan. When reading has terminated abnormally, “D2” remains ON for one scan and the error cause is stored in status “D3.”

Notes: 1) During read processing, do not change data storage device “S9.” If changed, the read data is not guaranteed.
2) Read data size “S8” is limited by the communication module used.
3) Specify read data size “S8” and read data storage device “S9” (heading address) so that the read data is not overlapped with other data areas than the read data storage device area. If they overlap, data will be overwritten.
4) When data read from a device other than SPH series devices (“S6” $\neq$ 0) is less than read data size “S8,” processingcompletes after reading that data only.

<Status>

Status	Description
68 (44h)	Memory address specification error The address specified in read address "S7" has an error.
69 (45h)	Memory size over The memory size exceeds the allowable address range of the partner CPU module specified in read address "S7" and read data size "S8." In this case, the read data is not guaranteed.
160 (A0h)	Communication partner specification error The communication partner is SPH (S6 = 0) and the CPU specified in read address "S7" does not exist.
162 (A2h)	No command response Receiving no response to command after a predetermined period of time
164 (A4h)	Message send error The communication partner is missing or no module exists for the specified SX station No.
165 (A5h)	Message receive busy The message communication partner is busy on the SX bus and message sending is disabled.
170 (AAh)	Message send busy The resource in the CPU for sending messages is busy and message send is disabled.
171 (ABh)	Internal resource exhausted The internal resource for executing R_READ or R_WRITE has been exhausted.
177 (B1h)	Parameter error - SX bus station No. is not 1 to FE. - Read data size "S8" is 0. - Device specification method "S6" is other than the specified value.
193 (C1h)	Channel open error The value specified in channel number "S4" is abnormal.
195 (C3h)	Message send error - The value specified in network station number "S5" is abnormal. - The value specified in channel number "S4" is abnormal. - The value specified in read address "S7" is abnormal.
197 (C5h)	Network send busy The communication module of the communication partner is busy and message send is disabled.
201 (C9h)	No empty ports An attempt was made to open ports exceeding the defined number in one communication module. READ_W/READ_B also use communication ports. For example, READ_W used for a device in the P link uses one communication port of the P link. There are five communication ports available for the P/PE link.
206 (CEh)	Buffer over The message data size limit value of the communication module used is exceeded.

(10) Remote data write (R_WRITE)

Symbol

R_WRITE		S1
I: V0001	REQ	S2
I: WV0002	MODULE_NO	S3
I: WV0003	CHANNEL_NO	S4
I: DV0004	STATION_NO	S5
I: WV0005	VAR_TYPE	S6
IO: _V0006	REMOTE_VAR	S7
I: WV0007	SIZE	S8
IO: _V0008	SD	S9
O: V0009	DONE	D1
O: V0010	ERROR	D2
O: WV0011	STATUS	D3

S1: Instance No.
S2: Write request
S3: Communication module SX bus station No.
S4: Communication module channel No.
S5: Station No.
S6: Device specification method
S7: Write address
S8: Write data size
S9: Write data storage device
D1: Normal completion flag
D2: Error flag
D3: Status

Available devices

	Bit devices										(Double) Word devices (W*, D*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S5	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S6	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S7	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S8	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
S9	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
D1	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–

* Only for station No. "S5," a double-word device is used.

Function

This FB reads data about devices on the network through the communication modules connected to the same SX bus.
It can read data about the following devices:
- Memory in SPH-series CPU modules through network (independent of types)
- Memory in MICREX-F/FLEX-PC CPU through T/P/PE links
- Memory of devices connected to the network of OPEN standard such as OPCN-1, FL-net

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Write request	Write processing will start at the rising edge of this signal. Though this signal is effective at its rising edge, it is ineffective at its rising edge during data write (until "D1" or "D2" goes ON after this signal goes ON).
S3	Communication module SX bus station No.	Specify the SX bus station No. of the communication module connected to the SX bus same as the CPU module.
S4	Communication module channel No.	Specify the channel No. in the communication module. For modules such as P/PE link modules with only one communication port, this is fixed to 0.
S5	Station No.	Specify the station No. of the communication partner on the network. For example, it is an IP address for Ethernet or P/PE link station No. for P/PE link.
S6	Device specification method	The method for specifying the write memory (device) depends on the access target of the communication partner. For details, see <Device specification method>.
S7	Write address	The write address specification depends on the device specification method. For details, see <Address specification format>.
S8	Write data size	Specify the number of words to be written, beginning at the address specified in "write address" (in word units).
S9	Write data storage device	Specify the heading address of the device where the write data is stored.
D1	Normal completion flag	When data write has completed normally, this flag remains ON for one scan.
D2	Error flag	When data write has terminated abnormally, this flag remains ON for one scan.
D3	Status	Indicates the write result. It goes to "0" at normal completion.

<Operations>

1) At the rising edge of write request “S2,” data is written to memory of network station “S5” through the communication port of the communication module specified in “S3” and “S4.”
2) The network station memory specification is determined by write address “S7” that matches the format specified in device specification method “S6.” For details, see page 3-147.
3) Writing is not completed in one scan. When writing has completed normally, normal completion flag “D1” remains ON for one scan. When writing has terminated abnormally, “D2” remains ON for one scan and the error cause is stored in status “D3.”

Note: 1) Specify write data size “S8” and write data “S9” (heading address) so that the write data is not overlapped with other data areas.

<Status>

Status	Description
35 (23h)	Transmission interlock Write processing was performed while the communication partner was being interlocked (program transfer, etc.). When this error has occurred, retry the communication.
68 (44h)	Memory address specification error The address specified in write address "S7" has an error.
69 (45h)	Memory size over The memory size exceeds the allowable address range of the partner CPU module specified in write address "S7" and write data size "S8." In this case, data may be written in the partner CPU module.
160 (A0h)	Communication partner specification error The communication partner is SPH (S6 = 0) and the CPU specified in write address "S7" does not exist.
162 (A2h)	No command response Receiving no response to command after a predetermined period of time
164 (A4h)	Message send error The communication partner is missing or no module exists for the specified SX station No.
165 (A5h)	Message receive busy The message communication partner is busy on the SX bus and message sending is disabled.
170 (AAh)	Message send busy The resource in the CPU for sending messages is busy and message send is disabled.
171 (ABh)	Internal resource exhausted The internal resource for executing R_READ or R_WRITE has been exhausted.
177 (B1h)	Parameter error - SX bus station No. is not 1 to FE. - Read data size "S8" is 0. - Device specification method "S6" is other than the specified value.
193 (C1h)	Channel open error The value specified in channel number "S4" is abnormal.
195 (C3h)	Message send error - The value specified in network station number "S5" is abnormal. - The value specified in channel number "S4" is abnormal. - The value specified in read address "S7" is abnormal.
197 (C5h)	Network send busy The communication module of the communication partner is busy and message send is disabled.
201 (C9h)	No empty ports An attempt was made to open ports exceeding the defined number in one communication module. READ_W/READ_B also use communication ports. For example, READ_W used for a device in the P link uses one communication port of the P link. There are five communication ports available for the P/PE link.
206 (CEh)	Buffer over The message data size limit value of the communication module used is exceeded.

<Device specification method>

The contents of device specification methods (“S6” and read/write address “S7”) for remote data read R_READ and remote data write R_WRITE instructions are defined for individual access targets of the communication partner.

S6	Variable Specification Method	Target of Access
0	SPH address	CPU memory in the SPH system via a network (of any type)
1	MICREX_F or FLEX-PC address	CPU memory in the MICREX or FLEX-PC via a T-link, P-link, or PE-link
2	Sequence of numerals	Device connected to an open standard network such as OPCN1
3	Character string	Device connected to an open standard network such as OPCN1

<Specification format of address “S7”>

● **When “S6” = 0**

F 0
CPU No.
Memory type *
Address, lower-order
Address, upper-order

● **When “S6” = 1**

F 0
File No.
Word address within file

● **When “S6” = 2**

F	0
Effective size n	
-	Address 1
-	…
-	…
-	Address n

In this case, the effective data is placed in the lower-order 8 bits of a 16-bit array. This is because an SPH system cannot handle 8-bit data.

● **When “S6” = 3**

F
ASCII character string The end of a string is represented by a null code.

For the “REMOTE_VAR” format, refer to the manuals for the communication module through which data is to be read or written and the device to be read from or written to.

* The memory type is the code for identifying standard memory, retain memory, etc., in user memory. It is the second column value in addressing (AT statement) any other CPU memory.

Name	Memory type code
Standard memory	1
Retain memory	3
User FB memory	5
System FB memory	9
System memory	10

Note: Do not specifiy 1, 3, 5, 9, or 10 for the memory type code.

(11) File data read (F_READ)

Symbol

F_READ		S1
I: V0001	REQ	S2
I: WV0002	MODULE_NO	S3
IO: _V0003	FILE_NAME	S4
I: WV0004	SIZE	S5
IO: _V0005	RD	S6
O: V0006	DONE	D1
O: V0007	ERROR	D2
O: WV0008	STATUS	D3
O: WV0009	F_SIZE	D4

S1: Instance No.
S2: Read request
S3: Target module SX bus station No.
S4: File name
S5: Read data storage size
S6: Read area heading address
D1: Normal completion flag
D2: Error flag
D3: Status
D4: Read file size

Available devices

	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	–	○	○	○
S4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S5	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	–	○	○	○
S6	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
D1	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D3	–	–	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	–	○	○	–
D4	–	–	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	–	○	○	–

Function

Use this FB to read data from the SRAM card of the memory card interface module in the same configuration (module connected to the same SX bus) or from the user ROM card of the high-performance CPU module (software version V59 or up).

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Read request	Read processing will start at the rising edge of this signal. Though this signal is effective at its rising edge, it is ineffective at its rising edge during file read (until "D1" or "D2" goes ON after this signal goes ON).
S3	Target module SX bus station No.	Specify the SX bus station No. of the module where the memory card to be read is mounted. For a user ROM card mounted on the local CPU, this is the SX bus station No. of the local CPU.
S4	File name	Specify the heading address of the device where the file name is stored. The file name must be specified in shift-JIS code (file name with 8 or fewer 1-byte characters plus extension with 3 or fewer 1-byte characters). Directories, if any, must also be specified. For the device storing the file name, reserve one additional word (for NULL code) for each character.
S5	Read data storage size	Specify the size in words of the device area that stores the read data.
S6	Read area heading address	Specify the heading address of the device area where the read data is stored. This is used in pair with "S5" to determine the read data storage area.
D1	Normal completion flag	When data read has completed normally, this flag remains ON for one scan.
D2	Error flag	When data read has terminated abnormally, this flag remains ON for one scan.
D3	Status	Indicates error code.
D4	Read file size	Indicates the size in words of the read file.

<Operations>

1) At the rising edge of read request “S2,” data is read from the file specified in “S4” on the memory card in the module specified in “S3” to the device (read data storage destination) specified in “S5” and “S6.”
2) Reading is not completed in one scan. When reading has completed normally, normal completion flag “D1” remains ON for one scan. When writing has terminated abnormally, error flag “D2” remains ON for one scan and the error cause is stored in status “D3.”

Notes: 1) Specify “S5” and “S6” so that the specified read data storage destination is not overlapped with other data areas (to prevent other devices from being overwritten). If they overlap, other data areas will be overwritten.
2) For file data read or write, do not use the memory card interface module that stores the N-to-1 redundant application program.

<Status>

Status	Description
35 (23h)	Transmission interlock error When a module mounting the memory card is accessed by a file data read/write instruction, that module is placed in transmission interlock state. In this case, this error will occur if another file data read/write instruction attempts to access the module mounting the memory card. When this error has occurred, retry the communications.
65 (41h)	File name error The file name specified in file name "S4" does not exist.
66 (42h)	Memory access error SUM check error occurred during file read.
162 (A2h)	No command response Receiving no response to command after a predetermined period of time
164 (A4h)	Message send error The communication partner is missing or no module exists for the specified SX station No.
165 (A5h)	Message receive busy The message communication partner is busy on the SX bus and message sending is disabled.
170 (AAh)	Message send busy The resource in the CPU for sending messages is busy and message send is disabled.
177 (B1h)	Parameter error - SX bus station number "S3" is not in the allowable range. - The read data storage destination specified in "S5" and "S6" exceeds the memory boundary.
197 (C5h)	Network send busy The communication module of the communication partner is busy and message send is disabled.
201 (C9h)	No empty ports An attempt was made to open ports exceeding the defined number in one communication module. F_READ/F_WRITE also use communication ports.

(12) File data write (F_WRITE)

Symbol

F_WRITE		S1
I: V0001	REQ	S2
I: WV0002	MODULE_NO	S3
IO: _V0003	FILE_NAME	S4
I: WV0004	SIZE	S5
IO: _V0005	WD	S6
O: V0006	DONE	D1
O: V0007	ERROR	D2
O: V0008	STATUS	D3

S1: Instance No.
S2: Write request
S3: Target module SX bus station No.
S4: File name
S5: Write data size
S6: Write area heading address
D1: Normal completion flag
D2: Error flag
D3: Status

Available devices

	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	–	○	○	○
S4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S5	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	–	○	○	○
S6	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
D1	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D3	–	–	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	–	○	○	–

Function

This FB writes data to the SRAM card in the memory card interface module in the same configuration (module connected to the same SX bus) or to the user ROM card in the high-performance CPU module.

Note: The function for writing data to user ROM is available for a device that allows user ROM mounting/demounting of high-performance CPU with software version V59 or up.

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Write request	Write processing will start at the rising edge of this signal. Note: Though this signal is effective at its rising edge, it is ineffective at its rising edge during file write (until "D1" or "D2" goes ON after this signal goes ON).
S3	Target module SX bus station No.	Specify the SX bus station No. of the module where the memory card to write data is mounted. For a user ROM card mounted on the local CPU, this is the SX bus station No. of the local CPU.
S4	File name	Specify the heading address of the device where the file name is stored. The file name must be specified in shift-JIS code (file name with 8 or fewer 1-byte characters plus extension with 3 or fewer 1-byte characters). Directories, if any, must also be specified. For the device storing the file name, reserve one additional word (for NULL code) for each character.
S5	Write data size	Specify the size in words of the device area that stores write data.
S6	Write area heading address	Specify the heading address of the device where the write data is stored.
D1	Normal completion flag	When data write has completed normally, this flag remains ON for one scan.
D2	Error flag	When data write has terminated abnormally, this flag remains ON for one scan.
D3	Status	Indicates the error code.

<Operations>

1) At the rising edge of write request “S2,” data in the device (write data storage destination) specified in “S5” and “S6” is written to the file specified in “S4” on the memory card in the module specified in “S3.”
2) Writing is not completed in one scan. When writing has completed normally, normal completion flag “D1” remains ON for one scan. When writing has terminated abnormally, error flag “D2” remains ON for one scan and the error cause is stored in status “D3.”

Note: 1) For file data read or write, do not use the memory card interface module that stores the N-to-1 redundant application program.

<Status>

Status	Description
35 (23h)	Transmission interlock error When a module mounting the memory card is accessed by a file data read/write instruction, that module is placed in transmission interlock state. In this case, this error will occur if another file data read/write instruction attempts to access the module mounting the memory card. When this error has occurred, retry the communications.
65 (41h)	File name error The directory name specified in file name "S4" does not exist. If the specified directory exists and no files exist, a new file will be created.
66 (42h)	Memory access error SUM check error occurred during file write.
69 (45h)	No empty areas The memory card has no empty area and writing is disabled.
162 (A2h)	No command response Receiving no response to command after a predetermined period of time
164 (A4h)	Message send error The communication partner is missing or no module exists for the specified SX station No.
165 (A5h)	Message receive busy The message communication partner is busy on the SX bus and message sending is disabled.
170 (AAh)	Message send busy The resource in the CPU for sending messages is busy and message send is disabled.
177 (B1h)	Parameter error - SX bus station number "S3" is not in the allowable range. - The write data storage destination specified in "S5" and "S6" exceeds the memory boundary.
201 (C9h)	No empty ports An attempt was made to open ports exceeding the defined number in one communication module. F_READ/F_WRITE also use communication ports.

(13) Extension test & set (EXT_T_S)

Symbol

EXT_T_S		S1
I: V0001	REQ	S2
IO: V0002	IN	S3
O: V0003	DONE	D1
O: V0004	Q	D2
O: V0005	ERROR	D3
O: WV0006	STATUS	D4

S1: Instance No.
S2: Test & set request
S3: Global bit device
D1: Normal completion flag
D2: Test & set result
D3: Error flag
D4: Status

Available devices

	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D1	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D3	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–

Function

This instruction acquires a semaphore. It is used when the bit device used for semaphore has been assigned to module memory on a different processor bus.

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Test & set request	Test & set processing will start at the rising edge of this signal. Though this signal is effective at its rising edge, it is ineffective at its rising edge during file read (until "D1" or "D2" goes ON after this signal goes ON).
S3	Global bit device	Specify the global bit device in another CPU in the same configuration.
D1	Normal completion flag	When test & set has completed normally, this flag remains ON for one scan.
D2	Test & set result	0: Set completed, 1: Set disabled (already set)
D3	Error flag	When an error has occurred in test & set, this flag remains ON for one scan.
D4	Status	Indicates the error code when an error has occurred in test & set. It goes to "0" when it has completed normally.

<Operations>

1) At the rising edge of test & set request “S2,” test & set for the global bit device specified in “S3” is performed. The result of test & set is output to “D2.”
   ON: Set completed (This goes ON (set) when the target bit device is OFF.)
   OFF: Set disabled (The target bit device is already ON (set).)
2) Test & set is not completed in one scan. When test & set has completed normally, normal completion flag “D1” remains ON for one scan. When writing has terminated abnormally, error flag “D3” remains ON for one scan and the error cause is stored in status “D4.”

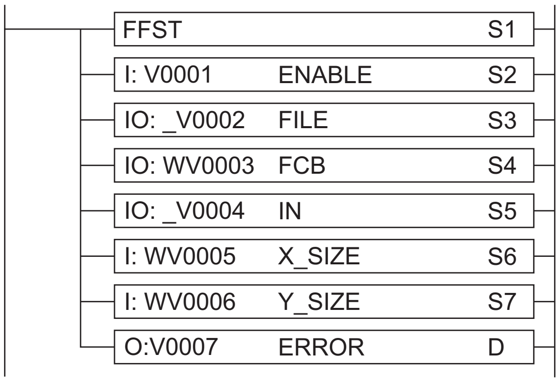
Notes: 1) For resetting the set data, use WRITE_B.
2) When the bit device used for the semaphore is in the local CPU or has been connected to the same processor bus, use T_S.
3) Do not use this instruction for memory in the P/PE link module or FL-net link module. If used, the FB will terminate abnormally (with code 32).
4) In order to ensure that the bit device used for the semaphore would pass the data when the operating and waiting CPUs are changed over in a redundant system, observe the following:
- Assign the bit device used for the semaphore to the standard memory area and do not set it as the target for equalization. If set, it will be cleared to 0 by changeover of CPUs.
- When a changeover of CPUs has occurred, perform semaphore acquisition processing again.

<Status>

Status	Description
32 (10h)	Unsupported instruction detection This instruction has been used for memory in the P/PE link module or FLI-net module.
170 (AAh)	Global bit device specification error The device specified in "S3" is memory in the local CPU.

(14) Sequential file store (FFST)

Symbol



S1: Instance No.
S2: File store execution
S3: File area heading address
S4: File control block
S5: Store data
S6: X size
S7: Y size
D: Error flag

Available devices

	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S5	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S6	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	○	○	○	○
S7	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	○	○	○	○
D	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–

Function

This instruction stores data in the specified device area.

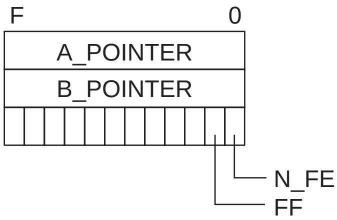
<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	File store execution	When this signal is set ON, file store starts.
S3	File area heading address	Specify the file area heading address. The file size is determined by "S6" and "S7."
S4	File control block	As a file pointer for file store or a work area for file-related flags, three words beginning at the address specified here are used.
S5	Store data	Specify the heading address of the device that stores data.
S6	X size	In this register, specify the number of words of data to be stored.
S7	Y size	The size of the area to store data is determined by (X size) x (Y size) in words.
D	Error flag	When file store has terminated abnormally, this flag remains ON for one scan.

<Operations>

1) When file store execution “S2” is set ON, the data block determined by store data “S5” and X size “S6” is written to the sequential file area determined by file area heading address “S3” and X size “S6” and Y size “S7.”
2) If the file is already full, old data is discarded and the data specified by “S5” and “S6” is written to the area beginning with the pointer.

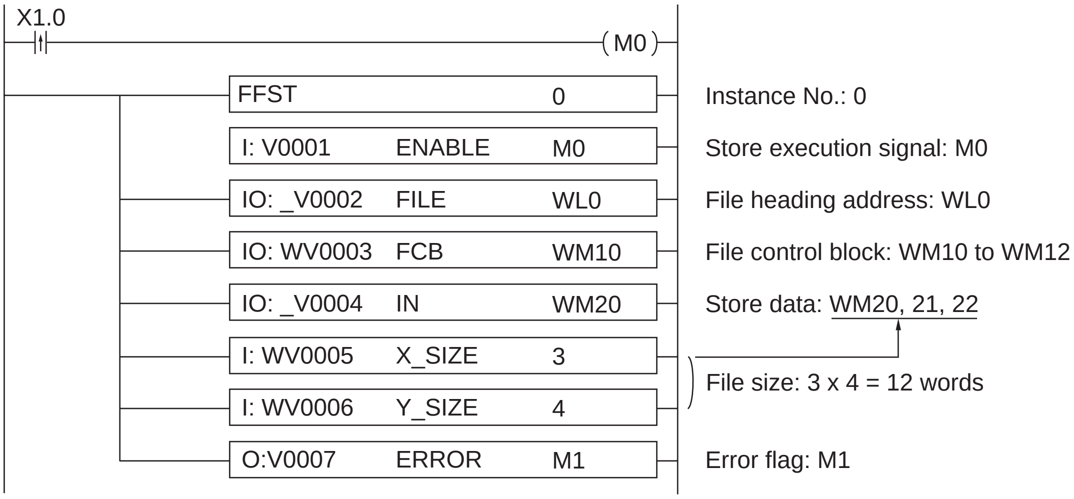
<File control block>



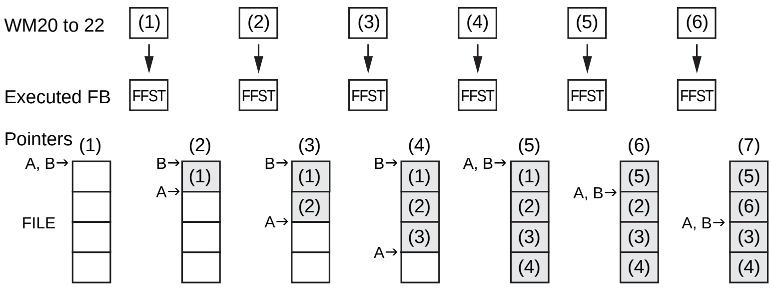
- A_POINTER : FFST write pointer
  : FILO read pointer
- B_POINTER : FIFO read pointer
- N_FE : When 0, it indicates that the file is empty.
- FF : When 1, it indicates that the file is full.

Program examples

At the rising edge of input X1.0, this instruction stores data stored in WM20 to WM22 sequentially to the 12-word file area of WL0 to WL11.



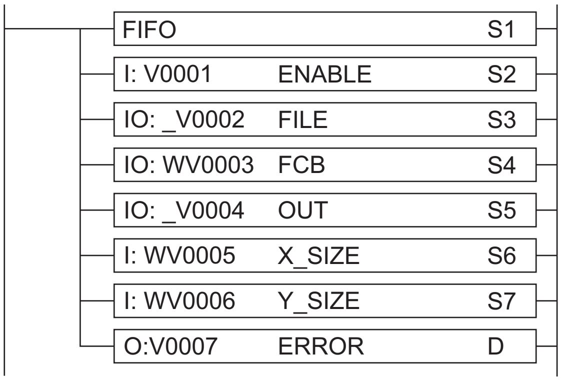
<Operation>



	(1)	(2)	(3)	(4)	(5)	(6)	(7)
A_PNT	0	1	2	3	0	1	2
B_PNT	0	0	0	0	0	1	2
FF	0	0	0	0	1	1	1
N_FE	0	1	1	1	1	1	1

(15) Sequential file load first (FIFO)

Symbol



S1: Instance No.
S2: File read execution
S3: File area heading address
S4: File control block
S5: Read data storage device
S6: X size
S7: Y size
D: Error flag

Available devices

	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S5	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S6	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	○	○	○	○
S7	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	○	○	○	○
D	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–

Function

This instruction reads data stored by FFST to the specified device, sequentially from the oldest data.

<Parameter description>

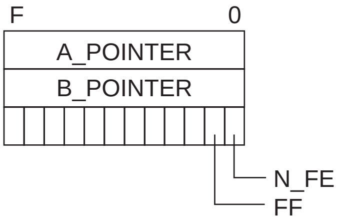
Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	File read execution	When this signal goes ON, data reading from the specified file starts.
S3	File area heading address	Specify the file area heading address. The file size is determined by "S6" and "S7." Here, specify the same device as the one specified in "S3" of FFST of the pair.
S4	File control block	As a file pointer for file store or a work area for file-related flags, three words beginning at the address specified here are used. Here, specify the same device as the one specified in "S4" of FFST of the pair.
S5	Read data storage device	Specify the heading address of the device that stores the read data.
S6	X size	In this register, specify the number of words of data to be read. Here, specify the same device as the one specified in "S6" of FFST of the pair.
S7	Y size	The size of the area to store data is determined by (X size) x (Y size) in words. Here, specify the same device as the one specified in "S7" of FFST of the pair.
D	Error flag	When file read has terminated abnormally, this flag remains ON for one scan.

<Operations>

1) When file read “S2” is set ON, the oldest data is read from the sequential file area determined by file area heading address “S3” and X size “S6” and Y size “S7” to the area of X size “S6” beginning with read data storage device “S5.”
2) When one read operation is performed, “B_POINTER” increments by 1. If the FF flag (file full) is ON at this time, the FF flag goes OFF.
3) When “B_POINTER” has reached “A_POINTER,” the “N_FE” flag goes OFF (indicating that the file is empty).

Note: When this instruction is executed for an empty file, it performs nothing.

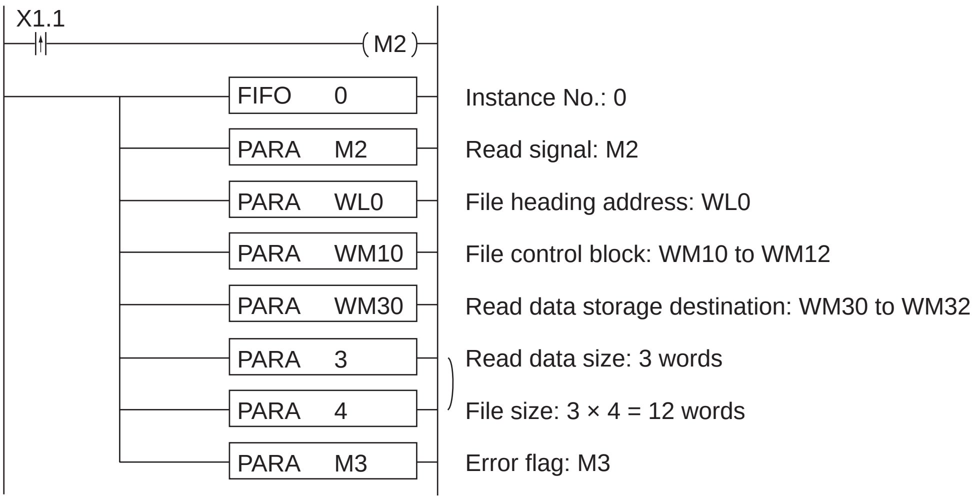
<File control block>



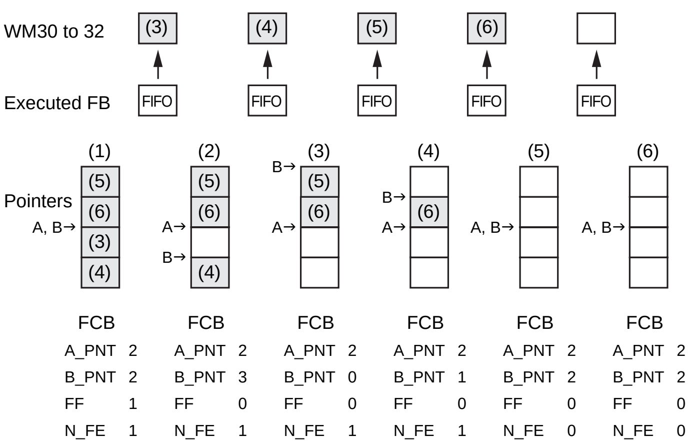
- A_POINTER : FFST write pointer
  : FILO read pointer
- B_POINTER : FIFO read pointer
- N_FE : When 0, it indicates that the file is empty.
- FF : When 1, it indicates that the file is full.

Program examples

At the rising edge of input X1.1, this instruction reads data from the file area (WL0 to WL11) to WM30 to WM32, sequentially beginning with the oldest data.

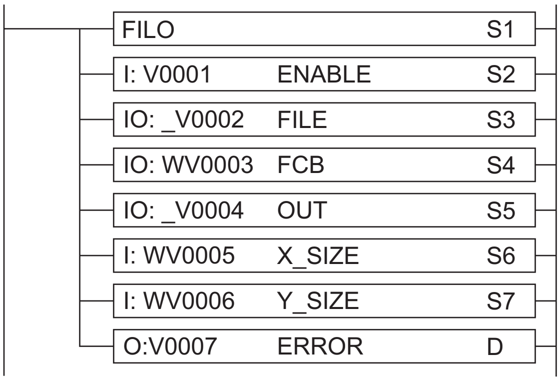


<Operation>



(16) Sequential file load last (FILO)

Symbol



S1: Instance No.
S2: File read execution
S3: File area heading address
S4: File control block
S5: Read data storage device
S6: X size
S7: Y size
D: Error flag

Available devices

	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S5	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
S6	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	○	○	○	○
S7	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	○	○	○	○
D	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–

Function

This instruction reads data stored by FFST to the specified device, sequentially from the newest data.

<Parameter description>

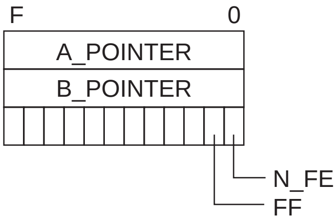
Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	File read execution	When this signal goes ON, data reading from the specified file starts.
S3	File area heading address	Specify the file area heading address. The file size is determined by "S6" and "S7." Here, specify the same device as the one specified in "S3" of FFST of the pair.
S4	File control block	As a file pointer for file store or a work area for file-related flags, three words beginning at the address specified here are used. Here, specify the same device as the one specified in "S4" of FFST of the pair.
S5	Read data storage device	Specify the heading address of the device that stores the read data.
S6	X size	In this register, specify the number of words of data to be read. Here, specify the same device as the one specified in "S6" of FFST of the pair.
S7	Y size	The size of the area to store data is determined by (X size) x (Y size) in words. Here, specify the same device as the one specified in "S7" of FFST of the pair.
D	Error flag	When file read has terminated abnormally, this flag remains ON for one scan.

<Operations>

1) When file read “S2” is set ON, the newest data is read from the sequential file area determined by file area heading address “S3” and X size “S6” and Y size “S7” to the area of X size “S6” beginning with read data storage device “S5.”
2) When one read operation is performed, “A_POINTER” decrements by 1. If the FF flag (file full) is ON at this time, the FF flag goes OFF.
3) When “B_POINTER” has reached “A_POINTER,” the “N_FE” flag goes OFF (indicating that the file is empty).

Note: When this instruction is executed for an empty file, it performs nothing.

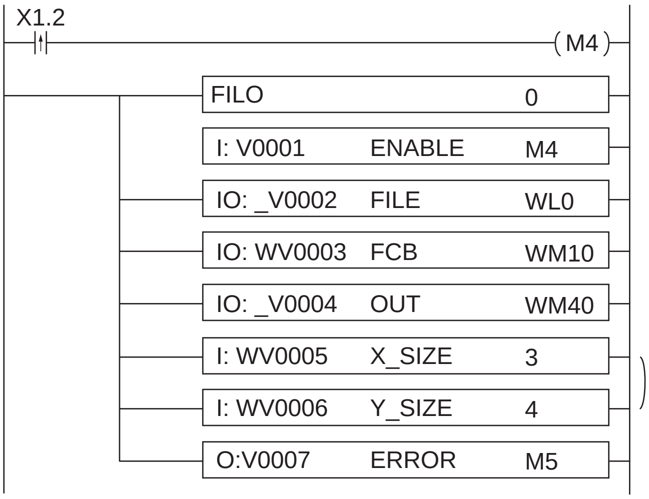
<File control block>



- A_POINTER : FFST write pointer
  : FILO read pointer
- B_POINTER : FIFO read pointer
- N_FE : When 0, it indicates that the file is empty.
- FF : When 1, it indicates that the file is full.

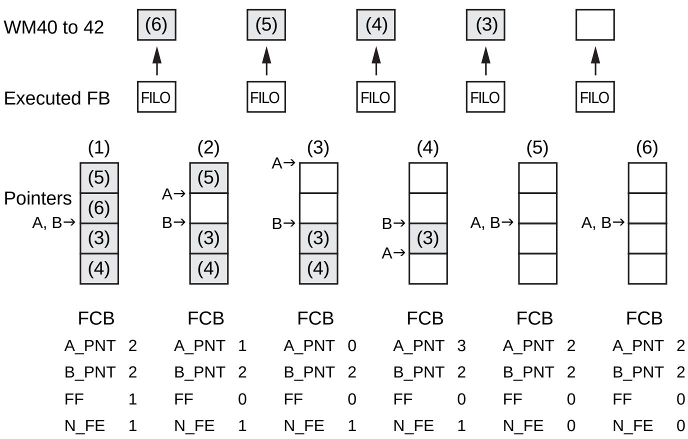
Program examples

At the rising edge of input X1.2, this instruction reads data from the file area (WL0 to WL11) to WM40 to WM42, sequentially beginning with the newest data.



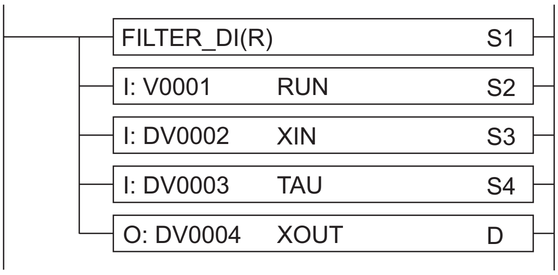
Instance No.: 0
Read signal: M4
File heading address: WL0
File control block: WM10 to WM12
Read data storage destination: WM40 to WM42
Read data size: 3 words
File size: 3 × 4 = 12 words
Error flag: M5

<Operation>



(17) Filter (FILTER_DI, FILTER_R)

Symbol



S1: Instance No.
S2: Filter enable
S3: Input signal (to be filtered)
S4: Filter time constant
D: Filter output

Available devices

	Bit devices										Double word devices (D*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	–	○	○	–
S4	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
D	–	–	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	–	○	○	–

Function

This instruction filters an input signal to be output.

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Filter enable	When this signal is ON, this instruction filters input signal "S3" and outputs it to "D."
S3	Input signal (to be filtered)	Specify the device that stores the input signal to be filtered. FILTER_DI => Specify a double-word device (double-precision integer). FILTER_R => Specify a double-word device (real).
S4	Filter time constant	Specify a time constant of up to 86400000 ms (24 hours). The larger the specified time constant is, the slower the curve of output "D" becomes. When a small time constant is specified, error in "D" increases. Specify a filter time constant which is larger than the execution cycle of this instruction.
D	Filter output	Filter instruction output

<Operations>

1) When filter enable “S2” is ON, input signal “S3” is filtered and output to “D.” When “S2” is OFF, input signal “S3” is directly output to “D.”

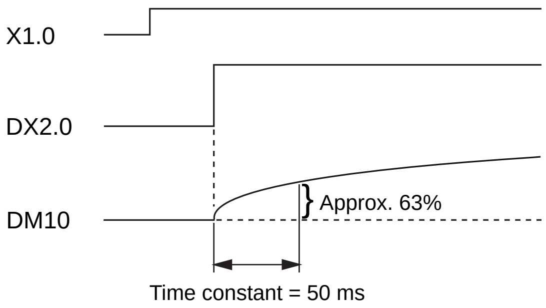
Notes: 1) Input such value that would not cause overflows during operation or in the operation result. If an overflow has occurred, the value of "D" is not guaranteed.
2) Use this instruction on POU assigned to the fixed task. For a standard CPU, the period of the fixed cycle should be 2 ms or more.

Program examples

When input X1.0 is ON, this instruction filters the value of input signal DX2.0 and outputs the result to DM10.

FILTER_DI		0	Instance No.: 0
I: V0001	RUN	X1.0	Filter enable: Enabled when DX1.0 is ON.
I: DV0002	XIN	DX2.0	Input signal: DX2.0
I: DV0003	TAU	50ms	Filter time constant: 50 ms
O: DV0004	XOUT	DM10	Filter output: DM10

<Operation>



<Formulas>

(FILTER_DI)

$$D=\frac{(S3-D')\triangle T+WORK'}{S4}+D'$$

WORK is a remainder of $\frac{(S3-D')\triangle T+WORK'}{S4}$

(FILTER_R)

$$D=\frac{(S3-D')\triangle T}{S4}+D'$$

D' and WORK' are previous values.
$\triangle$T is the execution cycle of this FB.

(18) Integrate (INT_DI, INT_R)

Symbol

INT_DI(R)		S1
I: V0001	RUN	S2
I: V0002	R1	S3
I: DV0003	XIN	S4
I: DV0004	XO	S5
I: DV0005	I_T	S6
O: V0006	XOUT	D1
O: V0007	Q	D2

S1: Instance No.
S2: Integration execution
S3: Preset execution
S4: Input value
S5: Initial value
S6: Integration time
D1: Integration value
D2: Preset execution invert flag

Available devices

	Bit devices										Double word devices (D*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S4	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	○	○	–	○	○	–
S5	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	○	○	–	○	○	○
S6	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	○	○	–	○	○	○
D1	–	–	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	–	○	○	–
D2	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–

Function

This instruction integrates the input data with the set integration time and outputs the result.

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Integration execution	When this signal is ON, input value "S4" is integrated and the result is output to integration value "D1." When this signal is OFF, the integration value is held.
S3	Preset execution	When this signal goes ON, the value of initial value "D5" is output to "D1."
S4	Input value	Specify the input data to be integrated.
S5	Initial value	Specify the initial output value.
S6	Integration time	Specify the integration time. The maximum allowable value is 86400000 ms (24 hours). Specify a integration time which is larger than the execution cycle of this instruction.
D1	Integration value	The result of integration of "S4" is output.
D2	Preset execution invert flag	Logically inverted value of preset execution "S3" is output.

<Operations>

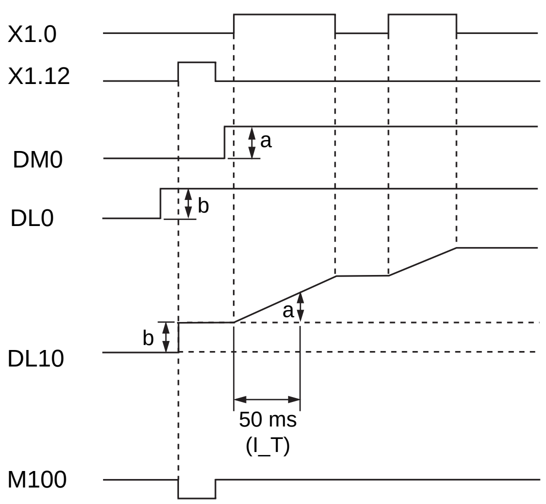
1) When integration execution “S2” is ON, data input to “S4” is integrated and the result is output to “D1.” When “S2” is OFF, value of “D1” is held.
   A linear value, like the one given to “S4,” is calculated when integration time “S6” has elapsed and the sum of that linear value and the initial value in “S5” is output to “D1.”
2) When preset execution “S3” goes ON, the initial value in “S5” is output to “D1.”
3) At the timing “S2” goes ON, the previous value is output to “D1.”

Notes: 1) Input such value that would not cause overflows during operation or in the operation result. If an overflow has occurred, the integration value is not guaranteed.
2) Use this instruction on POU assigned to the fixed task. For a standard CPU, the interval of the fixed cycle should be 2 ms or more.

Program examples

INT_DI		0	Instance No.: 0
I: V0001	RUN	X1.0	Integration execution: X1.0
I: V0002	R1	X1.1	Preset execution:X1.1
I: DV0003	XIN	DM0	Input value:DM0
I: DV0004	XO	DL0	Initial value:DL0
I: DV0005	I_T	50ms	Integration time: 50 ms
O: V0006	XOUT	DL10	Integration value: DL10
O: V0007	Q	M100	Preset execution invert flag: M100

<Operation>



<Formulas>

(INT_DI)

$$D1=\frac{(S4\triangle T+WORK')}{S6}+D1'$$

WORK is a remainder of $\frac{S4\triangle T+WORK'}{S6}$

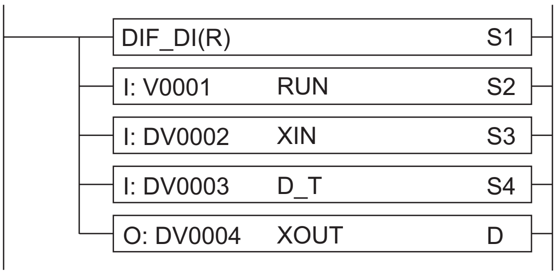
(INT_R)

$$D1=\frac{S4\triangle T}{S6}+D1'$$

D1’ and WORK’ are previous values.
$\triangle$T is the execution cycle of this FB.

(19) Differentiate (DIF_DI, DIF_R)

Symbol



S1: Instance No.
S2: Differentiation execution
S3: Input value
S4: Differentiation time
D: Differentiation value

Available devices

	Bit devices										Double word devices (D*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	○	○	–	○	○	–
S4	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	○	○	–	○	○	○
D	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–

Function

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Differentiation execution	When this signal is ON, the differentiation value is obtained from input value "S3" and previous input value and output to "D1." When this signal is OFF, 0 is output.
S3	Input value	Specify the input data to be differentiated.
S4	Differentiation time	Specify the differentiation time. The maximum allowable value is 86400000 ms (24 hours). Specify a differentiation time which is larger than the execution cycle of this instruction.
D	Differentiation value	The differentiated value is output.

<Operations>

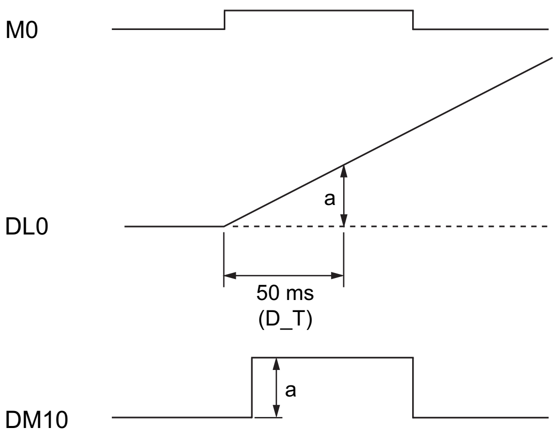
1) When differentiation execution “S2” is ON, the variation is calculated from the current and previous values “S3” and the variation after elapsing the differentiation time is anticipated and output to “D.”
   When “S2” is OFF, output “D” goes to 0.
2) Output to “D” is obtained based on the variation after “S2” was set ON. Therefore, at the time “S2” is set ON, 0 is output to “D.”
3) The number of significant digits in the output is 6.

Notes: 1) Input such value that would not cause overflows during operation. If an overflow has occurred, the integration value is not guaranteed.
2) Use this instruction on POU assigned to the fixed task. Note that error will increase for a short fixed cycle. For a standard CPU, the interval of the fixed cycle should be 2 ms or more.

Program examples

DIF_DI		0	Instance No.: 0
I: V0001	RUN	M0	Differentiation execution: M0
I: DV0002	XIN	DL0	Input value: DL0
I: DV0003	D_T	50ms	Differentiation time: 50 ms
O: DV0004	XOUT	DM10	Differentiation value: DM10

<Operation>



<Formulas>

$$D=\frac{(S3-XM)S4}{\triangle T}$$

XM is the previous value of S3.
$\triangle$T is the execution cycle of this FB.

(20) Pulse count (PULSE_CNT)

Symbol

PULSE_CNT		S1
I: V0001	RUN	S2
I: V0002	IN	S3
I: DV0003	DURATION	S4
O: WV0004	OUT	D1
O: V0005	CMP	D2

S1: Instance No.
S2: Pulse count enable
S3: Pulse input
S4: Count time
D1: Count value
D2: Count completion

Available devices

	Bit devices										(Double) Word devices (W*, D*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S4	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	○	○	–	○	○	○
D1	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–
D2	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–

Function

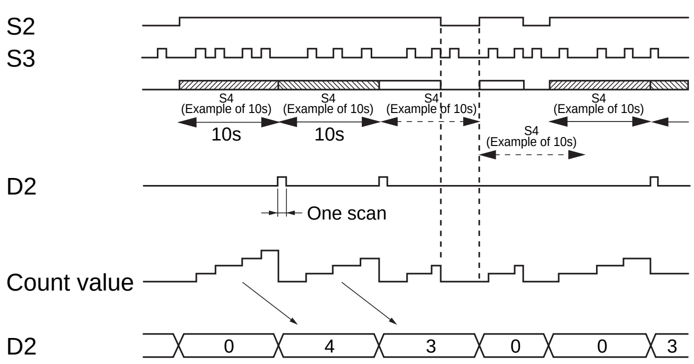
This instruction counts the pulses input in the specified period of time.

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Pulse count enable	When this signal goes ON, the pulses input to "S3" during the time (period) specified in "S4" are counted. When this signal is OFF, the current value in "D1" is held.
S3	Pulse input	Specify the device to which pulses are input.
S4	Count time	Specify the period for counting pulses. The allowable range is 0 to 4294967295 (ms).
D1	Count value	The count value is output. The range is -32768 to 32767.
D2	Count completion	When the time specified in "S4" has elapsed, this signal remains ON for one scan.

<Operations>

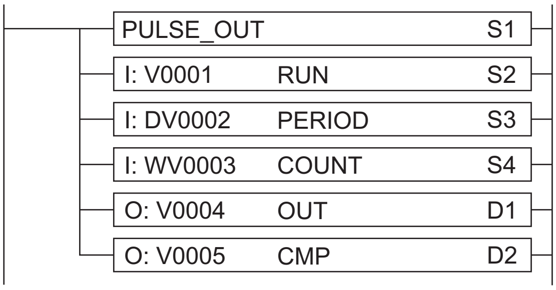
1) When pulse count enable “S2” goes ON, the pulses input to “S3” during the period of time specified in “S4” are counted and the count value is output to “D1.” When the time specified in “S4” has elapsed, “D2” remains ON for one scan. If “S2” is ON, “S1” is cleared and counting occurs again.
2) If “S4” is 0 ms, “D2” remains ON while “S2” is ON.



Notes: 1) Because timing processing for “S4” is performed likewise a timer, an error of +0 to +2 scans will occur.
2) The pulses input to “S3” must have a width two times or more the instruction execution cycle.

(21) Pulse output (PULSE_OUT)

Symbol



S1: Instance No.
S2: Pulse count enable
S3: Pulse output cycle
S4: Output pulse count
D1: Pulse output
D2: Output completion

Available devices

	Bit devices										(Double) Word devices (W*, D*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	–	○	○	○
S4	–	–	–	–	–	–	–	–	–	–	○	–	○	○	–	–	–	○	○	○	○
D1	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–

Function

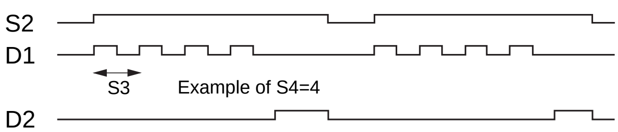
This instruction outputs pulses of the specified number.

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Pulse count enable	When this signal goes ON, pulses of the number specified in "S4" and with the width specified in "S3" are output.
S3	Pulse output cycle	Specify the cycle of output pulses. The duty ratio is 50:50. For example, when outputting pulses with width of 1 second, specify 2 seconds (2000 ms).
S4	Output pulse count	Specify the number of pulses to be output. The allowable range is -32768 to 32767.
D1	Pulse output	Specify the device to which the pulses are output.
D2	Output completion	This signal goes ON upon completion of pulse output. When "S2" is set OFF, this signal also goes OFF.

<Operations>

1) When pulse output “S2” goes ON, the pulses of the number specified in “S4” and with the width specified in “S3” are output to “D1.” When the output has completed, output completion “D2” goes ON. When “S2” is set OFF, “D2” goes OFF.
2) When “S2” is set OFF during pulse outputting, the pulse output stops.
3) When “S4” <= 0, pulses are output while “S2” is ON.



Notes: 1) For the “D1” cycle, an error of +0 to +2 scans will occur for “S3.”
2) Specify a time in “S3” which is sufficiently longer than the instruction execution cycle (two times or more).

(22) Modulate pulse width (PWM)

Symbol

```
PWM                      S1
I: V0001     RUN         S2
I: DV0002    PERIOD      S3
I: DV0003    WIDTH       S4
O: V0004     OUT         D
```

S1: Instance No.
S2: Pulse count enable
S3: Pulse output cycle
S4: Pulse width
D: Pulse output

Available devices

	Bit devices										(Double) Word devices (W*, D*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	○	○	–	○	–	–	–	–	○	○	○
S4	–	–	–	–	–	–	–	–	–	–	○	○	–	○	–	–	–	–	○	○	○
D	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–

Function

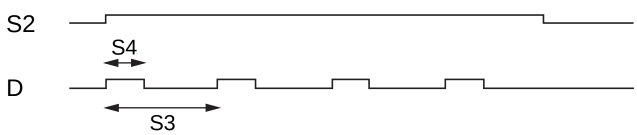
This instruction outputs pulses with the specified pulse width.

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Pulse count enable	When this signal goes ON, pulses are output.
S3	Pulse output cycle	Specify the cycle of output pulses.
S4	Pulse width	Specify the pulse-ON time (width).
D	Pulse output	Specify the device to which the pulses are output.

<Operations>

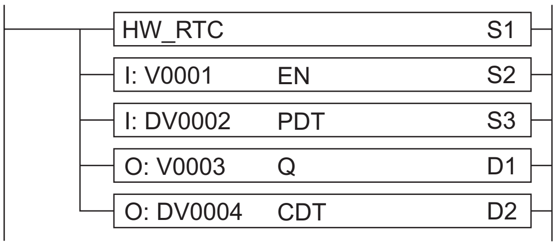
1) When pulse output “S2” goes ON, pulses with the cycle specified in “S3” and ON width specified in “S4” are output. When “S2” is set OFF, the pulse output stops.
2) Specify the pulse output cycle and pulse width so that **S3 - S4 > instruction execution cycle**.



Notes: 1) Because timing processing for “S4” is performed likewise Timer instruction, an error of +0 to +2 scans will occur for the “D” cycle and ON width.
2) When specifications are made in such a way that “S3” <= “S4” or “S4” = 0 is met, no pulses will be output.

(23) Hardware RTC (HW_RTC)

Symbol



S1: Instance No.
S2: Calendar value rewrite command
S3: Set value
D1: Rewrite completion flag
D2: Calendar value output

Available devices

	Bit devices										Double word devices (D*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
D1	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	–

Function

This instruction rewrites/reads the calendar IC in the CPU module.

<Parameter description>

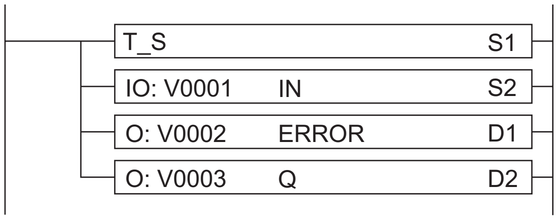
Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Calendar value rewrite command	At the rising edge of this signal, the calendar value in "S3" is written to the calendar IC in the CPU module.
S3	Set value	Set the calendar data for rewriting. Allowable set range: 1970-01-01-00:00:01 to 2069-12-31-23:59:59
D1	Rewrite completion flag	This flag goes ON when rewriting the calendar value in the calendar IC in the CPU module has completed.
D2	Calendar value output	The current calendar value in the calendar IC in the CPU module is output.

<Operations>

1) At the rising edge of calendar rewrite command “S2,” the value of set value “S3” is written to the calendar IC in the CPU module. When “S3” is 0 (1970-01-01-00:00:00), the current calendar value is written to “D2.”
2) “D1” goes ON when “S2” is ON and the value of “S3” has been written to the calendar IC. Note that if “S3” is 0, “D1” goes ON immediately after “S2” goes ON.
3) When “S2” goes OFF, “D2” is not updated. “D1” goes OFF.

(24) Test & set (T_S)

Symbol



S1: Instance No.
S2: Semaphore bit
D1: Error flag
D2: Test & set result

Available devices

	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D1	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–

Function

This instruction acquires a semaphore in memory in one CPU module. It is used when sharing some resources (memory) by multiple tasks, etc.

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Semaphore bit	This instruction, when executed, acquires a semaphore using the device specified here as a semaphore bit.
D1	Error flag	When this instruction is not executed normally, this flag remains ON for one scan.
D2	Test & set result	Indicates the execution result of this instruction. OFF: Set completed, ON: Set disabled (already set)

<Operations>

1)This instruction sets the device specified in "S2" and outputs the result to “D2.”

Note: In order to ensure that the bit device used for the semaphore would pass the data when a changeover of operating and waiting CPUs has occurred in redundant system, observe the following:
- Do not set the bit device used for the semaphore as the target for equalization. (If set, it will be cleared to 0 by changeover of CPUs.)
- When a changeover of CPUs has occurred, perform semaphore acquisition processing again.

(25) Change bank (BANK_CHG)

Symbol

BANK_CHG		S1
I: V0001	REQ	S2
I: WV0002	MODULE_NO	S3
O: V0003	DONE	D1
O: V0004	ERROR	D2
O: WV0005	STATUS	D3
O: V0006	BUSY	D4

S1: Instance No.
S2: Bank change request
S3: Target SX bus station No.
D1: Normal completion flag
D2: Error flag
D3: Status
D4: Busy

Available devices

	Bit devices										Word devices (W*)										Constants
	X	Y	M	L	SM	T	TR	C	F	V	X	Y	M	L	SM	T	TR	C	F	V	
S1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	○
S2	○	○	○	○	○	○	○	○	○	○	–	–	–	–	–	–	–	–	–	–	–
S3	–	–	–	–	–	–	–	–	–	–	–	–	○	○	–	–	–	–	○	○	○
D1	–	○	○	○	–	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D2	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–
D3	–	–	–	–	–	–	–	–	–	–	–	○	○	○	–	–	–	–	○	○	–
D4	–	○	○	○	○	–	–	–	○	○	–	–	–	–	–	–	–	–	–	–	–

Function

This instruction is used to reserve the uniqueness of data in processor link modules such as P/PE link and FL-net modules.

<Parameter description>

Parameter	Name	Description
S1	Instance No.	Specify the instance No. The allowable range is 0 to 65535.
S2	Bank change request	At the rising edge of this signal, a bank change request is issued to the processor link module specified in "S3."
S3	Target SX bus station No.	Specify the SX bus station No. of the processor link module to be bank-changed. The SX bus station No. is 246 for CPU No. 8 and 245 for CPU No. 9.
D1	Normal completion flag	When bank change has completed normally, this flag remains ON for one scan.
D2	Error flag	When there is an error in bank change, this flag remains ON for one scan.
D3	Status	When there is an error in bank change, the error cause is set. When bank change has completed normally, the status goes to "0."
D4	Busy	This goes ON during bank change processing.

<Operations>

1) At the rising edge of bank change request “S2,” the processor module specified in “S3” is requested for a bank change. When bank change is requested, busy “D4” goes ON; it goes OFF when bank change has completed. When accessing the broadcast communication area fron an application program, data can be synchronized by starting read/write after “D1” goes ON.

Notes: 1) Multiple bank change FBs for one CPU can be written. Note that issuing a change request to confirming change completion should be performed by one FB.
2) The CPU module and P/PE link module that specify bank change should be mounted to the same processor bus (on the same base). If this instruction is executed without connecting these modules to the same processor bus, an error occurs while executing the application program.

<Status>

Status	Name
64 (40h)	SX bus station No. error An SX bus station No. has been specified for a module other than the processor link modules.
65 (41h)	Multiple requests for bank change Two or more bank change requests have been issued from one CPU.
66 (42h)	Bank change processing processor bus error An error occurred during bank change processing.

Section 4 System Definition

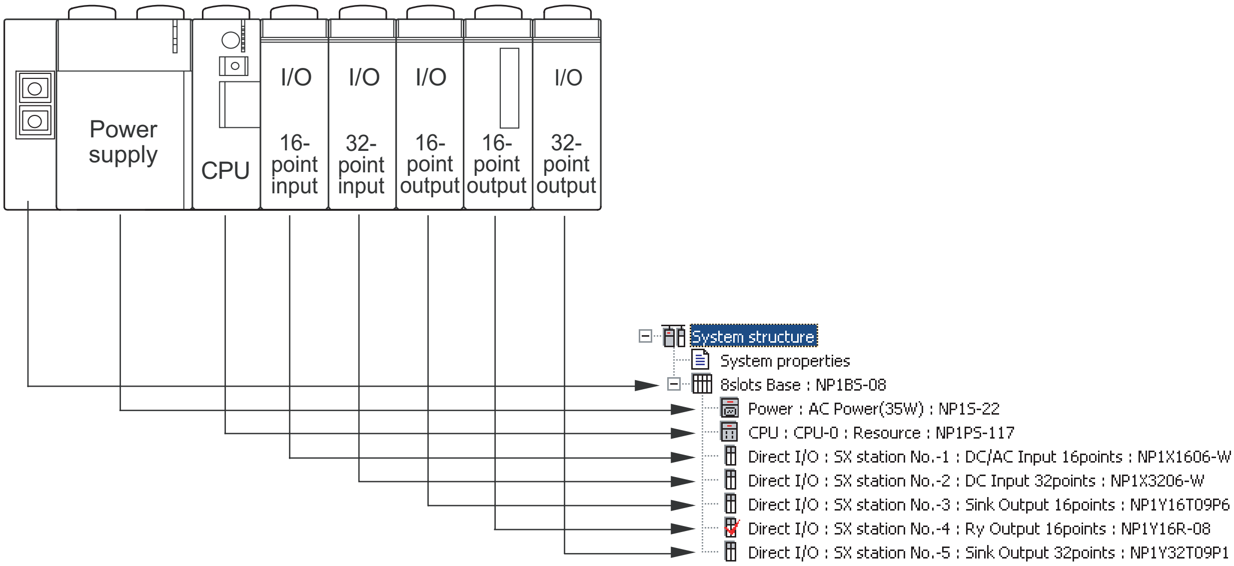
4-1 System Definition Summary

Name	Definition	Reference	Recognition Timing	
			High-performance CPU	Standard CPU
System configuration definition	Module registration and SX bus station number in the SPH system	4-2	SPH system reset Download	SPH system reset
System running definition	SX bus Takt period, configuration check waiting time, initialization method	4-3-1	SPH system reset	SPH system reset
Redundant mode definition	Redundant mode enabled/disabled, 1-to-1 redundant mode, N-to-1 redundant mode	4-3-2 Section 5	SPH system reset	Not supported
System fail-soft enabling definition	The fail-soft start-up start SX bus station number	4-3-3	SPH system reset	Not supported
CPU running definition	Watchdog timer, running specification at power on, battery-less run, constant scanning, execution band ratio, user ROM run, compulsion setting hold state	4-4-1	SPH system reset	SPH system reset
CPU memory size definition	Data memory size, AT specification range, reserve memory	4-4-2	SPH system reset	SPH system reset
I/O group	I/O group registration	4-4-3	SPH system reset	SPH system reset, Download
Fail-soft setting	Fail-soft enabled/disabled for modules (I/O module, etc) other than common modules	4-4-4	SPH system reset	SPH system reset
Input filtering time	Input filtering time for digital input modules (DC input devices)	4-5-1	Download	SPH system reset
Output hold definition	Hold/reset registration for output modules	4-5-2	Download	SPH system reset
System output definition	System output module	4-5-3	SPH system reset	SPH system reset
T-link master module parameter	Individual output hold station definition	4-6	SPH system reset	SPH system reset

Note: Reset includes all resets (configuration resets), individual reset (resource reset), and power-on.

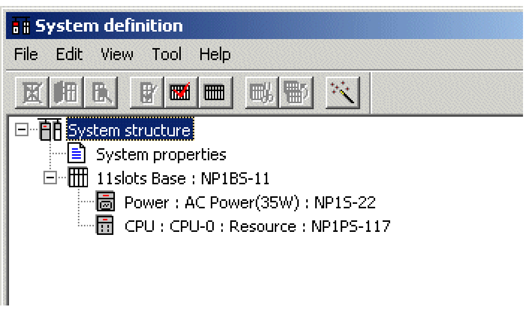
At system start-up, the CPU module of the MICREX-SX compares the system configuration information stored in the CPU module with the actual system configuration. After confirming that there is no mismatch between them, application processing is performed. Therefore, it is necessary to register all the modules to be used under one SPH system according to the actual configuration.
The following explains module registration procedures using the sample system configuration below.

<Sample system configuration>

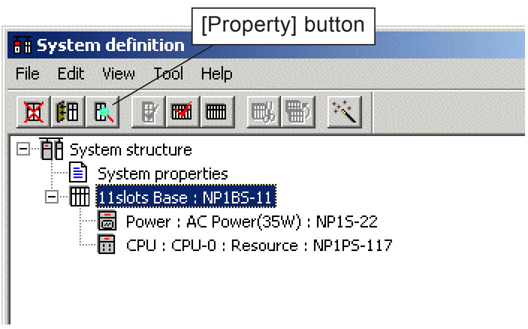


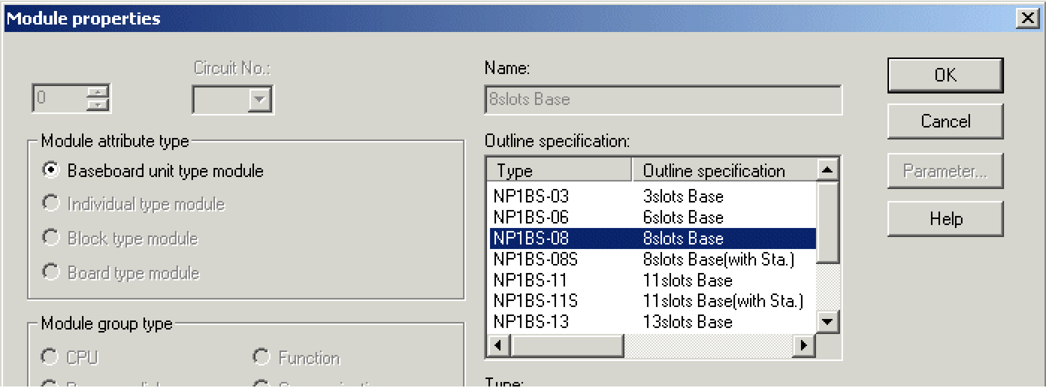
<Registration method>

◆ Execute the “PLC function” menu → [System Definition] in the project tree. The system definition screen appears.

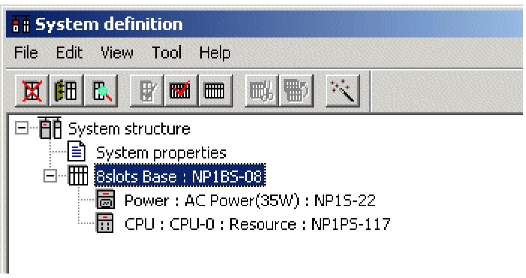


◆ Change the base board to 8-slot base to be used actually. Select a base board and then click the [Property] button. The [Module properties] dialog appears.

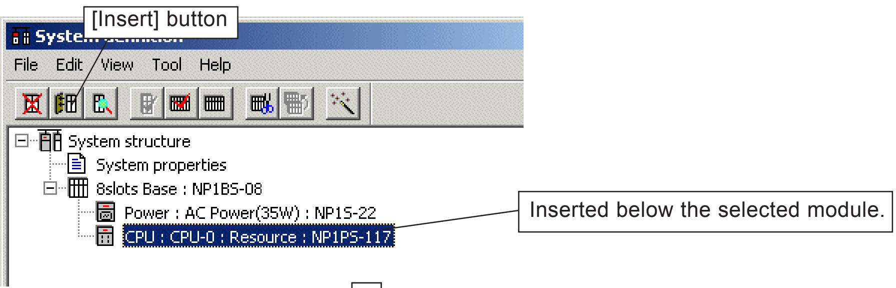


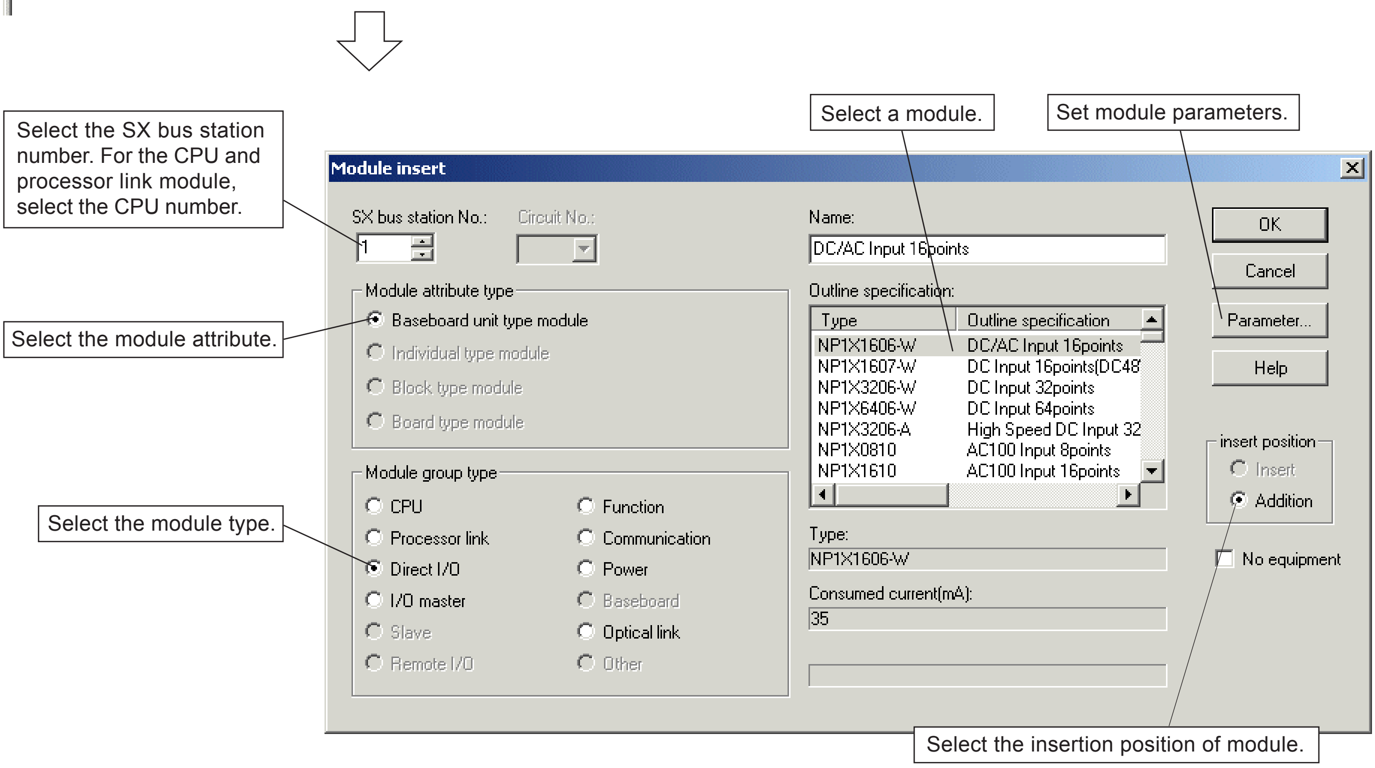


◆ Select "NP1BS-08 8slots Base" from the [Outline specification] list box and then click the [OK] button. The base board becomes the 8-slot base.

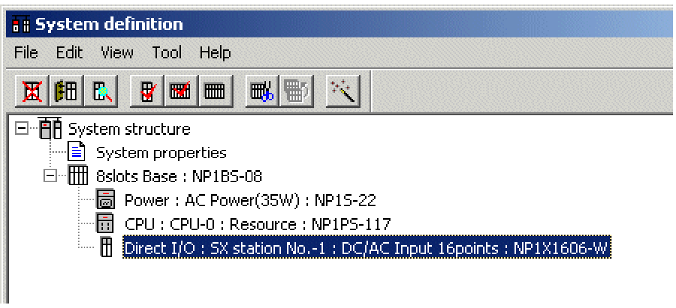


◆ Register module registration sequentially from the left of a base board. Since a power supply and a CPU module are registered by the default, register the 16-point input module. To register (add) the module below the CPU module, select the CPU module and then click the [Insert] button. The [Module insert] dialog appears.

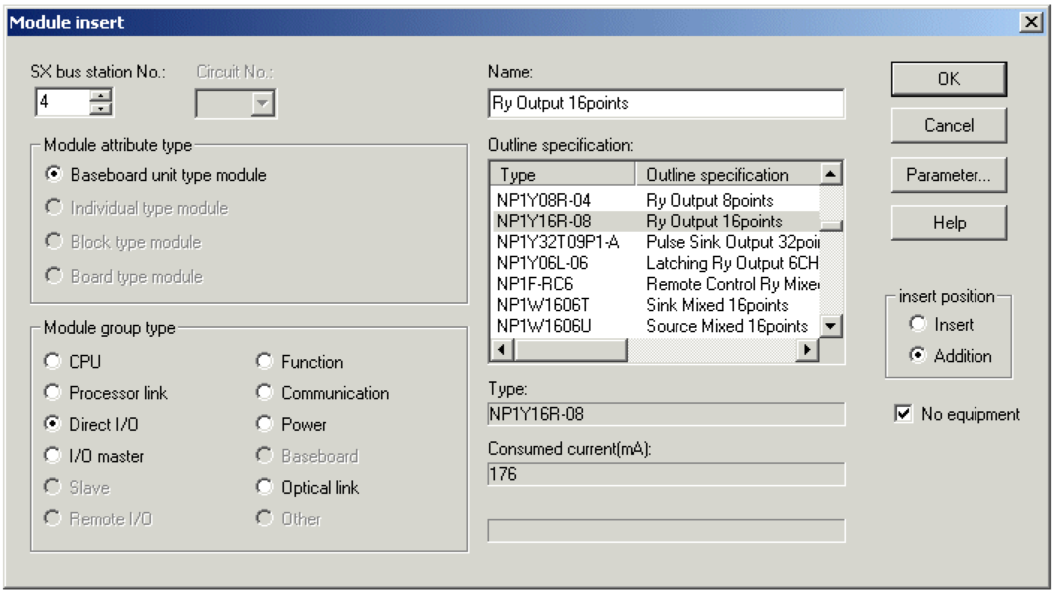


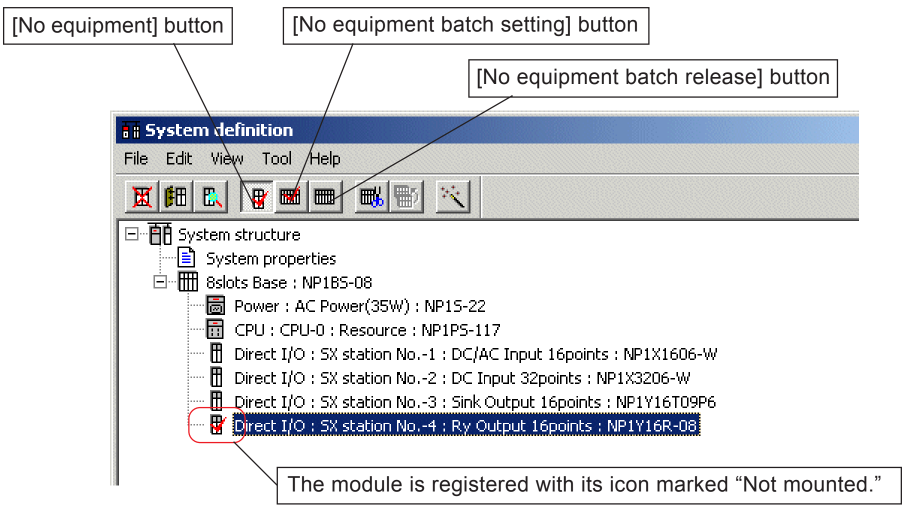


◆ Select the module (NP1X1606-W DC/AC Input 16points) according to the actual configuration. For SX bus station number, an arbitrary number from 01 to EF(h) can be specified. However, SX bus station numbers are usually assigned in order of 01, 02, 03... from the right side of the CPU module.
When you select a module and then click the [OK] button, the module is registered.



◆ Register modules in the same manner. If you want to mount a module in the future and leave the slot unused as shown in the system configuration, specify the SX bus station number and the module to be mounted in the same manner as other modules, check [No equipment], then click the [OK] button.





* With V2.2.0.0 or later version, no equipment registration of modules/units registered in the system definition can be collectively made or cancelled on the system definition screen. When collectively making/cancelling no equipment registration of the entire system, press the button with the “System structure” selected. Likewise, when collectively making/cancelling no equipment registration of modules/units on a base board, press the button with the base board selected. For units connected to a remote I/O master module, press the button with the remote I/O master selected.

4-3 System Properties

With the system property, the following three items are set: “System running definition”, “Redundancy setting” and “Fail-soft start-up operation setting”. How to set the system running definition and fail-soft start-up operation definition are explained below.

4-3-1 System running definitions

The system running definition includes the following four setting items: SX bus Takt time, configuration check waiting time, initialization method and start up system without “CPU0”.

1) SX bus Takt time

The SX bus Takt time is defined as the period at which data is exchanged between the modules (such as input/output modules) that are connected to the SX bus. The SX bus Takt time that can be set are determined by the CPU software version and CPU model, as shown in the table below. The default is 1 ms.

CPU software version	Setting range
SPH300 V34 and earlier and V3A to V3Z	0.5 ms, 1 ms, 2 ms, ..., 20 ms (steps of 1 ms)
Earlier than SPH300 V50	0.5 ms, 1 ms, 2 ms, ..., 10 ms (steps of 1 ms)
SPH300 V50 and later, SPH2000	0.5 ms, 1 ms, 1.5 ms, 2 ms, ..., 10 ms (steps of 0.5 ms)
SPH200	1 ms, 2 ms, ..., 20 ms (steps of 1 ms)

Note: A 0.5 ms Takt period may be executed in such conditions as having a single CPU in a high-performance CPU, 256 or less directly-connected I/Os, and no remote I/O and communication module.

2)Configuration check waiting time

At system power-on, the CPU module starts a configuration check for all the modules on the SX bus. When all the modules start up within the “waiting time for structure check”, the system starts operating then. Adjust the “waiting time for structure check”, for example, if each base is powered up at different timings.
(20 seconds is selected for the “waiting time for structure check” by default. The setting range: 1 to 180 seconds.)

3) Initialization method

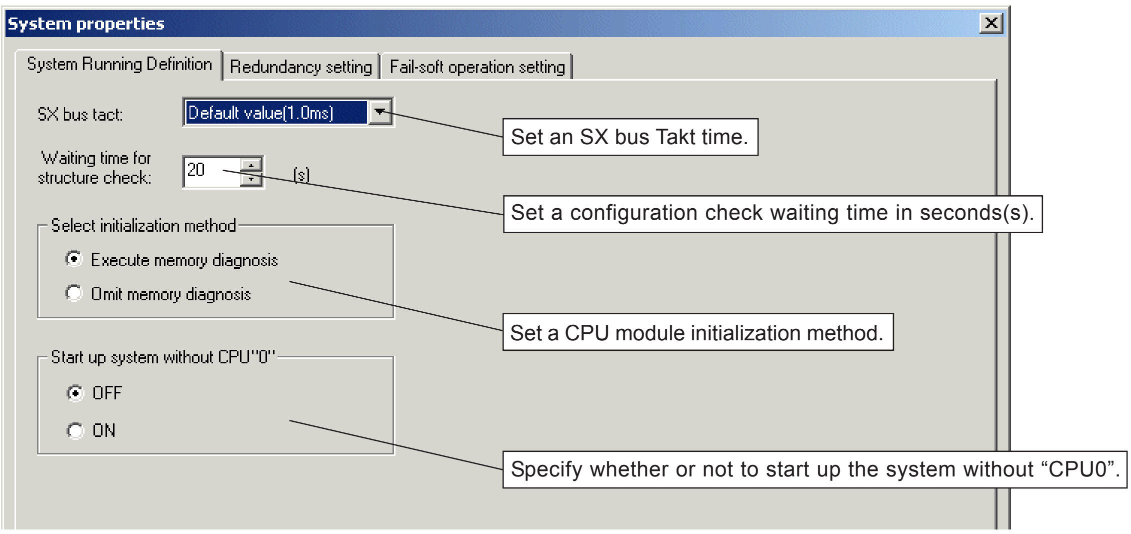
The CPU module initializes itself at system power-on. The user can specify whether the CPU is to perform diagnostics on the internal memory in the CPU.
* Performing memory diagnostics cause a delay in the system starting up for the system memory check time.

4) Start up system without “CPU0”

Specify whether or not to start up the system without “CPU0” in a redundant system, etc.

<Setup procedure>

◆ Select the system property from the system definition screen and click the [Property] button. Then, the [System Running Definition] screen for the [System properties] dialog appears.



◆ After setting all necessary items, click the [OK] button.

4-3-2 System redundancy definition (not supported by standard CPU)

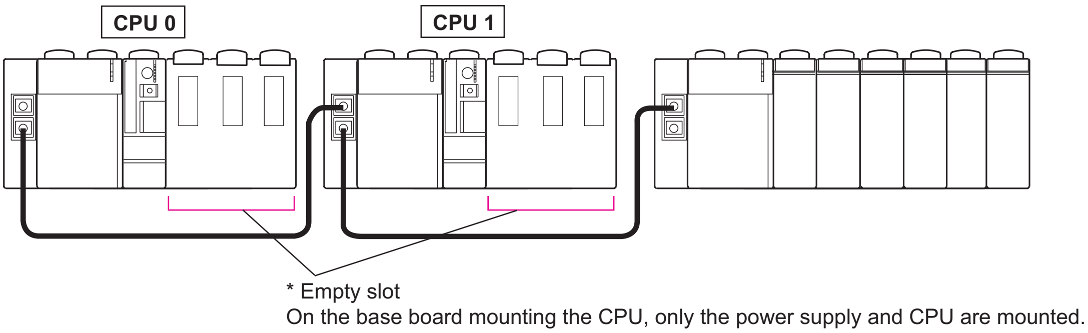
A control system in which CPU modules are used doubly for the purpose of improving safety and reliability is said to be “CPU redundant system”. There are two modes of CPU redundancy: 1-to-1 redundancy and N-to-1 redundancy.

(1) 1-to-1 redundant system

A redundant system with one standby CPU for one working CPU. Each of CPU0-CPU1, CPU2-CPU3, CPU4-CPU5, and CPU6-CPU7 indicates a pair of a working CPU and a standby CPU. Therefore, when building a multi-CPU 1-to-1 redundant system, up to four CPUs are installed. In this case, the same application program is used.

<Example of redundant system configuration>

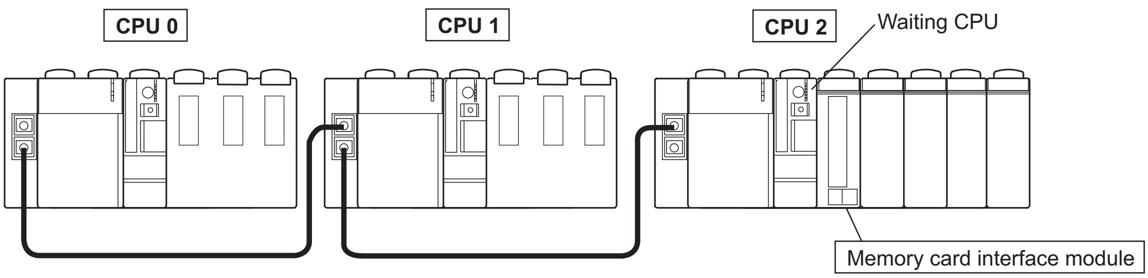
Generally, the working CPU and standby CPU are mounted on different base boards and other modules to be controlled by the CPU are mounted on another base board, as shown below. If the working CPU fails and the standby CPU is activated, this configuration makes it possible to replace the failed CPU during operation of the standby CPU.



(2) N-to-1 redundant system

A redundant system with one standby CPU for multiple (2 to 7) working CPUs. Up to two pairs of N-to-1 redundant groups can be defined for each configuration. The CPU module with the largest CPU number functions as the standby CPU within a registered group. Store the run project of each CPU in the memory card interface module.

<Example of N-to-1 redundancy system configuration>



* For detailed specifications and setup procedures for the redundant system, refer to “Section 5 CPU redundant System.”

Notes: 1) To use the N-to-1 redundant mode, the **30 or later version of memory card interface module (30 or later version of firmware) must have been installed. For the high-performance CPU modules, the 1030 version or later must have been installed.

2) In the case of N-to-1 redundant systems, the maximum program capacity of the NP1PS-245R is 119808 (117K) steps.

3) In the case of N-to-1 redundant systems, the maximum program capacity within a single POU is 4096 steps.

4-3-3 System fail-soft start-up

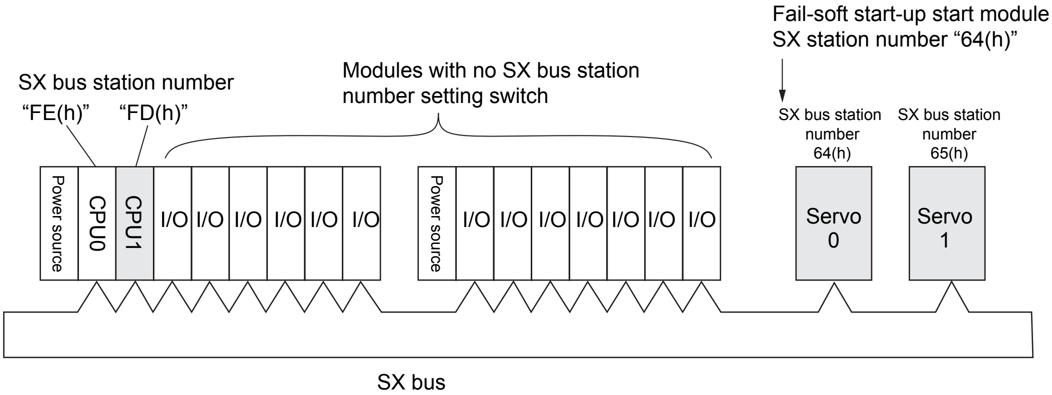
(1) Fail-soft disabled

If all the modules registered in the system definition do not start within the system configuration check waiting time (a default: 20 sec.), a system error occurs (the CPU ALM goes on).

(2) Partial fail-soft start-up of modules with SX bus station numbers

At MICREX-SX system start-up, if some modules on which power is not turned on (modules under fail-soft operation, such as the servo module) are detected, the system is re-started excluding these modules after the system configuration check waiting time has passed. The system runs with a non-fatal fault (the CPU module RUN is on and the ALM is on).

<Example of system configuration at fail-soft start-up>



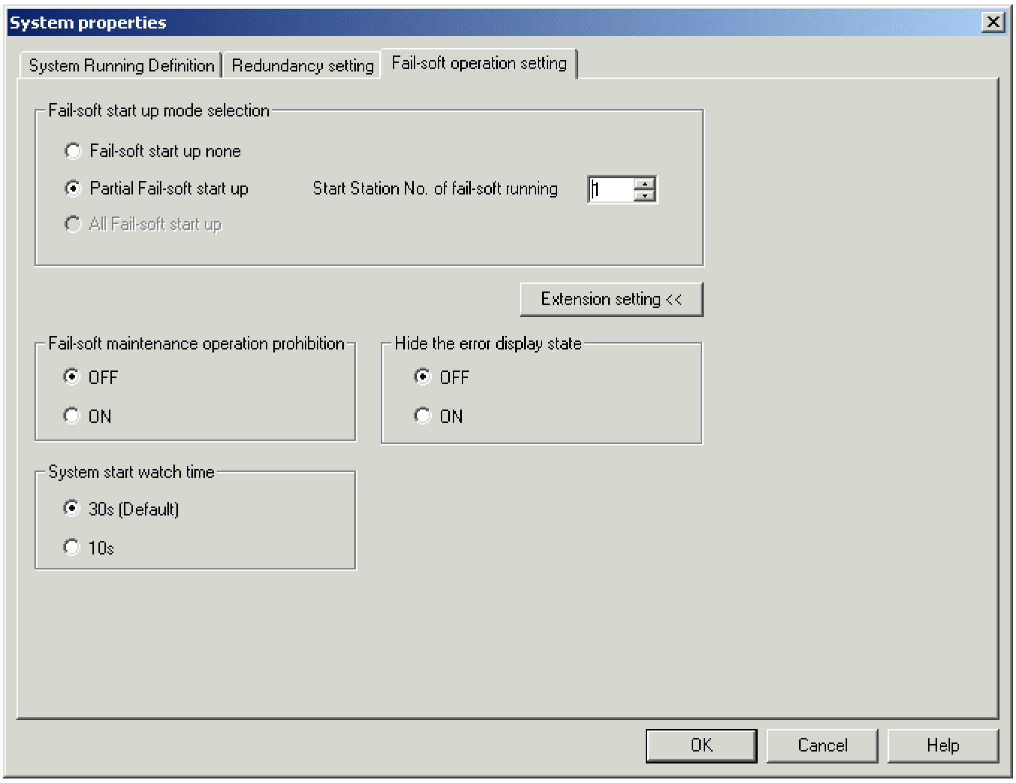
In the above example, the modules to be partially fail-soft started are “servo 0,” “servo 1,” and “CPU 1” in ▭.

1) The fail-soft start-up start module is any of a range from the SX bus station number, which has been set as the first one to be fail-soft started to FD (h). (Note 1)
   Set the numbers excluding the start number for the SX bus station numbers for the module to be fail-soft started.
2) Be sure to assign the SX station numbers after one is declared as the start number only to the modules with an SX bus station number setting switch. (You do not need to assign consecutive numbers to them.) (Note 2)
3) When the module to be fail-soft started is turned on after fail-soft start-up (in the non-fatal fault state), the module is activated and the system runs normally (The RUN is on and the ALM is off).

Notes: 1) The CPU module, PE-link module, and P-link module are fail-soft started as well. Note that CPU0 is excluded.
2) If any module with no SX bus station number setting switch is detected, a system fatal fault (at initial start-up) or non- fatal fault occurs (at successive start-up).
3) Be sure to use the system fail-soft start-up mode only in the fail-soft compatible modules (version 20** or later). If any module incompatible with fail-soft is detected at initial start-up or at successive start-up, a system fatal fault occurs.

<Setting system fail-soft start-up>

◆ Open the “System properties” dialog box and click the [Fail-soft operation setting] tab to display the following window.



◆ If there is a module to be fail-soft started, select “Partial Fail-soft start up” and enter the fail-soft start number.
◆ After setting all necessary items, click the [OK] button.

* Clicking the [Extension setting] button displays the boxes to set [Fail-soft maintenance operation prohibition] mode and [Error state display hiding] mode.

- **Fail-soft maintenance operation prohibition**

  If this item is set to ON, even when the actual configuration does not agree with the system configuration in fail-soft start up mode, additional connection/reconnection of a module are not performed. By setting this, even if an unpowered module exists in the configuration, takt drift can be minimized.
  When this mode is selected, the operation maintenance is not performed. Therefore, if a module is added or reconnected, you need to start up the system again. (Ditto for replacement of a CPU in a redundant system)

- **Hide the error display state**

  If this item is set to ON, even when a module to be fail-soft started does not exist, the system does not go into a nonfatal fault state.
  When using this function in a multi-CPU system or redundant system, if CPUs are individually started up due to replacement of a CPU etc., mask information of each CPU may be different. CPUs must be start up all together, or if started up individually, each system must be in the same state.

- **System start watch time (with V65 or later software version of SPH300)**

  This item is used to shorten the start-up time of a system in which the main power of an inverter/servo directly connected to the SX bus is turned ON after the system is started up, by using the “fail-soft start-up” function. The default is “30s”. If “10s” is selected, the system start-up time can be shortened by 20 seconds. In addition, if “10s” is selected, the system memory (SM2565) is set ON.

  Note: For a multi-CPU system, redundant system, or a system using the PC card interface module (type: NP1F-PC2), do not set the watch time at “10s”.

4-4-1 CPU running definitions

The CPU running definitions include the “watchdog timer”, “running specification at power on”, “battery less run”, “constant scanning” (SPH2000 only), “execution band ratio” (SPH2000 only) and “user ROM run” (SPH2000 only).

(1) Watchdog timer

The user can set a watchdog timer preset value from 1 ms to 4095 ms. The default value is 4095 ms.

(2) Running specification at power on

This parameter specifies the operation that the CPU module is to perform when system power is turned on when the key switch on the CPU module front panel is set to RUN or TERM.
The table given below shows the relationship between the CPU module key switch positions and the CPU module operations.
A default value is “RUN = Running/TERM = Running.”

<CPU operations and key switch positions>

System Definition Setting	Operation	
	RUN	TERM
RUN=run/TERM=run	run	run
RUN=run/TERM=preceding state	run	preceding state (Note)
Run=stop/TERM=stop	stop	stop

Note: The preceding state is the state of the CPU established before system power is shut off. The state is set to run if the CPU was running and to stop if it was stopped.

(3) Battery less run

In case of battery-less running, the CPU module, when activated, initializes its memory to start operation. No data backup error is detected. The default value is battery-less running disabled. For the standard CPUs, battery-less running is not applicable if they have no user ROM card (MP8PMF-16) inserted.

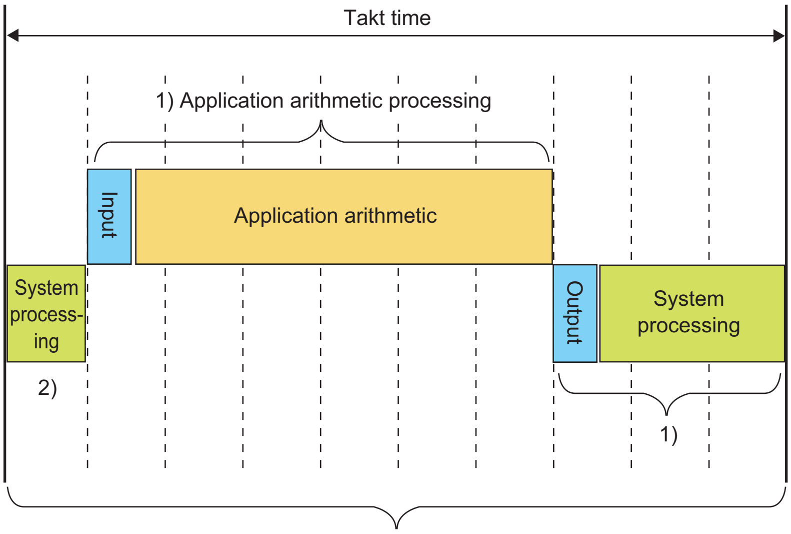
(4) Constant scanning (SPH2000 only)

Processing of the default task (input + operation + output) is executed at regular intervals (scan time specified here).
The setting range is as follows: 1 takt time <= (set time) <= 2550ms. Set an integral multiple of the takt time of the system.
* By default, “No” is selected for the constant scanning setting.

Note: Constant scan operates in an integral multiple of the set takt. If the system cannot operate in the time, constant scan cannot operate symultaneously, either. Be sure to set the takt time at a proper value according to the system configuration.

(5) Execution band ratio (SPH2000 only)

This sets the ratio of application processing time to system processing time within a single Takt time.
The SPH2000 system makes ample use of message-communication FBs (e.g. M_SEND, M_RECEIVE, R_READ, and R_WRITE). If the processing load of message communications is large, you can change the execution band ratio to increase the efficiency of message-communication processing.
The ratios of application arithmetic processing to system processing are 1 to 9, 2 to 8, 3 to 7, 4 to 6, 5 to 5, or 6 to 4.
The default is 6 to 4.



A takt time is divided into 10 portions. The relation of system processing to application arithmetic processing can be changed.

1) Application arithmetic processing
Input processing and application arithmetic

2) System processing
Output processing, SPH2000 system monitoring, message-communication processing, application arithmetic monitoring, etc.

(6) User ROM run (SPH2000 only)

This sets whether to perform ROM operation using the user ROM card installed in the CPU module.
* By default, user ROM operation is set to be performed.

Note: Even if user ROM operation is not set, it is possible to read and write data to/from a user ROM card from an application program.

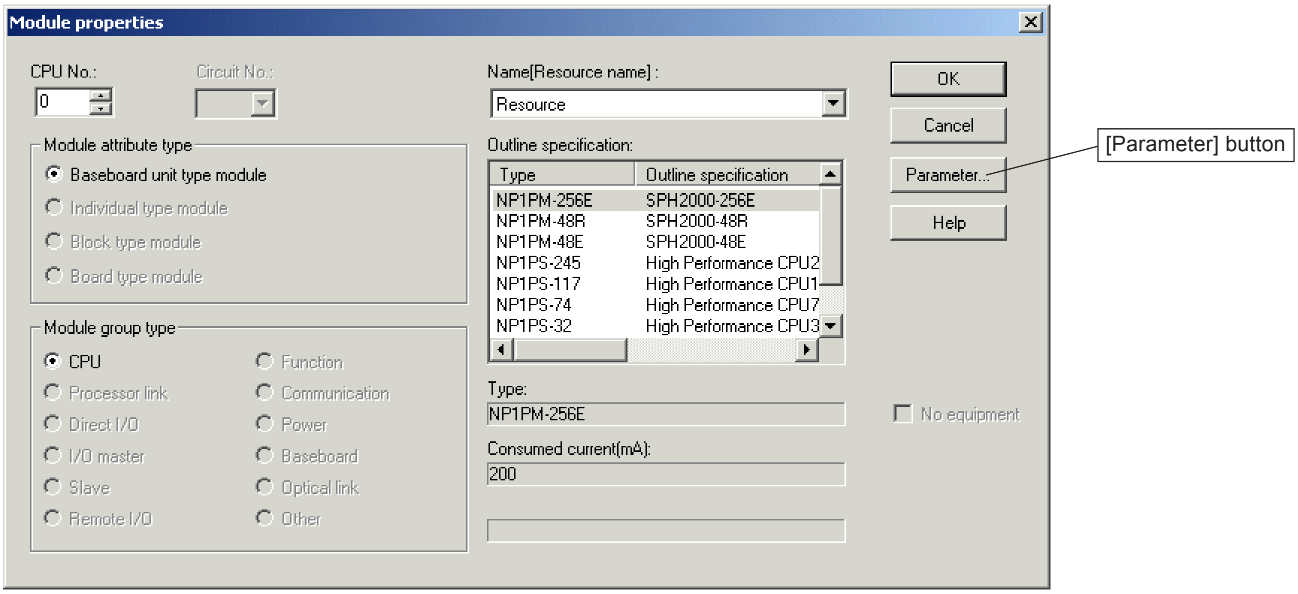
(7) Compulsion setting hold state (with V65 or later software version of SPH300)

In this mode, operation is started with compulsion settings for I/O held. By default, the compulsion setting state when energized last time is cleared when the power of the CPU module is turned on (including reset). However, if the “compulsion setting hold state” is set ON, the compulsion settings are not cleared and operation is started with the compulsion settings held. When the “compulsion setting hold state” is set ON, the system memory (SM256E) is set ON. In addition, when there is a compulsion setting to hold, the system memory (SM256F) is set ON.

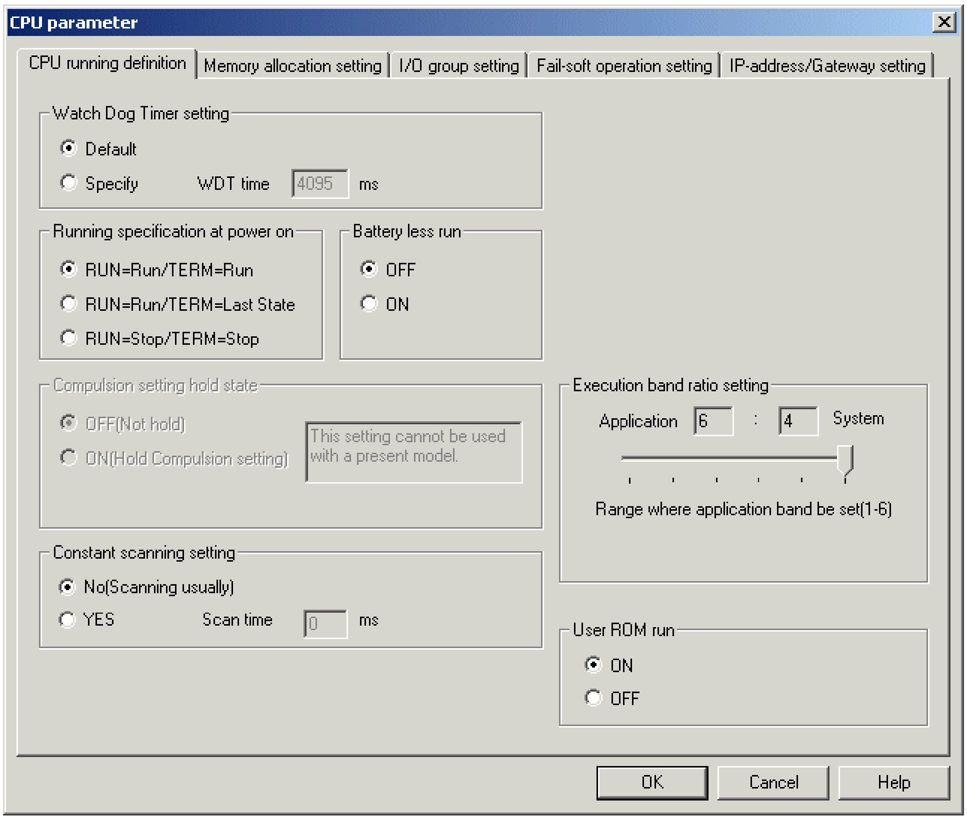
4-4 CPU Parameters

<Setup procedure>

◆ Select a CPU from the system definition screen and then click the [Property] button. The [Module properties] dialog for the CPU appears.



◆ Click the [Parameter] button. The [CPU parameter] dialog appears.
The dialog consists of four tab pages. The “CPU running definition” tab page is displayed first.



◆ After setting all necessary items, click the [OK] button.

4-4-2 CPU memory size definition

The user can set the size of the data memory in the CPU module.
Although the user memory area has a determined default size, the user can change the size of each area as required. The size of each area may be changed in 0.5K word increments.

<Allowable memory area size ranges>

The size of the user memory area may be changed in the range shown below. Note that the total number of words for user memory is fixed. For example, to set 32K words for the standard memory of High-performance CPU (NP1PS-32), specify all 0s for other memory areas.

SPH300 (NP1PS-32/NP1PS-32R)

Memory type		Min. value	Initial value	Max. value
Standard memory (non-retained memory)	(WM)	2K words	8K words	32K words
Retain memory	(WL)	0K word	4K words	30K words
User FB memory		0K word	4K words	28K words
System FB memory		0K word	16K words	30K words

SPH300 (NP1PS-74/NP1PS-74R)

Memory type		Min. value	Initial value	Max. value
Standard memory (non-retained memory)	(WM)	2K words	32K words	128K words
Retain memory	(WL)	0K word	16K words	126K words
User FB memory		0K word	16K words	113K words
System FB memory		0K word	64K words	126K words

SPH300 (NP1PS-117/NP1PS-117R)

Memory type		Min. value	Initial value	Max. value
Standard memory (non-retained memory)	(WM)	2K words	128K words	256K words
Retain memory	(WL)	0K word	32K words	254K words
User FB memory		0K word	32K words	113K words
System FB memory		0K word	64K words	254K words

SPH300 (NP1PS-245R)

Memory type		Min. value	Initial Value	Max. value
Standard memory (non-retained memory)	(WM)	2K words	256K words	383K words (note)
Retain memory	(WL)	0K word	127K words	381K words (note)
User FB memory		0K word	65K words	113K words
System FB memory		0K word	64K words	510K words

Note: Standard memory
+ Retain memory
<= 383K

SPH2000 (NP1PM-48R/48E)

Memory type		Min. value	Initial value	Max. value
Standard memory (non-retained memory)	(WM)	64K words	64K words	96K words
Multi-CPU non-retain memory	(WM)	0K word	0K word	32K words
Retain memory	(WL)	0K word	8K words	32K words
Multi-CPU retain memory	(WL)	0K word	0K word	32K words
User FB memory		0K word	8K words	32K words
System FB memory		0K word	16K words	32K words

SPH2000 (NP1PM-256E)

Memory type		Min. value	Initial value	MAx. value
Standard memory (non-retained memory)	(WM)	1664K words	1664K words	2048K words
Multi-CPU non-retain memory	(WM)	0K word	0K word	384K words
Retain memory	(WL)	0K word	32K words	32K words
Multi-CPU retain memory for	(WL)	0K word	224K word	384K words
User FB memory		0K word	64K words	384K words
System FB memory		0K word	64K words	384K words

SPH200 (NP1PH-16)

Memory type		Min. value	Initial value	Max. value
Standard memory (non-retained memory)	(WM)	0K word	8K words	31K words
Retain memory	(WL)	0K word	4K words	31K words
User FB memory		0K word	4K words	14928 words
System FB memory		0K word	8K words	31K words
Initial value setup area		0K word	7K words	31K words

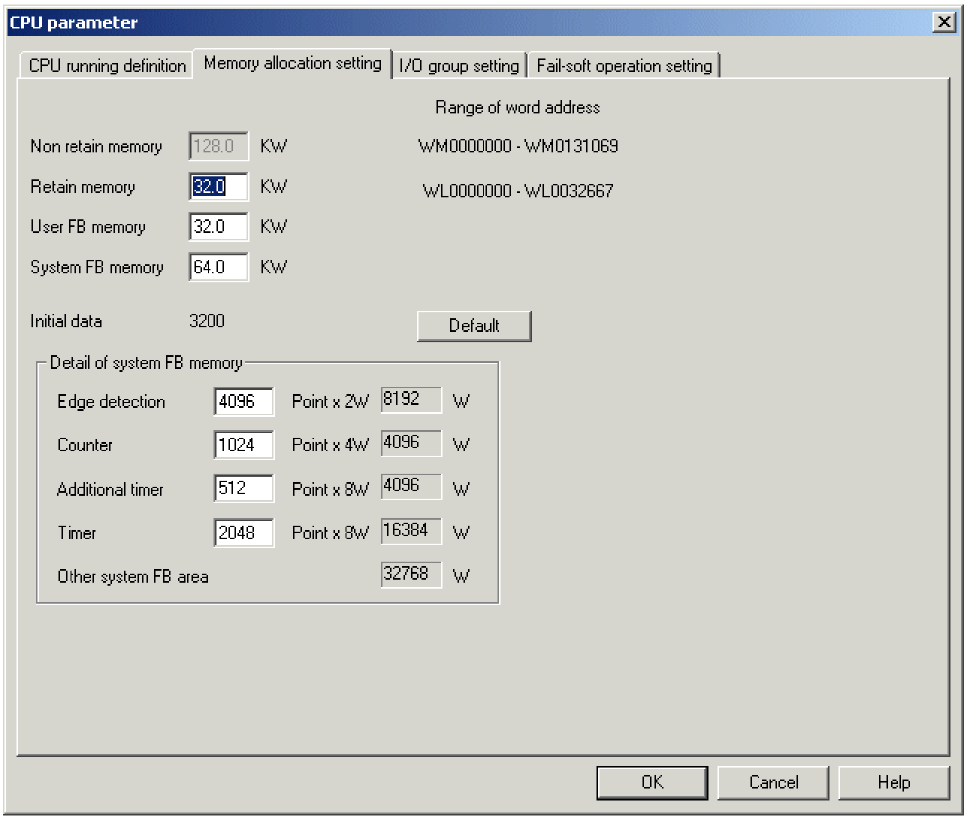
SPH200 (NP1PH-08)

Memory type		Min. value	Initial value	Max. value
Standard memory (non-retained memory)	(WM)	0K word	4K words	15K words
Retain memory	(WL)	0K word	2K words	15K words
User FB memory		0K word	2K words	7216 words
System FB memory		0K word	4K words	15K words
Initial value setup area		0K word	3K words	15K words

4-4 CPU Parameters

<Setup procedure (1) SPH300, SPH200>

◆ Click the [Memory allocation setting] tab on the [CPU parameter] dialog to display the following window.
Set the sizes of “Retain memory”, “User FB memory”, and “System FB memory”.
The size of remaining free memory is assigned to non-retain memory (standard memory). The memory size can be set in 0.5K word increments.



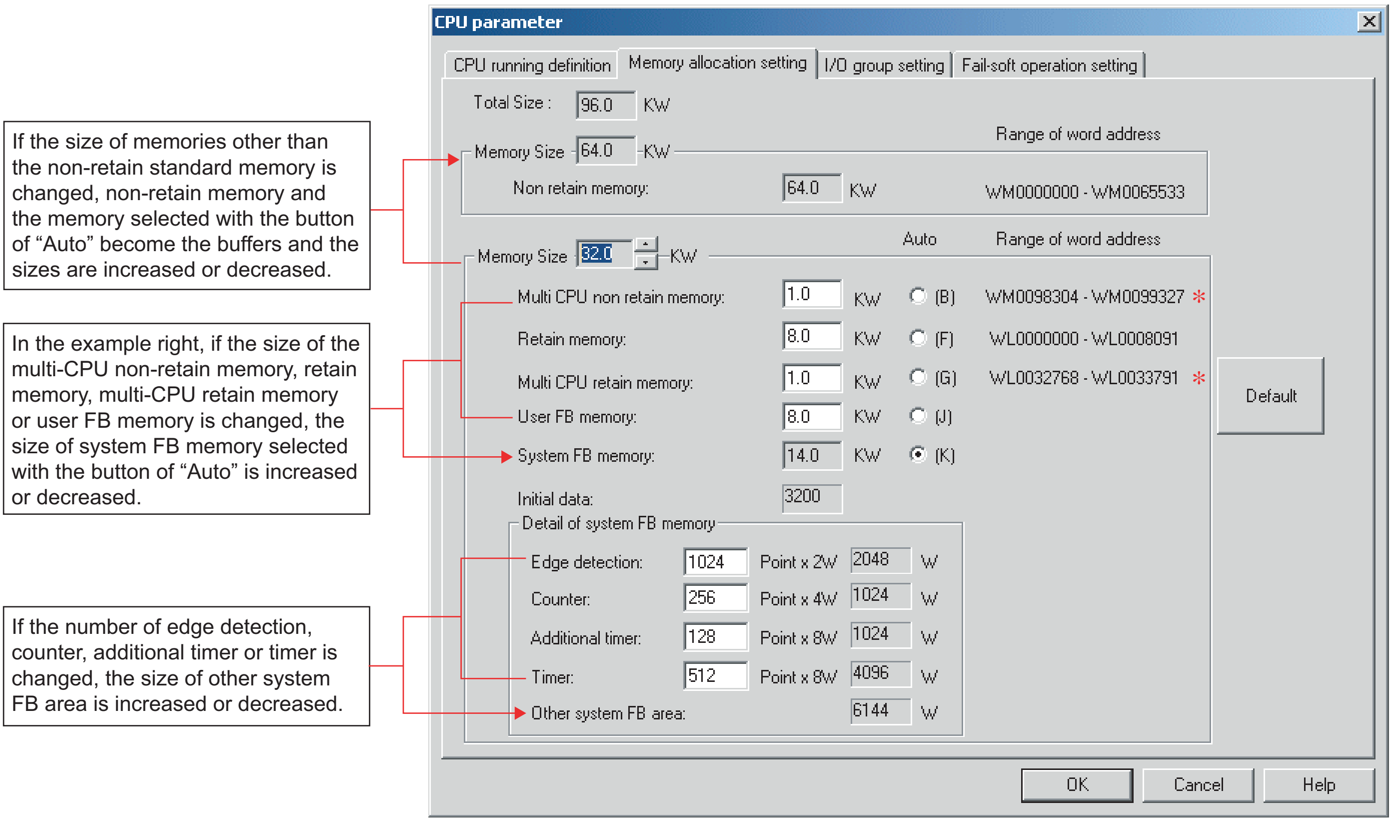
◆ After setting the memory sizes, click the [OK] button.

Note: For standard CPU, the number of initial data pieces vary depending on the non-retain memory, retain memory, user FB memory and system FB memory settings. (The user FB initial value area is also automatically calculated.)

4-4 CPU Parameters

<Setup procedure (2) SPH2000>

◆ Click the [Memory allocation setting] tab on the [CPU parameter] dialog to display the following window.
The non-retain standard memory and the other memories are separately set.



◆ After setting the memory sizes, click the [OK] button.

* In a multi-CPU system of the SPH2000 series, when accessing the memories between CPUs using the processor bus, the standard memory must be reserved in the multi-CPU non-retain memory and the retain memory must be reserved in the multi-CPU retain memory. The addresses of reserved memories are as follows:

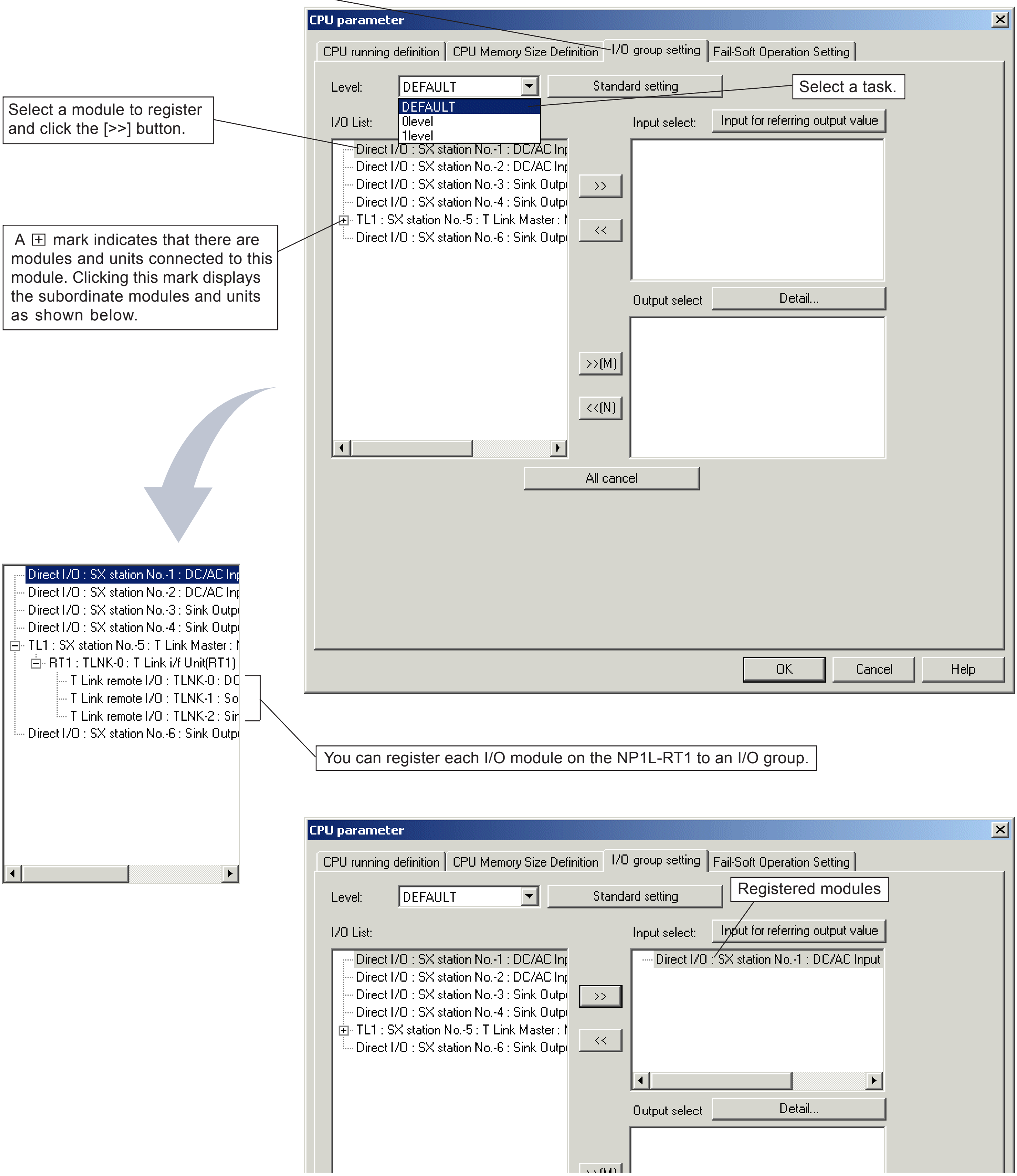
- Multi-CPU non-retain memory: from WM☐.98304
- Multi-CPU retain memory: from WL☐.32768

4-4-3 I/O group setting

The user can specify which I/O module in a SPH system can be controlled by which task in which CPU module.
Be sure to set a CPU I/O group for I/O modules. If there is an I/O module with no I/O group assigned, the confirmation dialog to ask if you want to assign it to the default task appears upon exiting the system definition. Even if an error occurs in a module with no I/O group assigned, the CPU cannot detect it.

<Setup procedure (1)>

◆ Click the [I/O group setting] tab in the [CPU parameter] dialog. The following window appears.

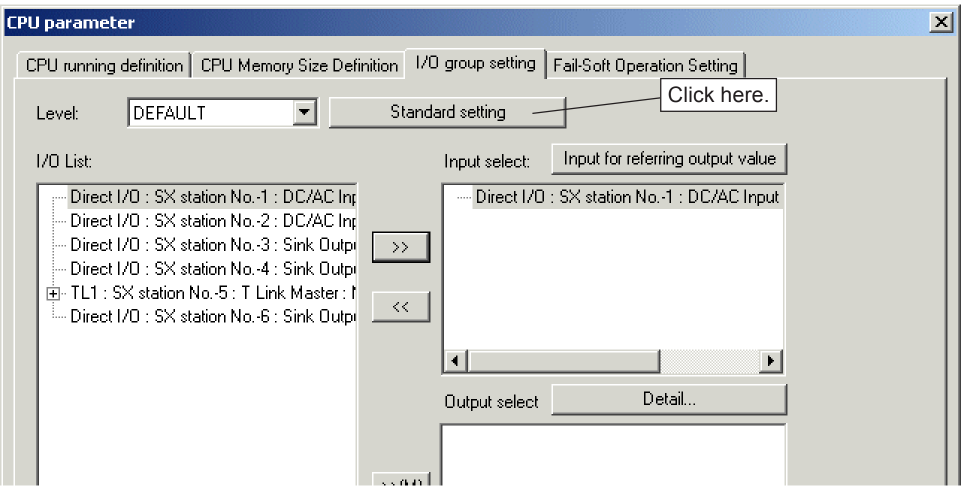


◆ The modules are registered as shown in the upper right window. After setting the necessary items, click the [OK] button.

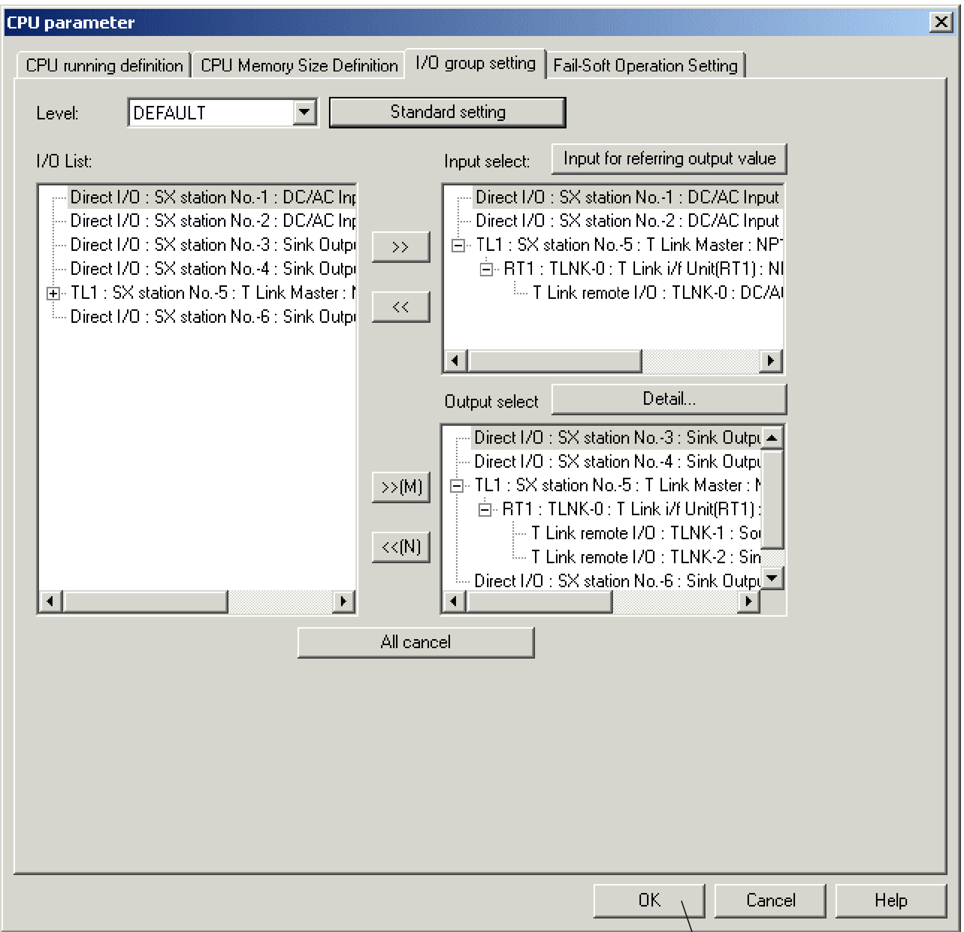
<Setup procedure (2)>

The user can use the [Standard setting] button when registering modules to I/O groups only for the “DEFAULT” tasks in a single CPU system.

◆ Make sure that the Level field is set to “DEFAULT” and click the [Standard setting] button.



◆ Input modules are automatically registered to the Input selection and the output modules to the Output selection as shown below.

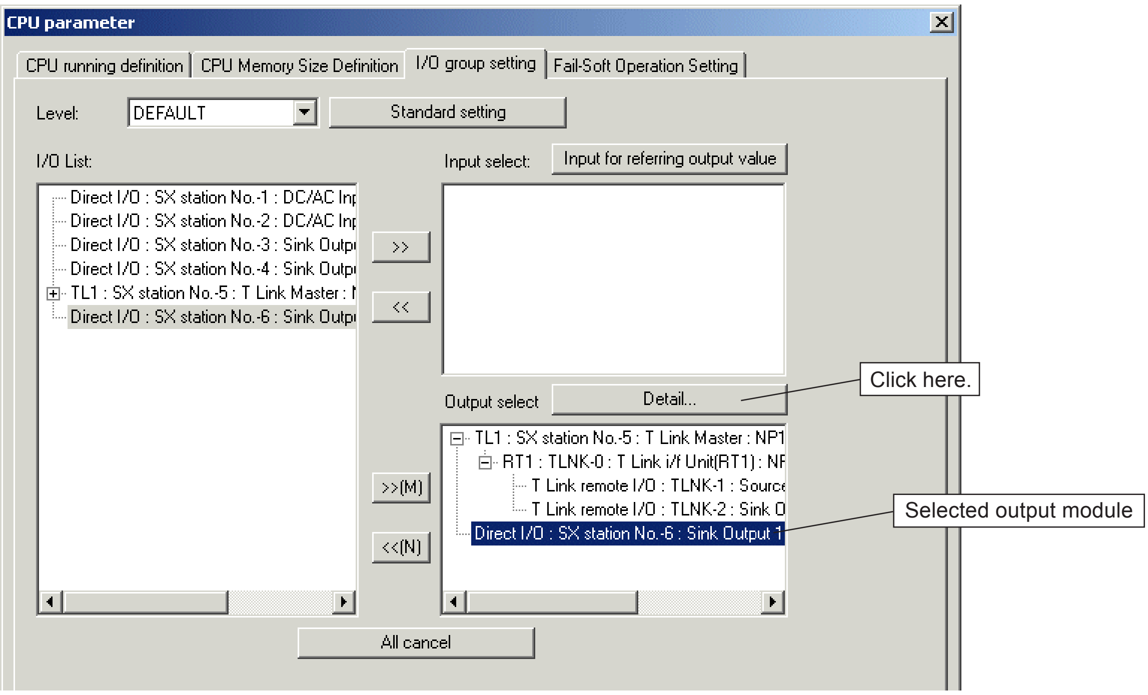


After determining the settings, click here.

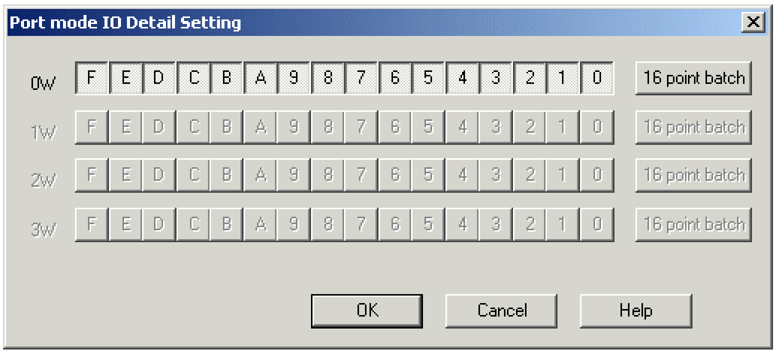
<Setup procedure (3)>

The user can register the output modules to different CPUs on a bit basis, for example, in a multi-CPU system.

◆ Select the output modules that are to be registered on a bit basis from the Output selection and click the [Detail...] button.

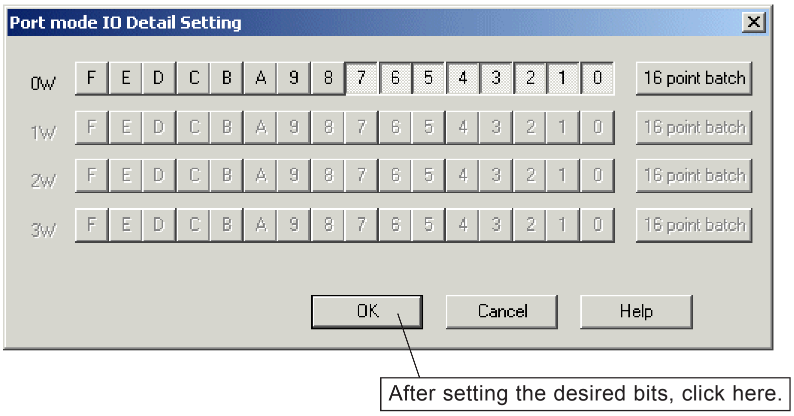


◆ The following dialog is displayed. Initially, all bits are selected.



* A pressed button indicates that the corresponding bit is registered to the I/O group.

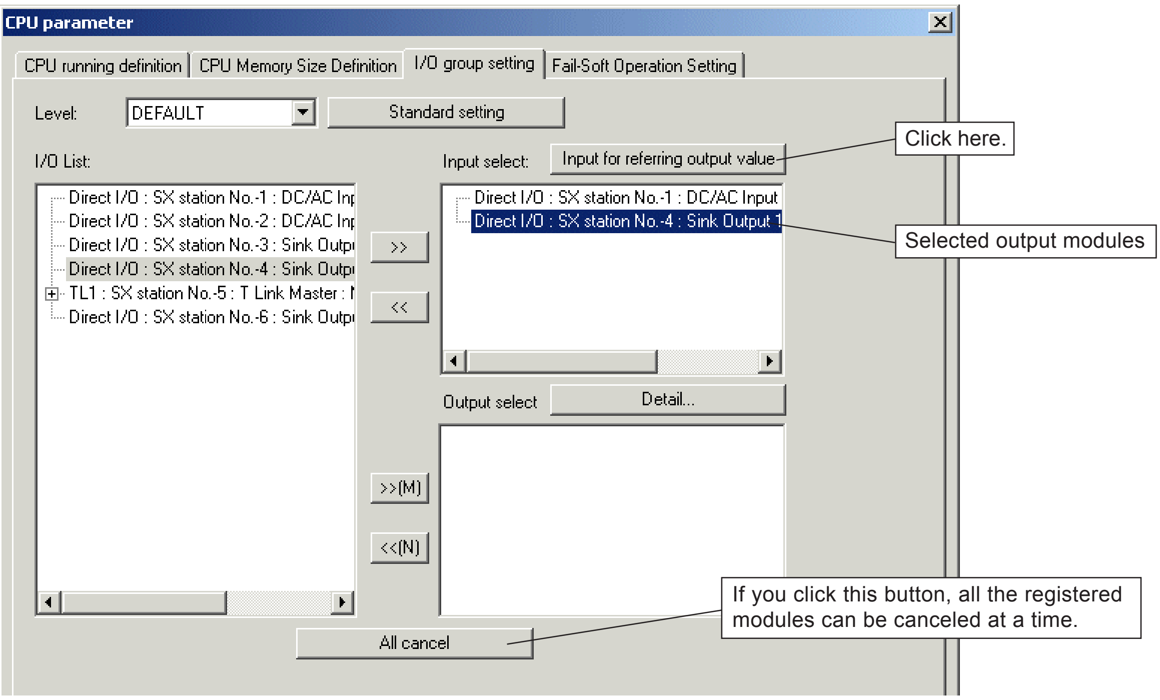
◆ Click a bit that is not registered to the I/O group to set off. In the figure shown below, bits 8-15 are set off.



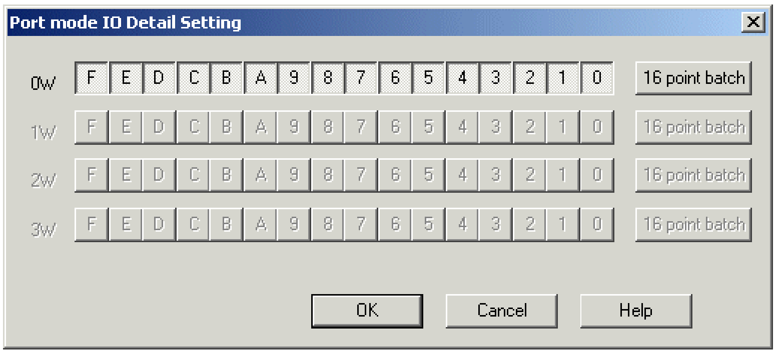
<Setup procedure (4)>

The user can register bits of an output module that is used by another CPU in a multi-CPU system to the input selection I/O group of the local CPU as “inputs” to an application program running on the local CPU.

◆ Select an output module and click the [Input for referring output value] button.

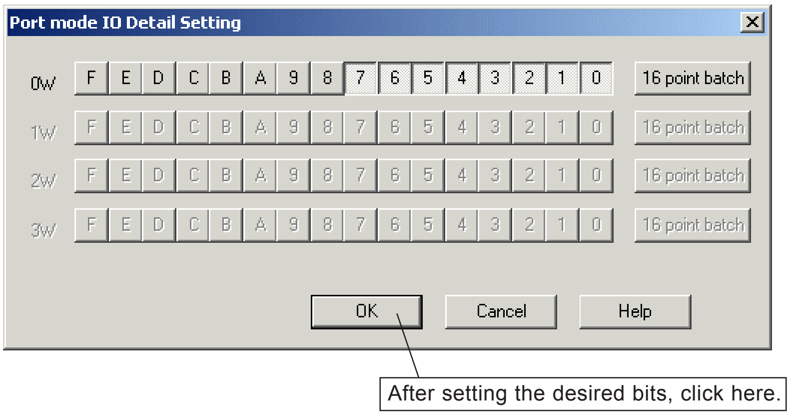


◆ The following dialog is displayed. Initially, all bits are selected.



* A pressed button indicates that the corresponding bit is registered to the I/O group for output reference.

◆ For a bit whose output is not to be referenced, click it to set off. In the figure shown below, bits 8-15 are set off.



4-4-4 Fail-soft running

Even if a fault occurs in the module with fail-soft enabled and goes down during system running, the entire system can continue running.

<Fail-soft operation of MICREX-SX system>

1) Module down

Even if the module with fail-soft enabled has a fault and goes down during system running, the CPU continues running with a non-fatal system fault (The RUN is On and the ALM is on).
If the module with fail-soft disabled has a fault and goes down during system running, the CPU stops with a fatal system fault (RUN is off and ALM is on).

2) Module recovery

When the down module is recovered, the system returns to its normal state.
Note that if more than one module has been down, no module can be activated unless all the down modules are recovered.

3) Modules with fail-soft enabled

Common modules with no I/O area	CPU, P/PE-link, and general communication modules	Unconditionally, fail-soft is applicable
Modules with I/O area	Digital I/O, analog I/O, and AS-I master modules	Register the modules with fail-soft enabled

Note: If you want to stop the system running when a fault occurs in the module to which fail-soft is unconditionally applicable, monitor any SPH system fault information in system memory (WSM68 to WSM83) and set the user fatal fault flag (WSM14 to WSM16) to “ON.”

4) I/O group and fail-soft registration

The module CPU does not control a module (having an I/O area assigned) not registered in an I/O group. Even if the module goes down, the CPU continues running normally.

Considerations in building the fail-soft system

The module and base unit versions, which support fail-soft, are listed below.
- Modules with internal firmware installed 1030
- modules with no internal firmware installed 10 (for example, a base unit)

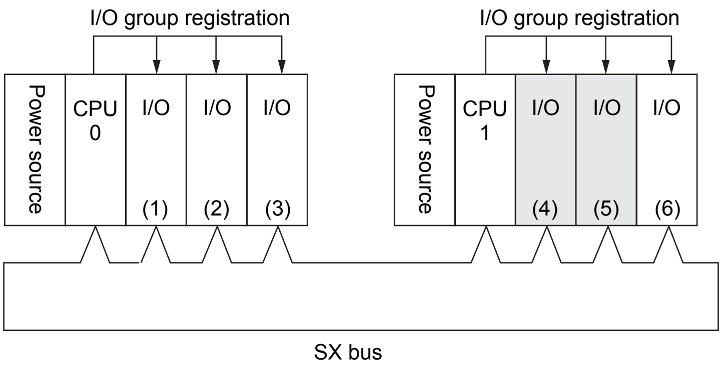
When any module, of a version earlier than that listed above, has been connected to the SX bus, no fail-soft can be implemented on it. The SX bus-connected module fail-soft mode flag (SMD) is set to "OFF" in system memory.

Note: Even in a system in which a module incompatible with fail-soft has been configured, the remote I/O fail-soft feature can be used.

5) Fail-soft registration for the multi-CPU system

For the multi-CPU system, register the modules controlled by the self-CPU in the I/O group and also register them for fail-soft. The system operates as described below in case of a fault.

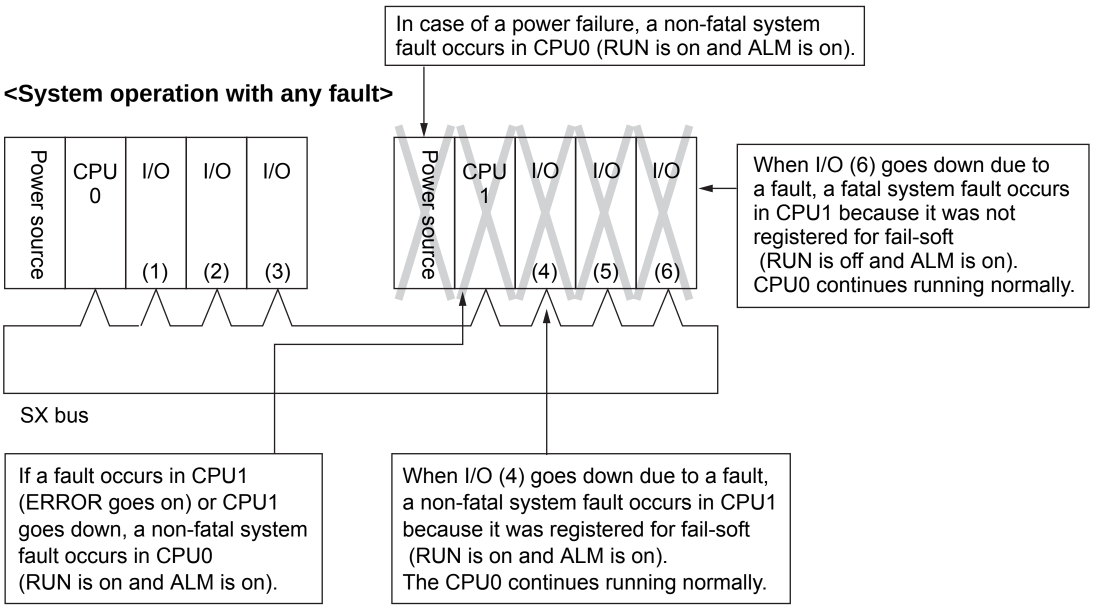
<Example of system configuration>



In CPU0, register I/O (1), (2), and (3) together as one I/O group and also register them for fail-soft.

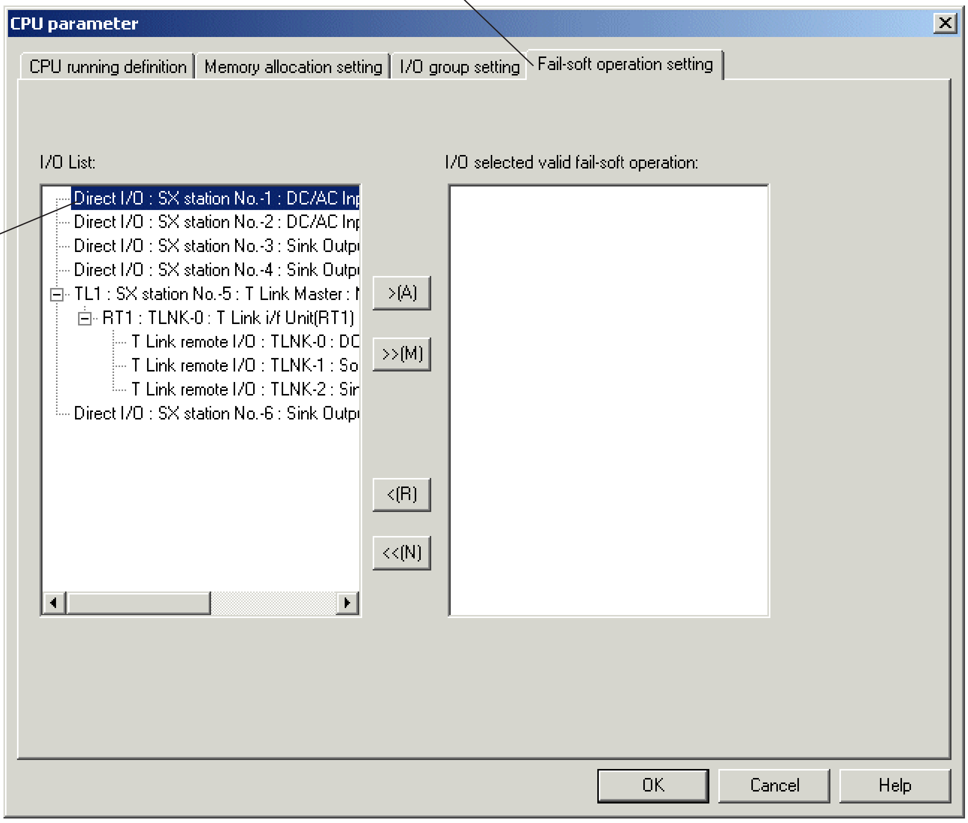
In CPU1, register I/O (4) and (5), and (6) together as one I/O group and also register (4) and (5) for fail-soft.

<System operation with any fault>



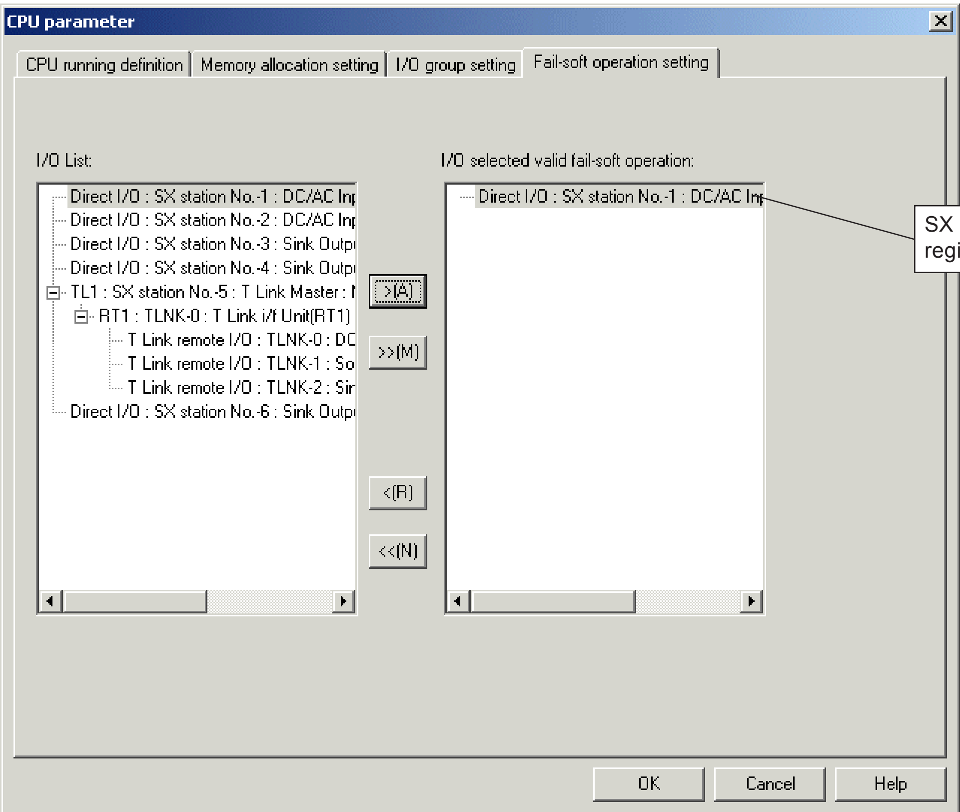
<Setup procedure>

◆ Open the [CPU parameter] dialog and click the [Fail-Soft Operation Setting] tab.



Select the module (left-click) for which fail-soft is to be set and click the [>>] button.
You cannot set fail-soft individually for the I/O modules on a group-type I/O unit.

◆ The modules are registered as shown below.



SX bus-connected modules can be registered for fail-soft individually.

* For a group-type I/O module unit, all I/O modules on the base board are registered for fail-soft.

4-4-5 IP address, gateway

Set the parameter of the CPU module with the Ethernet function of SPH2000.
* For the specifications and operations of the Ethernet function, refer to “User’s Manual SPH2000 Ethernet Communications (FEH193)”.

1) IP address, Default gateway

Set the IP address, subnet mask and default gateway of the CPU module.

2) Detail settings

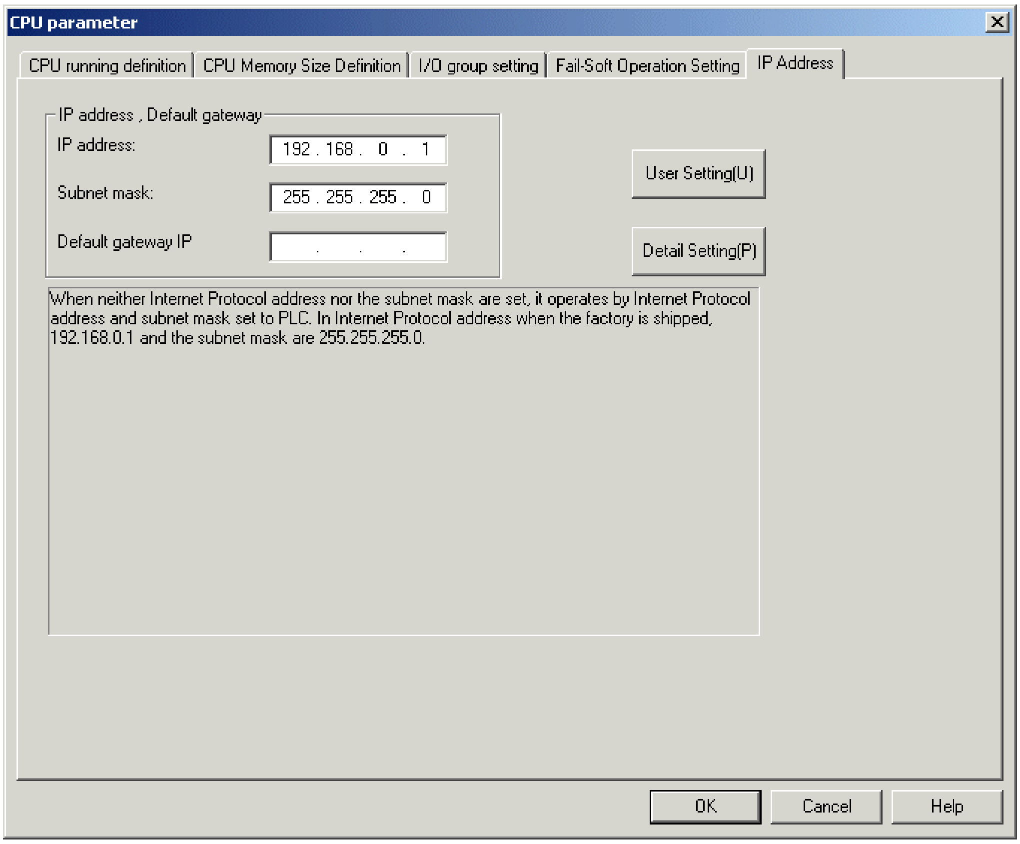
Perform detail settings for Ethernet communications. The setting items are as follows:
- TCP timeout value
- Maximum TCP end timer value
- Loader command watch timer value
- Sending retry timeout
- Sending retry number of times
- Close process during TCP sending timeout
- Close process during response receiving timeout
- Self port standard No.
- Response monitoring timer value

3) FTP user registration

The CPU module with the Ethernet function of SPH2000 has the FTP function. Users who will access the FTP server are registered.

<Setting procedure>

◆ Display the [CPU parameter] dialog and click the [IP Address] tab. The following window is displayed. After setting all necessary items, click the [OK] button.

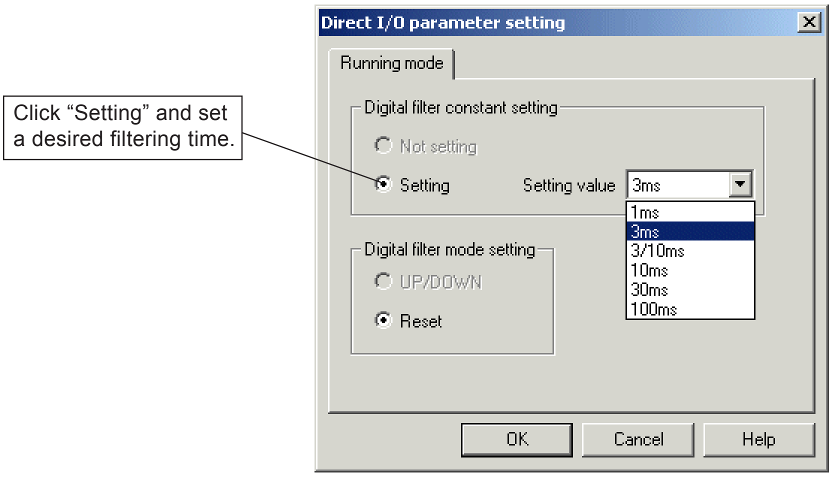


4-5-1 Input filtering time

The input filtering time is set for a DC-type digital input module. The time is set in the format (OFF-to-ON time) - (ON-to-OFF time). The user can select a value from 1-1ms, 3-3ms (default), 3-10ms, 10-10ms, 30-30ms, 100-100ms, and no filtering. Only for a fast input module (NP1X3206-A) can no filter or a 100-100μs input filtering time be set.

<Setup procedure>

◆ From the system definition tree, select a digital input module for which the input filtering time is to be set,and open the parameter set up dialog box.



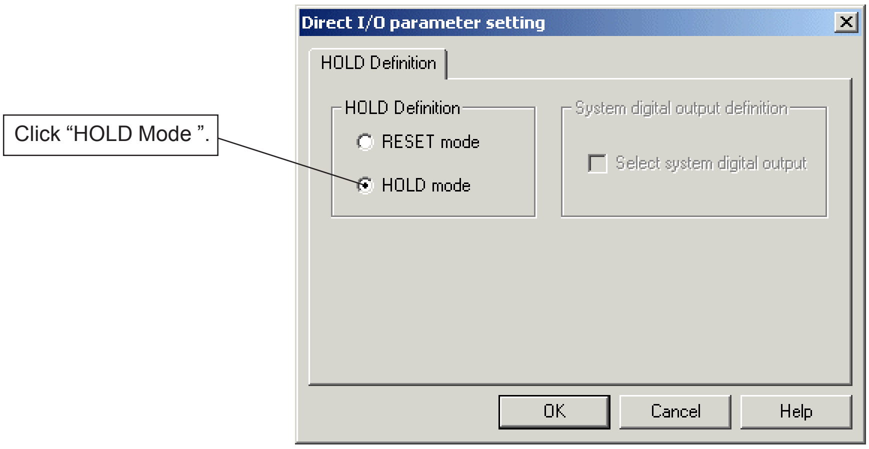
◆ After specifying a filtering time, click the [OK] button with the left mouse button.

4-5-2 Output hold definition

The output hold definition is used to preserve the output state established immediately before a system error occurs or a CPU module is shut down or to preserve the output state established immediately before the CPU is stopped while the CPU is held stopped.

<Setup procedure>

◆ Select, from the system definition tree, a digital output module for which the output hold option is to be set, and open the parameter set up dialog box.



◆ After selecting the hold mode, click the [OK] button with the left mouse button.

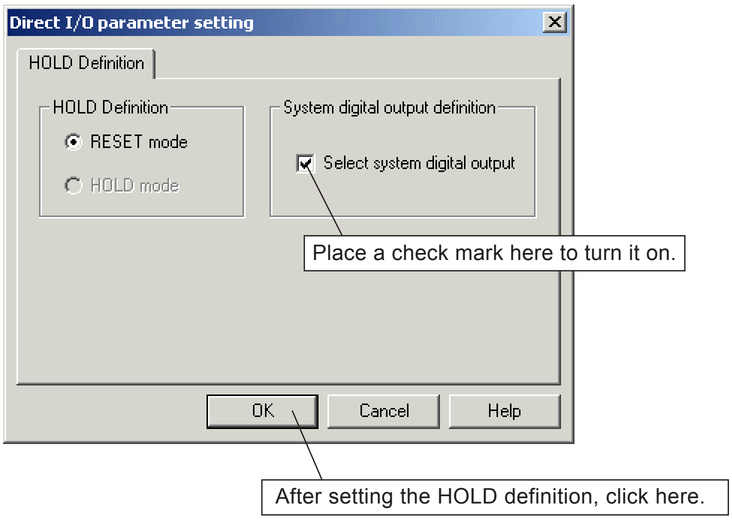
Note: For remote I/Os on OPCN-1, output hold cannot be set.

4-5-3 System output definitions

The user can define a bit, for each SPH system, that indicates the system operation state regardless of the application. This bit is set to ON when the entire system is running normally and set to OFF if the system has an error. The bit which can be set is bit 0 of the output module.

<Setup procedure>

◆ Select the digital output module for which system output is to be set and open the parameter setting dialog.



Notes: 1) Bit 0 of the module to be specified as system output cannot be registered in the I/O group definition.
2) A module to be specified as system output cannot be registered in the I/O group definition for any CPU other than CPU0.
3) Neither HOLD mode nor fail-soft can be set for a module that is specified as system output.

Section 5 CPU Redundant System

This is called the redundant mode because dual devices are used to improve system safety and reliability in the control system. In the MICREX-SX Series, the power modules and CPU modules (high-performance CPUs only) can be built into the dual systems. In this section, how to build the CPU modules into the redundant system is described.
The CPU redundant mode includes 1-to-1 and N-to-1 types.

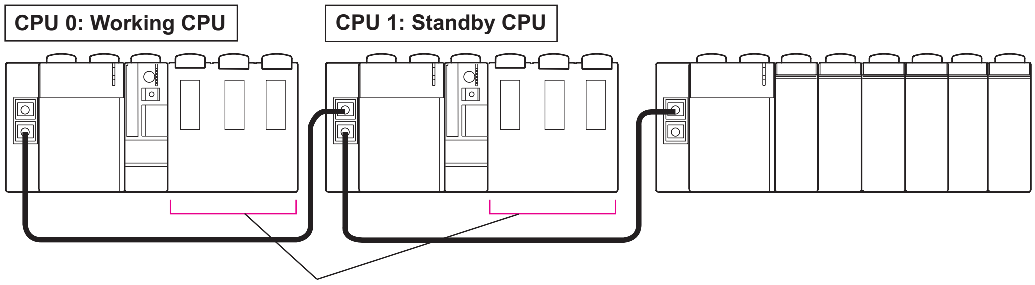
5-1 System Operation in the Redundant Mode

5-1-1 1-to-1 redundant mode

A redundant system with one standby CPU for one working CPU. Each of CPU0-CPU1, CPU2-CPU3, CPU4-CPU5, and CPU6-CPU7 indicates a pair of a working CPU and a standby CPU. The same application program is used.

<Example of 1-to-1 redundant system configuration>

Generally, the working CPU and standby CPU are mounted on different base boards and other modules to be controlled by the CPU are mounted on another base board, as shown below. If the working CPU fails and the standby CPU is activated, this configuration makes it possible to replace the failed CPU during operation of the standby CPU.



* Empty slot
On the base board mounting the CPU, only the power supply and CPU are mounted.

Note: It is possible to configure a redundant system with a base board on which both a working CPU and standby CPU are mounted. In this case, use a hot plug base board to replace the failed CPU during system running.

(1) System operation

At system power-on, the system starts running, assuming that the CPU modules with even CPU numbers assigned are on the working side while those with odd CPU numbers are on the standby side. (In the above example, CPU0 is a working CPU and CPU1 is a standby one.) Additionally, on redundant systems, successive startup with CPUs other than CPU0 is possible even if fail-soft start-up is not configured. When the working CPU has a fault and goes down, the standby CPU starts running.
The 1-to-1 redundant mode includes two types, warm standby in which the standby CPU inherits data from the working CPU and cold standby in which the standby CPU does not do so. The data inherited by the standby CPU is called equalized data, and its range is specified in the system definition.

(2) Replacing a faulty CPU with a new one

In the above example, since only the power module has been installed on the base board with a CPU mounted on it, the CPU0 can be replaced while CPU1 is running instead of the downed CPU0. To replace the CPU0 with a new one, turn the CPU0 power off, replace it with a new one (See note.), and turn the power on. A new CPU0 is assumed to be on the standby side.
When faults occur in both CPUs (on the working and standby sides), turn off the power on both of the systems and then re-start them.

Note: A new CPU module must to be the same type as the old one and you need to download the project into it in advance.

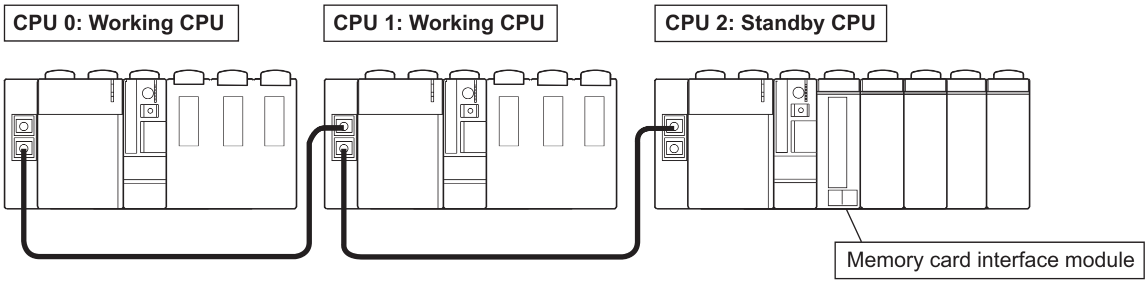
Key points

- Install the same application on both the working and standby CPUs. Additionally, all CPUs on one configuration have the same system configuration. If one of the CPUs has a different actual configuration, a system definition error will be generated for that CPU.
- If all CPUs on a redundant system (high-performance CPUs) have software version 63 or later, it is possible to start up the system even if CPU0 is not installed or is powered down. (System non-fatal fault state).

5-1-2 N-to-1 redundant mode

A redundant system with one standby CPU for multiple (2 to 7) working CPUs. Up to two pairs of N-to-1 redundant groups can be defined for each configuration. The CPU module with the largest CPU number functions as the standby CPU within a registered group.

<Example of 2-to-1 redundant system configuration>



(1) System operation

At system power-on, the CPU module with the largest CPU number in the N-to-1 redundant system is assumed to be a standby CPU. (In the above example, CPU0 and CPU1 are working CPUs and CPU2 is a standby CPU.)
When the system goes down due to a fault in CPU0 or CPU1, the standby CPU downloads the program of the faulty CPU from the memory card interface module and starts running.
In the N-to-1 redundant mode, only the cold standby method can be applied. No data is inherited from the working CPU.

(2) Replacing a faulty CPU with a new one

In the above example, since I/O modules, etc. have been installed on each base board with a CPU mounted on it, the system needs to be stopped to replace the faulty CPU. If you want to replace a CPU without stopping the system, it is necessary to configure a system in which no other module has been installed on the base board with a CPU mounted on it as shown in the example of 1-to-1 redundant system configuration.

Key points

- N (the number of working CPUs) application programs need to be stored on the memory card interface module. (create programs with 4096 steps per POU. N-to-1 redundant systems cannot use applications with 8192 steps per POU. Additionally, the maximum capacity of NP1PS-245R programs is 19808 steps.)
- In the N-to-1 redundant mode, only the cold standby method is applicable. No internal data and I/O data are inherited.
- All CPUs on the system have the same system definition. If one of the CPUs has a different actual configuration, a definition error will be generated for that CPU.
- When replacing a faulty CPU, the working CPU must be switched (using the loader.)
- In the N-to-1 redundant system, program read/write operation by the switches on the front face of the if memory card in the module is prohibited. Do not use the memory card if module for storing the application programs for the N-to-1 redundant system with file memory for file read/write access from the application program running on the CPU. Prepare another memory card if module for file read/write. If file memory is used with the memory card if module, an access contention occurs and changeover may not be performed between the Workting and standby CPU.
- When starting up the system in an N-to-1 redundant system, CPU0 is always required. Note, however, that the system will start up with a non-fatal error even if one of the default standby or default working CPU (excluding CPU0), or the standby CPU is not mounted.

5-2 Conditions for Changeover between Working and Standby CPUs and Performance

5-2-1 Conditions for changeover

The conditions in which changeover occurs between working and standby CPUs are shown below. The conditions are the same for both 1-to-1 and N-to-1 modes.

O: changeover -: no changeover

Fatal fault in working CPU	Fault in CPU	• Fault in application operation processor • Fault in OS processor	O
	Fault in memory	• Fault in system memory (ROM/RAM) • Fault in application memory (ROM/RAM) • Fault in memory battery backup	O
	Fault in SX bus	• Fault in SX bus control LSI • Fault in processor bus access (caused by self-module)	O
		• Duplicate station number • Excessive number of connected modules • Fault in SX bus transmission • Delay in I/O refresh	O (Note 1)
Fatal fault in working resource	Power failure	• Base power shutdown	O
	Application error	• User program error • Application WDT error • Application run error	O
	Fault in I/O module	• Fault in SX bus-connected I/O controlled by self CPU module and remote I/O module (Fail-soft disabled)	-
	User fatal fault	• User fatal fault detected	-
Changeover instruction by loader	Changeover between working and standby CPUs by loader		O
Multi-CPU relay switch	In the multi-CPU redundant system (1-to-1 mode), one CPU is switched due to a fault, followed by another CPU.		O

Notes: 1) Since the SX bus is a common resource for the entire system, changeover cannot be done between working and standby CPUs when faults occur in both CPUs.
2) An intentional stopping of the CPU is excluded from the conditions for changeover.

5-2-2 System performance in the redundant mode

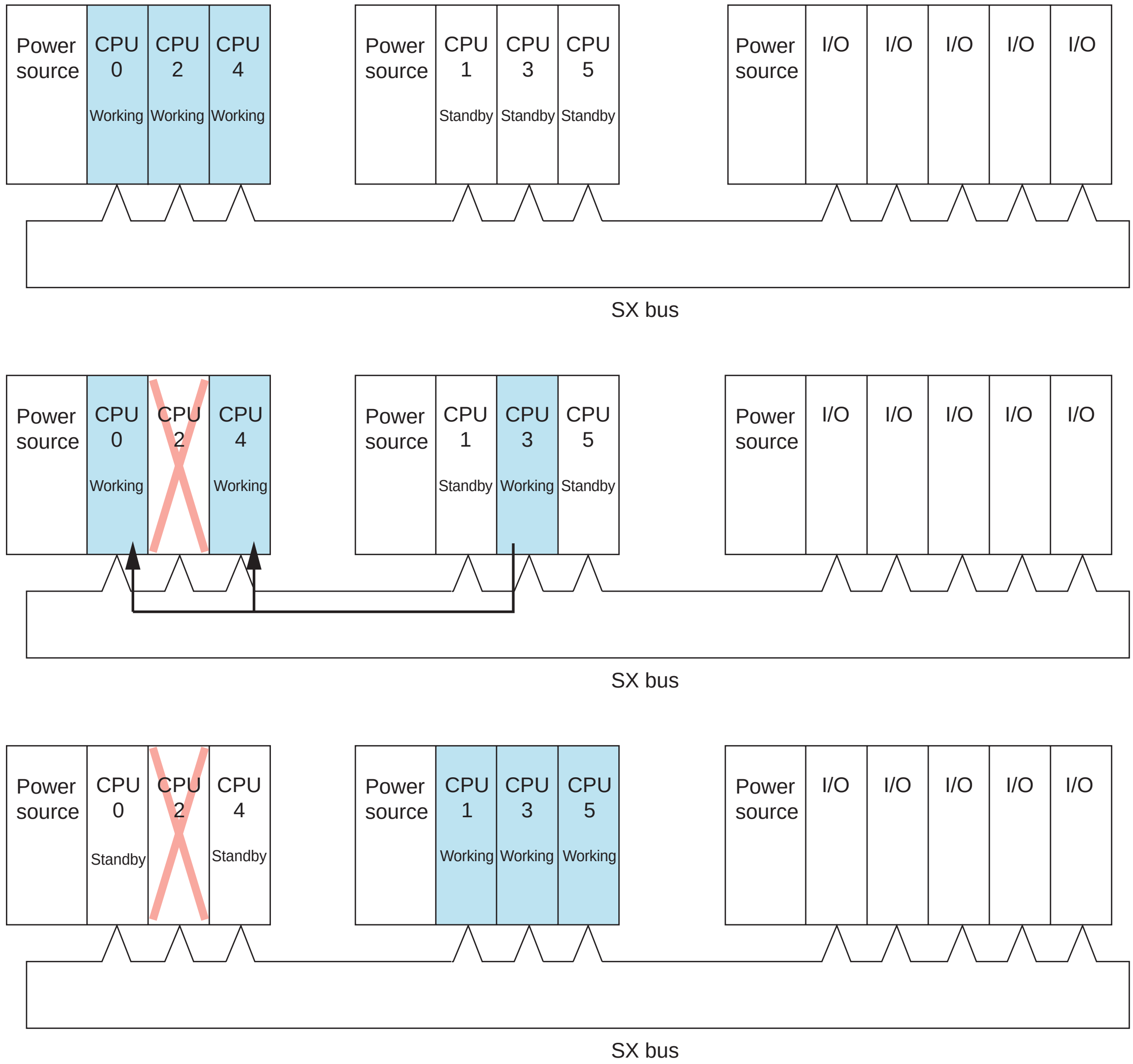
	Switching time	Takt time	Scan time
1-to-1 redundant mode cold standby	Within 130 ms (Note 2)	The same Takt time as in the ordinary multi-CPU system	The same scan time as in the ordinary multi-CPU system
1-to-1 redundant mode warm standby	Within 130 ms (Note 2)	Takt time in the ordinary multi-CPU system + 1 to 3 ms	Scan time in the ordinary multi-CPU system + several ms-several tens of ms (Note 1)
N-to-1 redundant mode cold standby	Several tens sec.	The same Takt time as in the ordinary multi-CPU system	The same scan time as in the ordinary multi-CPU system

Notes: 1) Depends on the quantity of equalized data.
2) If the message-related FB is used directly after changeover from the working CPU to the standby one, the busy status continues until the message closing process is completed (100 to 600 ms).

5-2-3 Multi-CPU relay switch

Usually, in the 1-to-1 redundant mode, changeover is done between a pair of working and standby CPUs. In the multi-CPU system, when you want to switch between another pair of CPUs following changeover between a pair of working and standby CPUs, set the relay switch mode. This enables the system to switch between both CPUs automatically.

<Example of three-pair relay switch system configuration in the 1-to-1 redundant mode>



In the above example, when CPU2 stops running due to a fault, CPU3 starts running instead. (The time for switching is 130 ms max.) Then, CPU3 issues the switch command to CPU0 and CPU4. CPU0 and CPU4, when receiving the command, go into the standby mode and CPU1 and CPU5 are switched to the working mode. (The time for switching is 130 ms max.)

Notes: 1) While one of the standby CPUs has a fault in the above system configuration, the system cannot switch to the working CPU if a fault occurs in the working CPU. The entire system goes down due to a fatal fault.

2) Relay switching can be done while both the working and standby CPUs are running normally.
For example, when changeover occurs during initialization (the working and standby sides are not undefined), even a CPU with an enabled relay switch may not be switched to another one. This means that both the working and standby CPUs run simultaneously in the CPU group for which the relay switch has been enabled.
Whether the relay switch has been operating normally can be determined by verifying that the resource running information (WSM48) bit, set to ON, matches that for the CPU group for which the relay switch has been enabled.

5-2-4 Data equalization

The 1-to-1 redundant warm standby configuration allows to make the internal data of the working CPU same as that of the standby CPU, which is referred to as data equalization. The specifications for data equalization are given below.

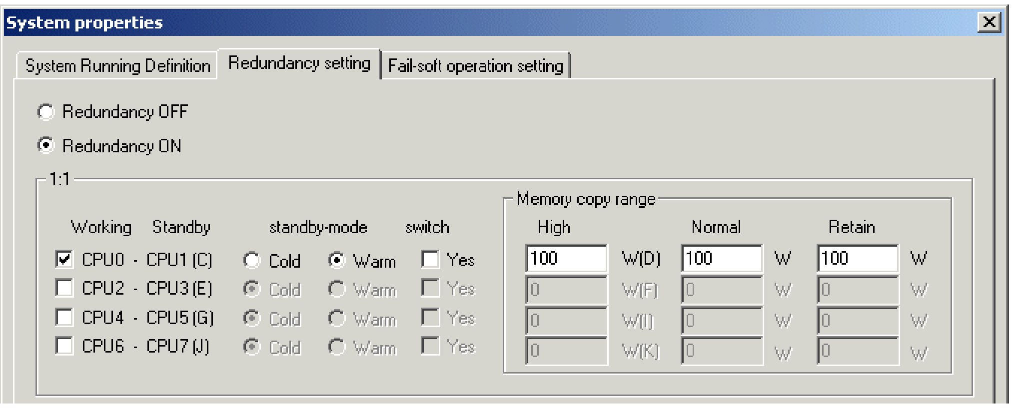
• User-specified equalized area	→	Standard memory (high-speed), standard memory and retain memory areas (depending on specifications (Note))
• System-equalized area	→	Devices defined for retain in the user FB Area with a retain attribute assigned in the system FB Current timer value area in the system FB

Note: For details, refer to <Setting size of user-specified equalized area> on the next page.

The size of equalized memory affects the default task start timings. When equalizing data in redundant systems, it is necessary to calculate the size of data to be equalized and check the default task start timings.

(1) User-specified equalized area

In principle, the user specifies the equalization area as the number of words from the beginning of the specified area.



The values set from Redundancy setting determine the user-specified equalization size.

Equalization size: S_{PG}

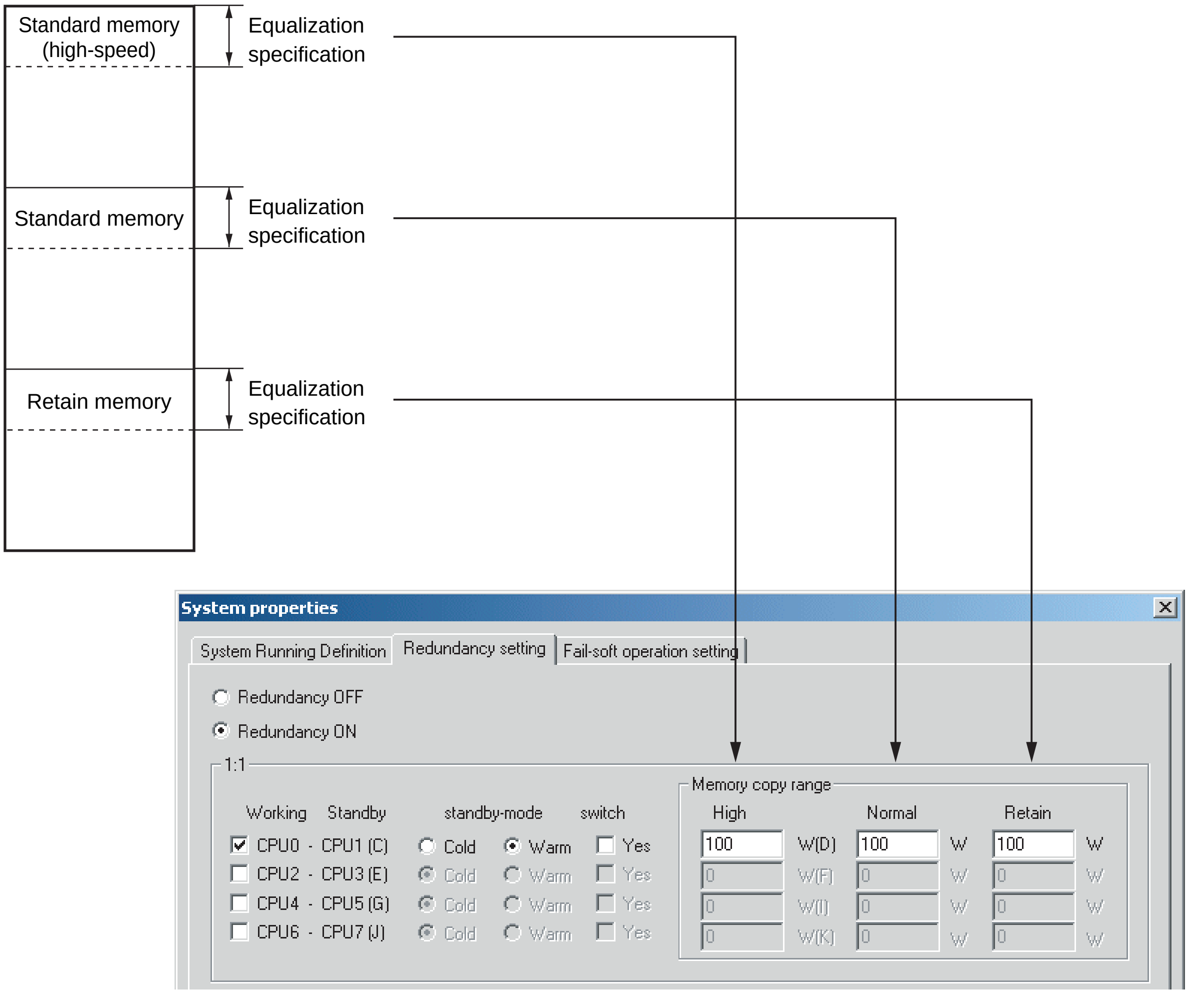
S_{PG} = (word count specified for high-speed memory) + (word count specified for standard memory) + (word count specified for retain memory) (words)

In the above example, the equalization size is as shown below.

(User-specified equalization size) = 100 + 100 + 100
= 300 (words)

<Setting size of user-specified equalized area>

Use the [Redundancy setting] window to set the user-specified equalized area. For each of the standard memory (high-speed), standard memory and retain memory, the equalization area with the word count specified from the [Redundancy setting] window is reserved from the beginning of the memory.



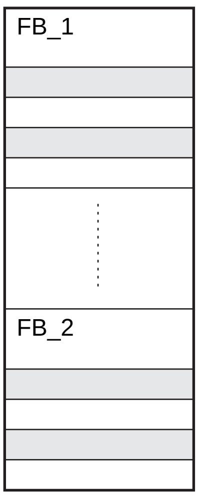
(2) System-equalized area

The system automatically equalizes the following areas:

- Memories defined for retain memory in the user FB
- Memories with a retain attribute assigned and current timer value area in the system FB

1) Devices defined for retain in the user FB

The devices in the user FB with the retain memory checked on the initial data list automatically become the equalization area.



: Devices defined for retain

<Device size>

- 16-bit data type device ⟶ 1 word

* From the above, calculate the total size of devices with the retain memory checked in the user FB.

Note: In the instance memories of all user FBs, the total size of areas that can be defined for retain memory is limited to 2048 words.

Equalization size: S_{UFB} (words) S_{UFB} = Devices with the retain memory checked

2) Memories with a retain attribute assigned and current timer value area in the system FB

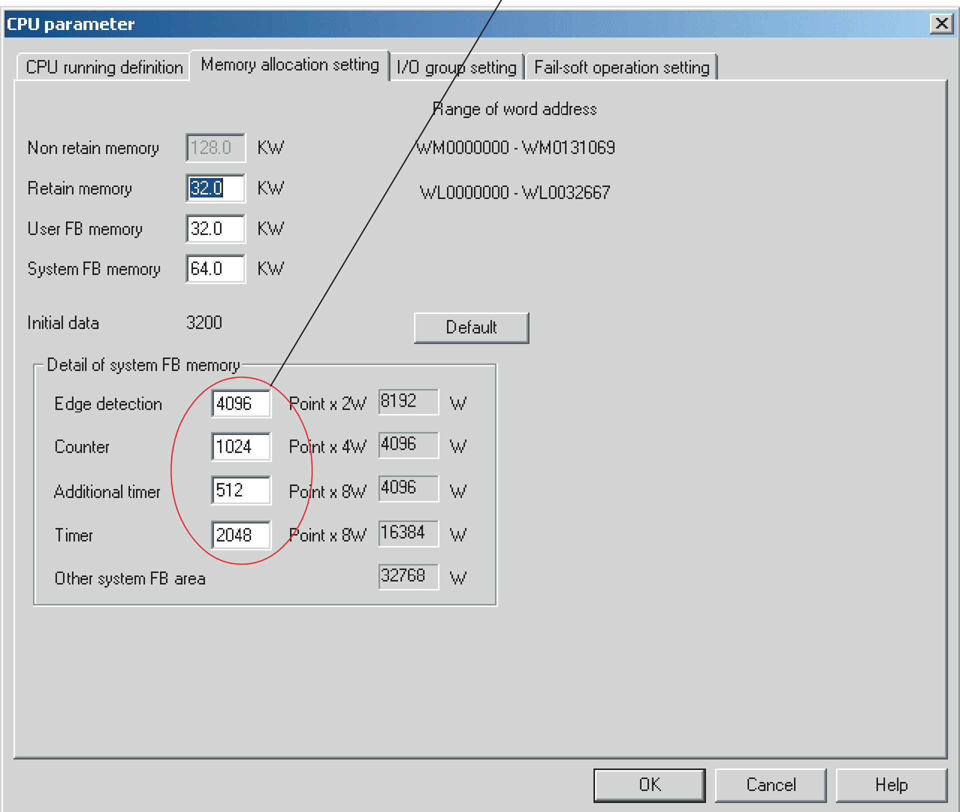
The instance memories with a retain attribute assigned in the user FB also automatically become the equalization area. The related FBs are listed below.

Equalized system FB	Equalized instance element	No. of words
Edge detection (R_TRIG, F_TRIG) [Rising edge contact, falling edge contact]	Old input value	2
Counter (CTU, CTD, CTUD, RCT)	Current counter value Old input value	2
Timer, totalizing timer (TP, TON, TOF, MR, TMR)	Current timer value, old input value, clocking flag	4

Equalization size S_{SFB}

S_{SFB} = No. of edge detection instructions   2 + No. of counter instructions   2 + No. of timer instructions   4
(Note 1)

Notes: 1) Includes a totaling timer.
2) The number of equalized instructions indicates the number of instructions available in each FB defined in the [Memory allocation setting] window on the [CPU parameter] dialog instead of that in the program.



(3) Equalized data size

The size of data that can be equalized is the sum of words calculated in (1) and (2) above. It is limited to 8192 words in a 1-to-1 redundant system with one pair.

$8192 >= S_{PG} + S_{UFB} + S_{SFB}$ (words)

Note: When using a high-performance CPU of 74K or higher in 1-to-1 redundant warm standby configuration, be sure to consider the equalization data in memory definitions. In memory definitions by default, equalization of system FB memories will require more than 8192 words.

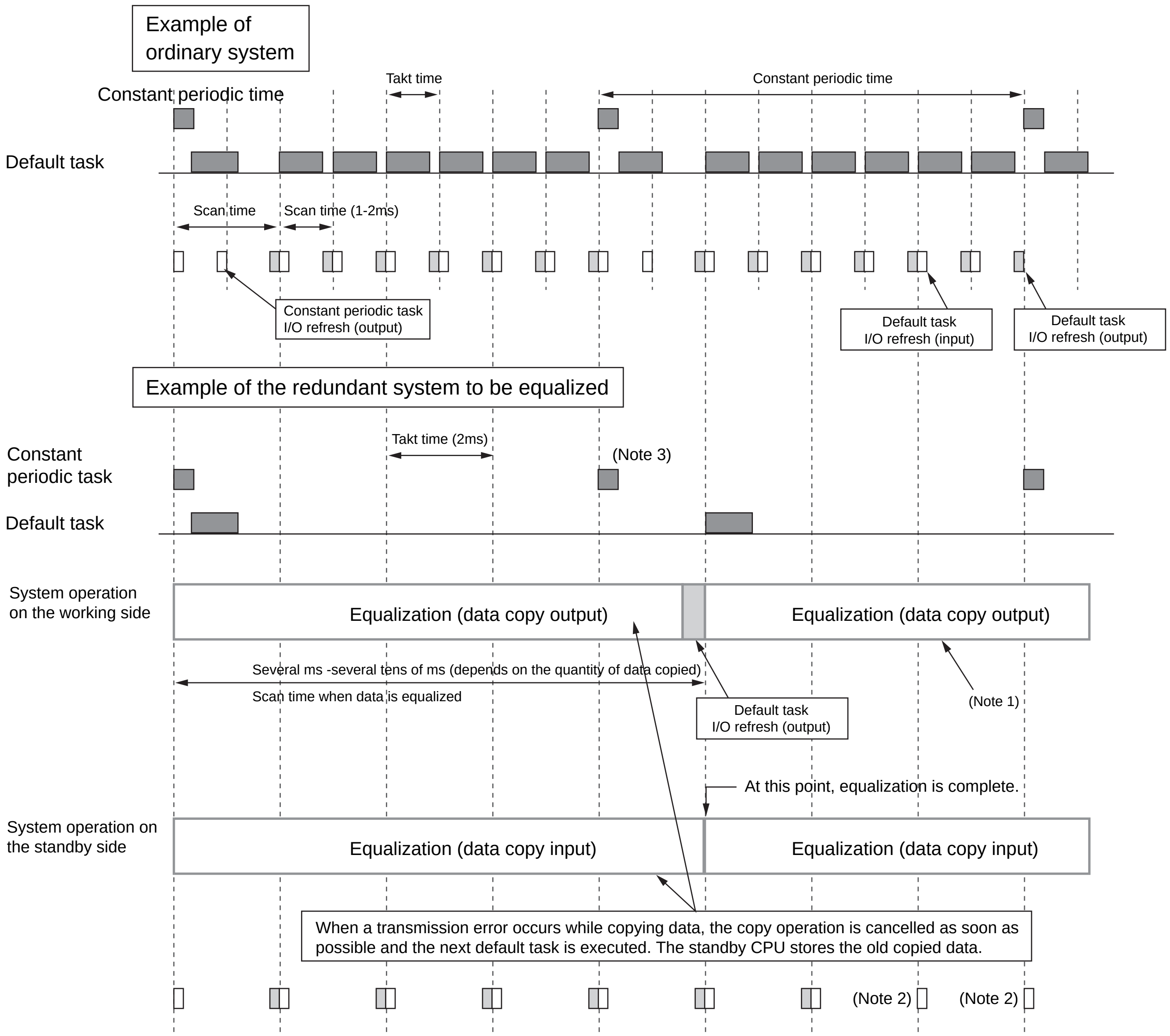
(4) Data equalization

After calculating the size of equalized data, the CPU scan time can be calculated. Because data equalization takes place after execution of the default tasks, the size of equalized data affects the default task execution intervals (scan time). The scan time can be calculated in the expression given below.

Scan time for redundant system

$$= (\text{scan time for ordinary system}) + \underbrace{(1 + \frac{\text{total number of words equalized}}{512 \text{ words}}}_{\text{Fractions truncated}}) \times \text{takt time (ms)}$$

<Timing for equalization>



Notes: 1) When a changeover event occurred during data copy, the standby CPU starts operation using the data equalized in the previous copy.

2) When a changeover event occurs, data output stops from the next takt. The I/O module output is not updated before the standby CPU starts operation.

3) A fixed task can be set to a time shorter than the scan time. However, because equalization processing is performed after completion of default tasks, equalization may be disabled for memories used by tasks with cycle shorter than the scan time (default task execution timing).

5-2-5 Memory operation at changeover between working and standby CPUs

System status	Memory or flag	1-to-1 redundant mode		N-to-1 redundant mode
		Cold standby	Warm standby	
System power-on (warm running)	Standard memory	Cleared		
	Retain memory	Old retained value		
	I/O memory	Reset hold specification		
	Default task start flag	ON		
	Initial flag	OFF		
Working ⇨ Standby	Standard memory	Values are retained during running (Memory operation and flag operation are not performed.)		
	Retain memory			
	I/O memory			
Standby ⇨ Working	Standard memory	Cleared	An area other than the area to be equalized is cleared, and the data in the area to be equalized is inherited by the working CPU.	Cleared
	Retain memory	Old values are retained.	Old values are retained and data in the area to be equalized is inherited by the working CPU.	Cleared
	I/O memory	Data remains unchanged	Data remains unchanged	Reset/hold
	Default task start flag	ON	ON	ON
	Initial flag	OFF (Note 1)	OFF (Note 1)	ON
	Working/Standby changeover flag SM460	ON	ON	ON
Cold running (at program download or initial start-up by loader)	Standard memory	Cleared		
	Retain memory	Cleared		
	I/O memory	Cleared		
	Default task start flag	ON		
	Initial flag	ON		

Notes: 1) When the standby CPU does not run after program download, the initial flag is set to ON.
2) In the redundant system, to pass data to the bit devices used for semaphores at changeover between the working and standby CPUs:
- Assign the bit devices for semaphores to the standard memory area to avoid equalization. (They are reset to 0 at changeover.)
- When changeover occurs, make an attempt to get semaphores again.

Key points

<System operation at changeover>

- 1-to-1 redundant mode standby system
  The system operates in the same manner as the system for single CPU warm running with the exception that data in I/O memory is inherited.
- 1-to-1 redundant warm standby system
  This is the cold standby system with data equalization added.
- N-to-1 redundant system
  Since the program is downloaded from the memory card interface module, the system uses the cold standby method.

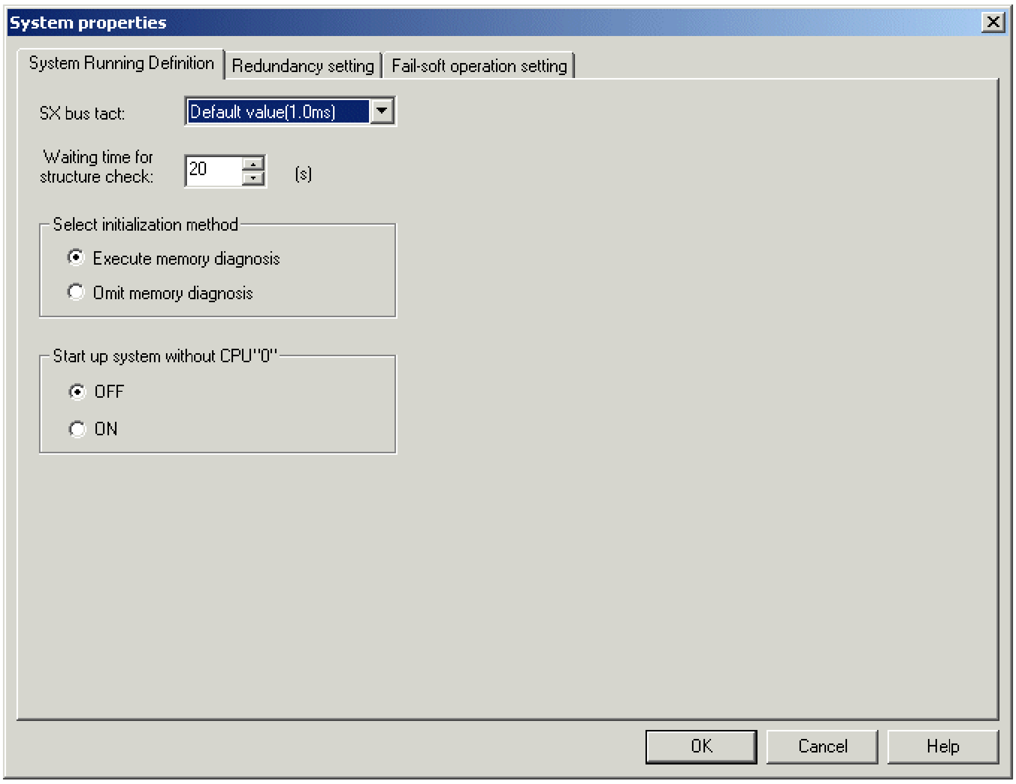
O : ON, - : OFF, △: Blinking

Faulty module / Module			The entire system operates normally	The entire system stops normally	Fatal fault in working CPU module	Fatal fault in standby CPU module	Fatal fault in working CPU module / Application error	I/O fault (fail-soft enabled) / Non-fatal fault in working/standby CPU resource	I/O fault (fail-soft disabled) / Fatal fault in working/standby CPU resource
Working CPU	LED indication	ON	O	O	-	O	O	O	O
		ERR	-	-	O	-	-	-	-
		RUN	O	-	-	O	-	O	-
		ALM	-	-	O	O	O	O	O
	Flag		Normal run	Normal run	Fatal fault in self-CPU resource	Fatal fault in standby CPU resource	Fatal fault in self-CPU resource	Fault in I/O module Non-fatal fault in self-CPU resource	Fault in I/O module Fatal fault in self-CPU resource
Standby CPU	LED indication	ON	O	O	O	-	O	O	O
		ERR	-	-	-	O	-	-	-
		RUN	△	-	O	-	O	△	-
		ALM	-	-	O	O	-	O	O
	Flag		Normal run	Normal run	Fatal fault in working CPU	Fatal fault in self-CPU	Fatal fault in working CPU	Fault in I/O module Non-fatal fault in self-CPU resource	Fault in I/O module Fatal fault in self-CPU resource
System DO	LED indication	ON	O	O	O	O	O	O	O
		ERR	-	-	-	-	-	-	O
		System DO	O	-	O	O	O	O	-
	Contact output		ON (Running)	OFF (Stop)	ON (Running)	ON (Running)	ON (Running)	ON (Running)	OFF (Stop)

5-4 System Startup with no CPU 0 Station

In a 1-to-1 redundant system, you can configure whether the system will start up when there is no CPU 0 station upon system startup, power down, not mounted, fault, or the like.

◆ On the [System Running Definition] window of the [System properties] dialog in the system definitions, set [Start up system without CPU “0”].



Notes: 1) When starting up the system without a CPU 0, configure all CPUs with software version 63 or higher.
2) When turning on the power, after checking for a CPU 0, other CPUs start up as an SX bus master CPU. For this reason, system startup will take 30 to 40 seconds longer than when there is a CPU 0 upon power-up.
3) If a project configured to start the system without a CPU 0 is downloaded to a CPU with software version below 63 via a user Rom card (CF card), it will operate in “do not start up system without CPU 0” mode.
4) Startup without CPU 0 is not possible in an N-to-1 redundant system.
5) If an error occurs reading system definitions (application ROM error), the system automatically operates in “start up system without CPU 0” mode.
6) If all CPUs are faulting, it will take up to 60 seconds longer to complete initialization.
Additionally, the ALM LED will flicker during this period.
7) The SX bus is a common resource for the system as a whole. For this reason, the system will not start up when there is a hardware fault in the SX bus circuit.
8) The SX system operates based on the system definitions. For this reason, the system will not start up if the system definitions are inconsistent with the actual system (error).

Appendix 1 Setting High-performance CPU Takt Periods

Appendix 1 Setting High-performance CPU Takt Periods

On the SX bus, data is exchanged between the CPU and I/O modules in synchronization with the Takt period. The application program on the CPU module performs 1) I/O data input updating, 2) arithmetic operations, and 3) I/O data output updating in execution units called tasks (default task, periodic task, and event task). These operations are carried out concurrently with data exchange over the SX bus.
This appendix introduces approximation formulas for calculating the scan time for each takt period. Basically, the Takt period of the SX bus is dependent on the system configuration. In a system that requires a task period which is based on the Takt period, the Takt period depends on the number of steps of the application. To obtain an exact execution time, it is necessary to measure it on an actual machine.

<System configuration that depends on the Takt period>

- Number of I/O points
- Number of CPUs
- Number of remote I/O master modules
- Number of communication modules

(1) Approximation formulas for calculating the Takt period from the system configuration data

Takt period T (µs)

1) 1 CPU + direct I/O configuration: T = Tb [Base time (Tb) based on the number of direct connection input/output points]

No. of direct connection I/O points (points)	0	32	128	256	512	1024	2048	3072	4096	6144	8192
Base time Tb (µs)	418	504	507	510	556	695	1042	1388	1520	1711	1911

The above numbers are based on the assumption that the ratio of I/O input to I/O output is 1 to 1. The more outputs there are, the longer the base time is, and vice versa. The range of fluctuation is approximately ±20%.

Notes: 1) 0.5 ms takt period is possible under the condition that there is a single CPU, the number of direct connection I/O points is 256 or less, and no communication module is used.
2) The user can select a Takt period from 0.5, 1, 1.5, 2, 2.5, 3, ..., 10 ms. The user should select a value that is obtained by rounding up the corresponding value listed in the above table.

2) Multi-CPU configuration: [No. of CPUs: n]

- T= Tb + 210n (No. of direct connection input/output points: 2048 or less)
- T= Tb + 200n + 190 (No. of direct connection input/output points: more than 2048)

3) Single CPU + remote I/O: [No. of remote I/O master modules: m]

- T= Tb + 250m + 430 (No. of direct connection input/output points: 2048 or less)
- T= Tb + 280m + 730 (No. of direct connection input/output points: more than 2048)

4) Multi-CPU + remote I/O: [No. of CPUs: n, number of remote I/O master modules: m]

- T= Tb + 340n + 200m + 400 (No. of direct connection input/output points: 2048 or less)
- T= Tb + 405n + 260m + 340 (No. of direct connection input/output points: more than 2048)

5) When communication modules are added to 1 CPU + direct connection I/O (1): [No. of communication module: p]

- T = Tb + 40p + 250

6) When communication modules are added to configurations ((2) to (4)): [No. of communication modules: p]

- T = (Time calculated for ((2) to (4)) + 85p [when there is no remote master module]
- T = (Time calculated for ((2) to (4)) + 128p [when there is a remote master module]

Note: 3) 2048 points / 1 line is assumed for remote I/O.

Appendix 1 Setting High-performance CPU Takt Periods

(2) Formula for calculating the performance when scanning by the Takt period

Computing time
= [Takt period] - [SBM overhead time (200µs)] - [I/O refresh time] - [POU control time]

- I/O refresh time = (2n + m + 60) µs <n: number of I/O modules, m: total number of I/O words>

- POU control time = PG control time + user FB control time + user FCT control time
  = (4a + 6b + 7c) µs
  <a: No. of PGs, b: No. of user FB calls, c: No. of user FCT calls>

Computing time
= [Takt period] - (2n + m + 60) - (4a + 6b + 7c) µs

No. of program steps = [computing time / single instruction execution time / 1024] k steps
No. of program steps = [computing time / 20.48] k steps <when single instruction execution time = 20ns>
= [computing time / 61.44] k steps <when single instruction execution time = 60ns>

Notes: 1) Refer to Appendix 7, "Instruction Processing Speed Chart," for the execution time of the individual instructions.
2) The instruction execution time varies depending on the time required to access the memory to which variables to be processed are assigned. Consequently, it is necessary to add the following access time increments to the instruction execution time according to the number of variables that the instruction accesses.
The memory access times are calculated as follows:
1) I/O memory and standard memory (high speed): Base time (20ns)
2) Standard memory, retained memory, user FB memory, system FB memory, system memory: Add 40ns.
3) Memory in another CPUs access via the processor bus: 3ms

(3) Sample time calculations

1) Single CPU

Scan Time (Takt Time)	System Configuration	POU Control Time (No. of PGs/FBs/FCTs)	Program Executable Time (in 20 ns steps)
0.5ms	CPU... 1 module (communication module disabled) Direct connection I/O: 256 points	68µs (4/4/4)	124µs (6k steps)
1ms	CPU... 1 module Direct connection I/O: 1024 points	136µs (8/8/8)	412µs (20k steps)
2ms	CPU... 1 module Direct connection I/O: 2048 points	480µs (16/32/32)	876µs (42k steps)
	CPU... 1 module Remote I/O ...1 module: 2048 points	480µs (16/32/32)	876µs (42k steps)
	CPU... 1 module Remote I/O ...2 modules: 4096 points Direct connection I/O: 2048 points	480µs (16/32/32)	492µs (24k steps)

2) Multi-CPU configuration

Scan Time (Takt Time)	System Configuration	POU Control Time (No. of PGs/FBs/FCTs)	Program Executable Time (in 20 ns steps)
4ms	CPU... 4 modules Remote I/O ... 2 modules: 4096 points Direct connection I/O: 2048 points	480µs (16/32/32)	2492µs (484k steps = 121k x 4)
	CPU... 4 modules Remote I/O ... 2 modules: 4096 points Direct connection I/O: 1024 points Communication modules... 2 modules	480µs (16/32/32)	2620µs (508k steps = 127k x 4)

(4) Estimation of the Takt periods in the 1-to-1 warm standby redundant system

In the 1-to-1 warm standby system, the Takt time is longer than that in the ordinary multi-CPU system. This is because, in this system, equalized data should be transferred between the operating and waiting CPUs. The estimate expression is described below. In the expression, the large Takt times are used for redundant system Takt time 1 (T_{R1}) and redundant system Takt time 2 (T_{R2}).

Redundant system Takt time 1: T_{R1} [µs]

T_{R1} = Usual Takt time note) + 596 x N + 430 (No. of directly connected I/Os: 2048 or less)
T_{R1} = Usual Takt time note) + 626 x N + 730 (No. of directly connected I/Os: more than 2048)
N: No. of pairs in the redundant system

Note: The usual Takt time is the time found by expression (1). The number of CPUs can be calculated using the number of CPU pairs in the redundant system.

Redundant system Takt time 2: T_{R2} [µs]

T_{R2} = (I/O refresh time) + T_{DMA} + T_{CPY} + 200 [µs]

- I/O refresh time : (2n + m + 60) [µs] [n: No. of I/O modules, m: Total No. of I/O words]
- T_{DMA} = [(No. of SX bus modules excluding CPUs) + (No. of CPU modules) x 2 + (No. of remote master modules x 55)
  + (No. of total words for all connected/remote I/Os) + 512] x 0.5 [µs]
- T_{CPY} = (No. of words for equalized variables in high-speed, standard, retain memory areas) x 0.3
  + (No. of words for user retain variables) x 0.35
  + (No. of edge detection instructions and counter instructions) x 0.3
  + (No. of timer instructions) x 0.45
  + 10 [µs]

Appendix 2 Setting Standard CPU Takt Periods

Appendix 2 Setting Standard CPU Takt Periods

On the SX bus, data is exchanged between the CPU module and I/O module in synchronization with the Takt period. For the application program on the CPU module, I/O data input update, arithmetical operations, and I/O data output update are performed in each task (process unit) in parallel with data exchange on the SX bus.
The standard CPU divides system software in one Takt period into the processes executed in every Takt period as well as those executed whenever an application is executed or a default task has been done. The loader process is performed in one excessive Takt period. The estimate expression of the scan time for each Takt period is shown below. The Takt period depends on the system configuration. Execution of system software also depends on the Takt period. Thus, the executable Takt periods must be set by system software.

<System configuration components depending on the Takt period>

- No. of I/O points
- No. of remote I/O modules
- No. of stations simultaneously issuing loader command

The Takt period can be selected among 1, 2, 3, ..., 19, 20ms. (1) The Takt period calculated is based on SX bus performance (No. of I/O modules) and (2) the necessary Takt period based on the run time of system software, whichever is larger, is determined by truncation

(1) The Takt period calculated is based on SX bus performance

Takt period T (μs)

1) One CPU + Connected I/O : T = Tb

No. of connected I/O points	0	32	128	256	512	1024	2048	3072	4096	6144	8192
Base time Tb (μs)	418	504	507	510	556	695	1042	1388	1520	1711	1911

The times listed above are calculated under the condition that the ratio of I/O input to output = 1 : 1. The time increases with an increased number of outputs and decreases with a reduced number of outputs. Its fluctuation ranges from + 20% to -20%.

2) One CPU + remote I/O mater: [No. f remote I/O masters]

- T = Tb + 250 m + 430 (No. of direct connection input/output points: 2048 or less)
- T = Tb + 280 m + 730 (No. of direct connection input/output points: more than 2048)

Note: 2048 points/1 line is assumed for remote I/O.

(2) The necessary Takt period based on the run time of system software

System software processes I/O transmission, tasks, and loader commands. Any of these processes should be done in one Takt period. Based on the number of stations simultaneously issuing the loader commands, recommended preset times and calculation times (the application program processing time) for the Takt periods are shown below.

Recommended Takt time (ms)	1	2	3	4	5	6	7	8
No. of stations simultaneously issuing loader commands	1	4	16	27	27	27	27	27
Calculation time (μs)	409	1209	2009	2809	3609	4409	5209	6009

Note: The stations simultaneously issuing loader commands include the loader, PODs, and the modules associated with message-related instructions (if the module has two ports, they are both counted.)

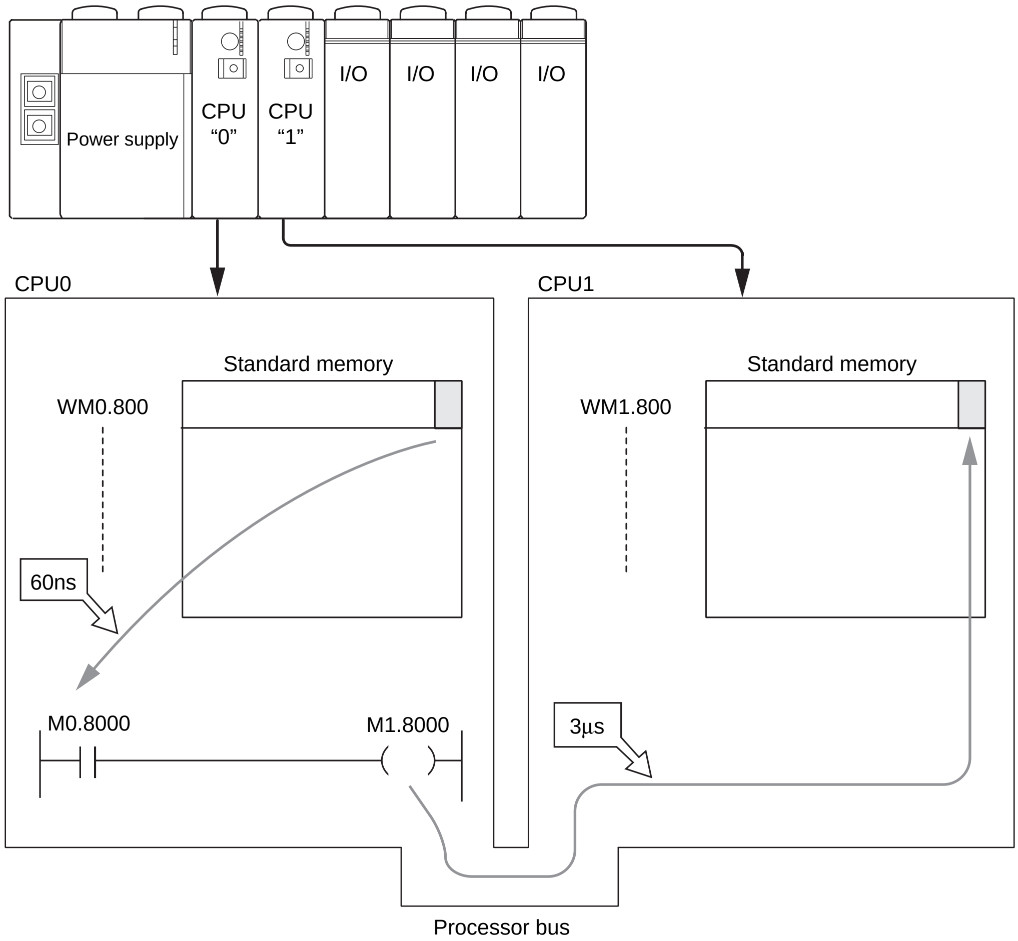
Key points

- To execute the task processes in synchronization with the Takt periods, the application program should be written so that it does not use more than the calculation time listed below.
- Compared with the recommended Takt time, even when the number of stations simultaneously issuing the loader commands is smaller, the calculation time does not vary. (The responsibility to the loader commands improves.)
  Compared with the recommended Takt time, when the number of stations simultaneously issuing the loader commands is larger, the calculation time is reduced. (The responsibility to the loader commands deteriorates.)

Appendix 3 Accessing the Processor Bus

Appendix 3 Accessing the Processor Bus

The processor bus can be used to read and write memory between CPUs in the multi-CPU system and to read and write memory between the CPU and the P/PE-link memory.



<Access time to each memory from an application (CPU)>

The access time to each memory in or out of the CPU is shown below.

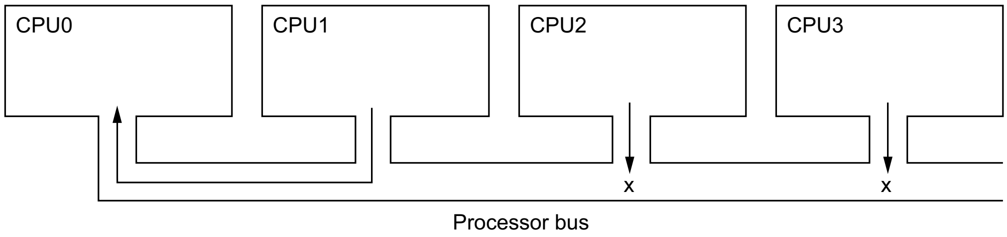
Accessed memory	Access time (/word)
High-speed memory in self-CPU (%MW1.0 to %MW1.2047)	20ns
Memory other than those in self-CPU	60ns
I/O area (X/Y)	20ns
Memory in any other CPU in a multi-CPU system (Note 1)	3µs (Note 2)
P/PE-link memory	3µs (Note 2)

Notes: 1) The high-speed memory area cannot be accessed in any other CPU.
2) The access time through the processor bus.

Appendix 3 Accessing the Processor Bus

<Considerations in reading /writing memory through the processor bus>

The processor bus cannot be accessed simultaneously by more than one CPU. If more than one CPU accesses the processor bus simultaneously, it can be sequentially used by the CPUs starting from the one with the highest priority assigned. The CPU with a lower priority waits for a long time. Any such delay affects the CPU processing speed.

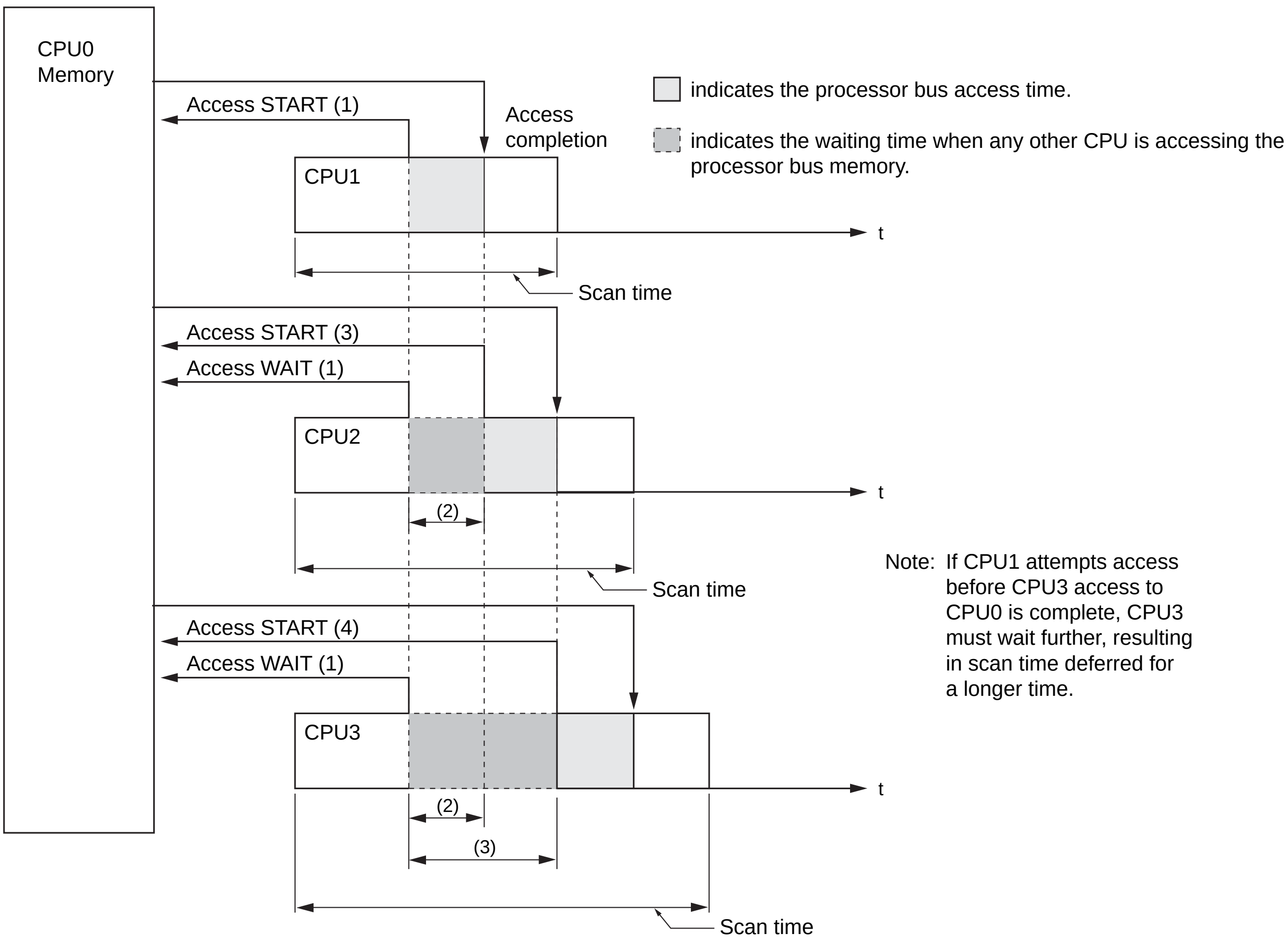


Note: The priority is determined in the ascending order of numbers, with the highest priority assigned to the smallest one.

(Example)

In the example below, a multi-CPU system consisting of four CPUs is shown. With reference to this example, the relationship between processor bus access and the CPU scan time is described.

1) When CPU1, CPU2, and CPU3 simultaneously access CPU0 memory, the first CPU1 with the highest priority has initial access to it. CPU2 and CPU3 are kept waiting.
2) The scan times for CPU2 and CPU3 are delayed by the CPU1 access time.
3) When CPU1 access is complete, the CPU2 with the secondly highest priority tries to gain access to CPU0 memory. CPU3 must wait further.
4) When CPU2 access is complete, CPU3 starts gaining access to memory. (Note)



Key point

- In the multi-CPU system and P/PE-link system, design the system to minimize the number of accesses to the processor bus. The recommended number of accesses is 128/ms.

Appendix 4   Procedures for Creating User FBs

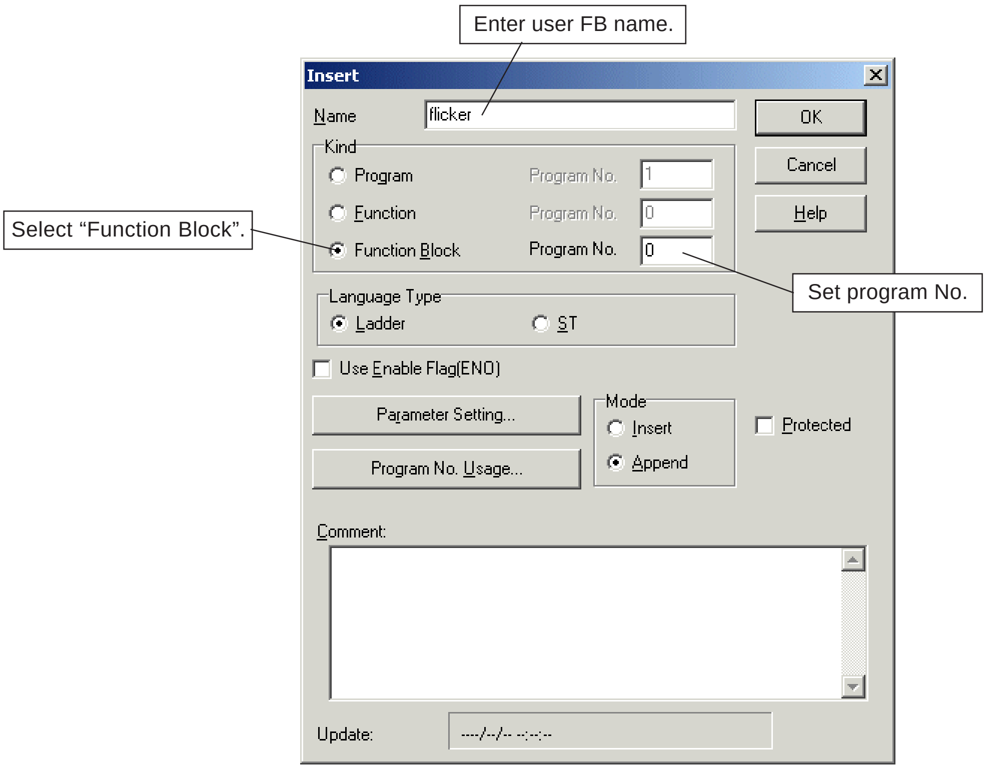
Appendix 4 Procedure for Creating User FBs

This section explains how to create and use user FBs, using an example of creating an FB that outputs flicker signals.

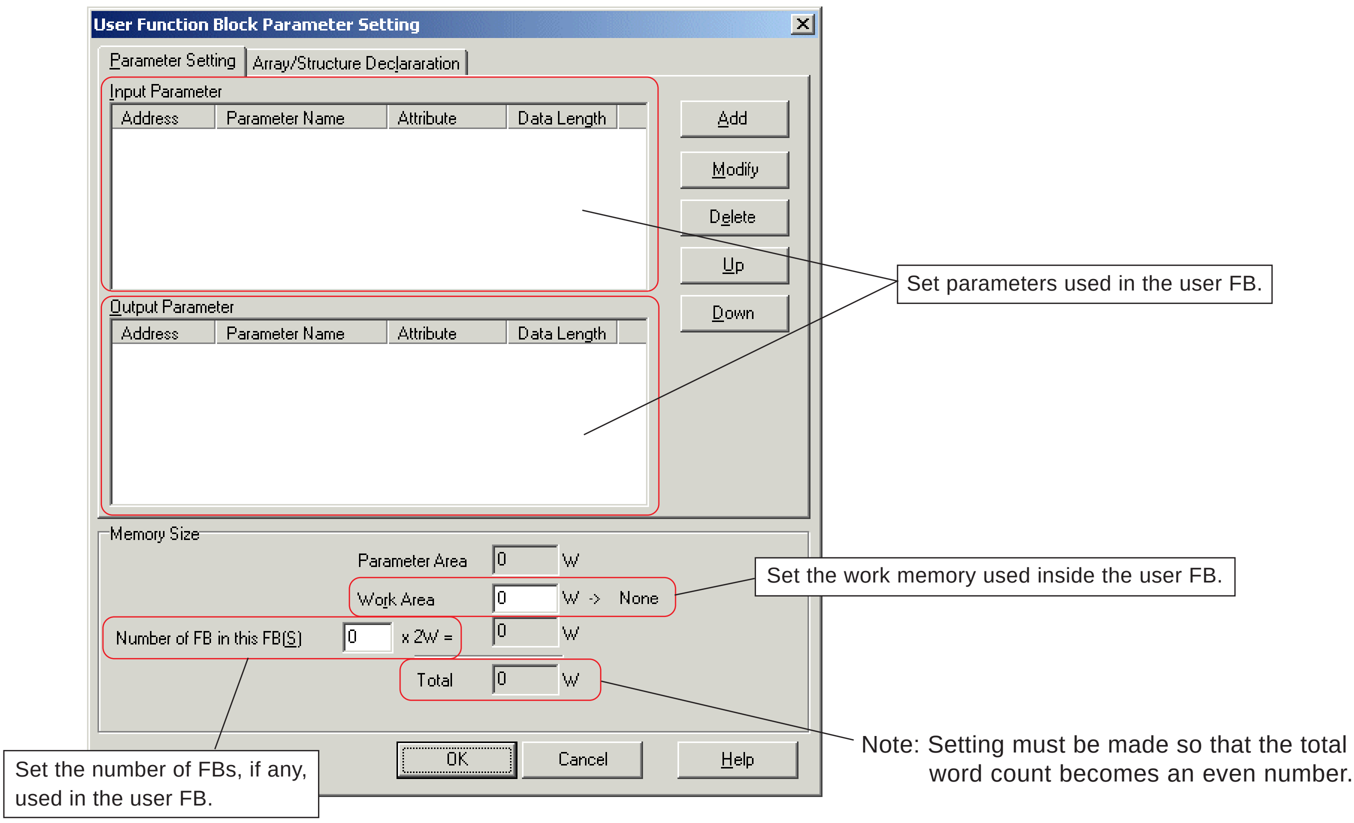
(1) Creating procedure

◆ Add a program for a user FB. Right-click the [Programs] icon or an existing program in the project tree and execute the [Insert] command from the popup menu. The [Insert] dialog appears.

◆ Select “Function Block” for “kind” and enter a user FB name and program No.

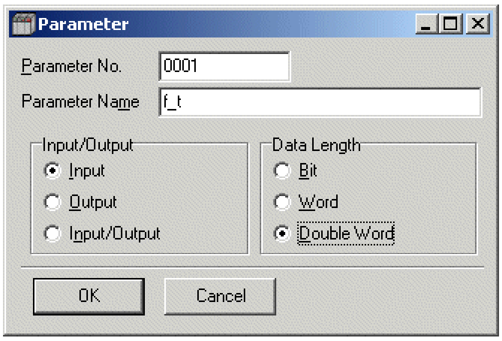


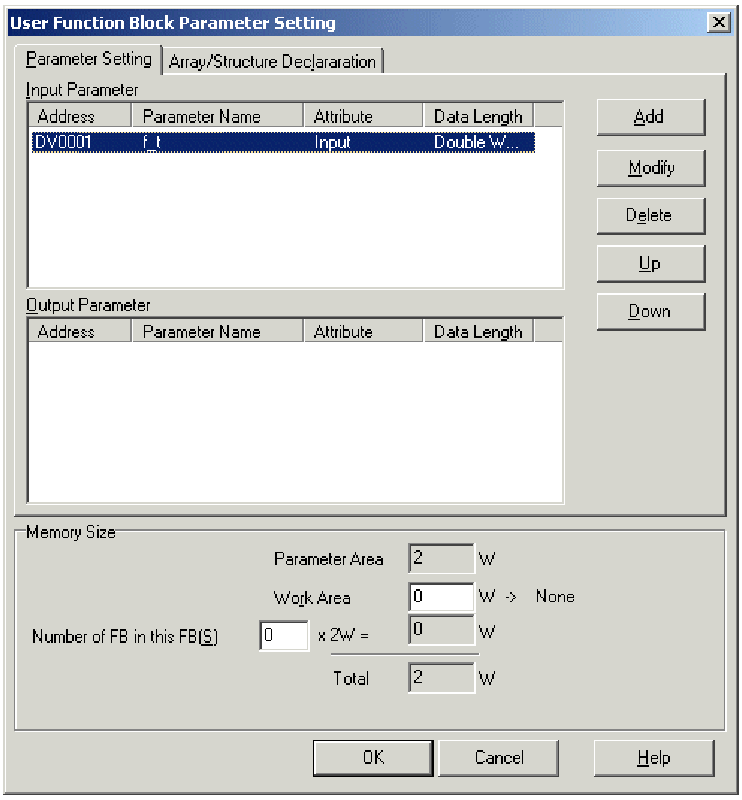
◆ Next, set the user FB parameters. Click the [Parameter Setting...] button to display the [User Function Block Parameter Setting] dialog.



Appendix 4 Procedure for Creating User FBs

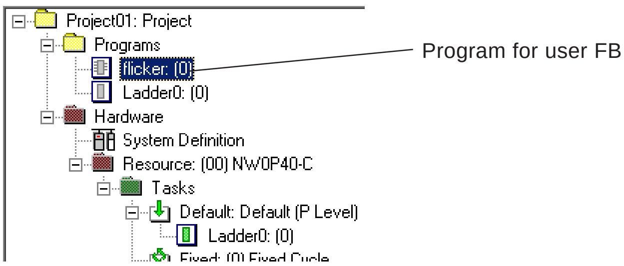
◆ Set the parameters. Clicking the [Add] button on the dialog displays the [Parameter] dialog. Set the parameter No., parameter name, input/output type and data length, and then click the [OK] button to set the parameter.





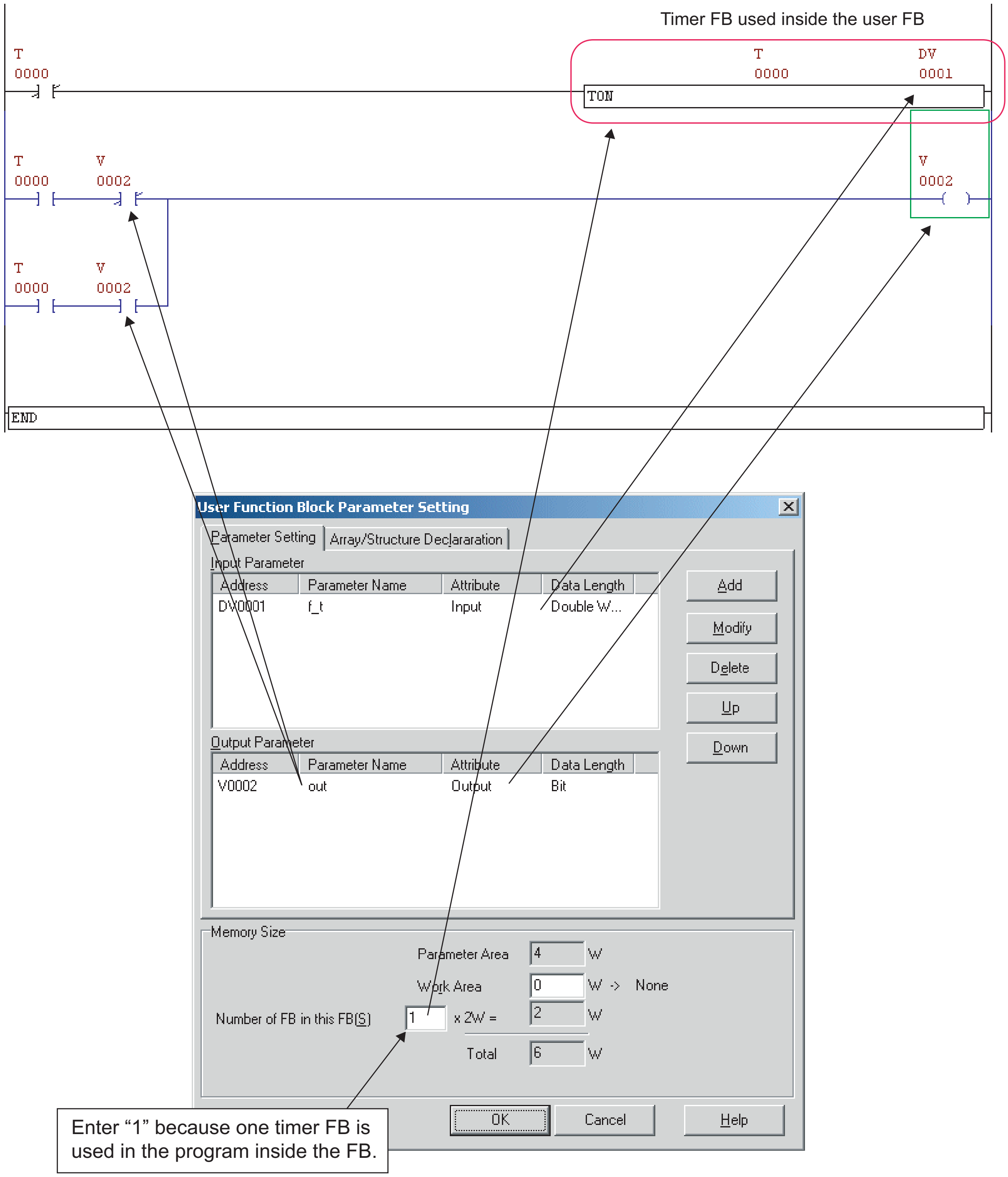
◆ In the same manner, set other necessary parameters.

◆ After setting all necessary items, click the [OK] button to return to the [Insert] dialog. Clicking the [OK] button on the [Insert] dialog inserts the program for the user FB.



Appendix 4 Procedure for Creating User FBs

◆ Next, create a program. If necessary, also set the No. of system FBs used or work areas.



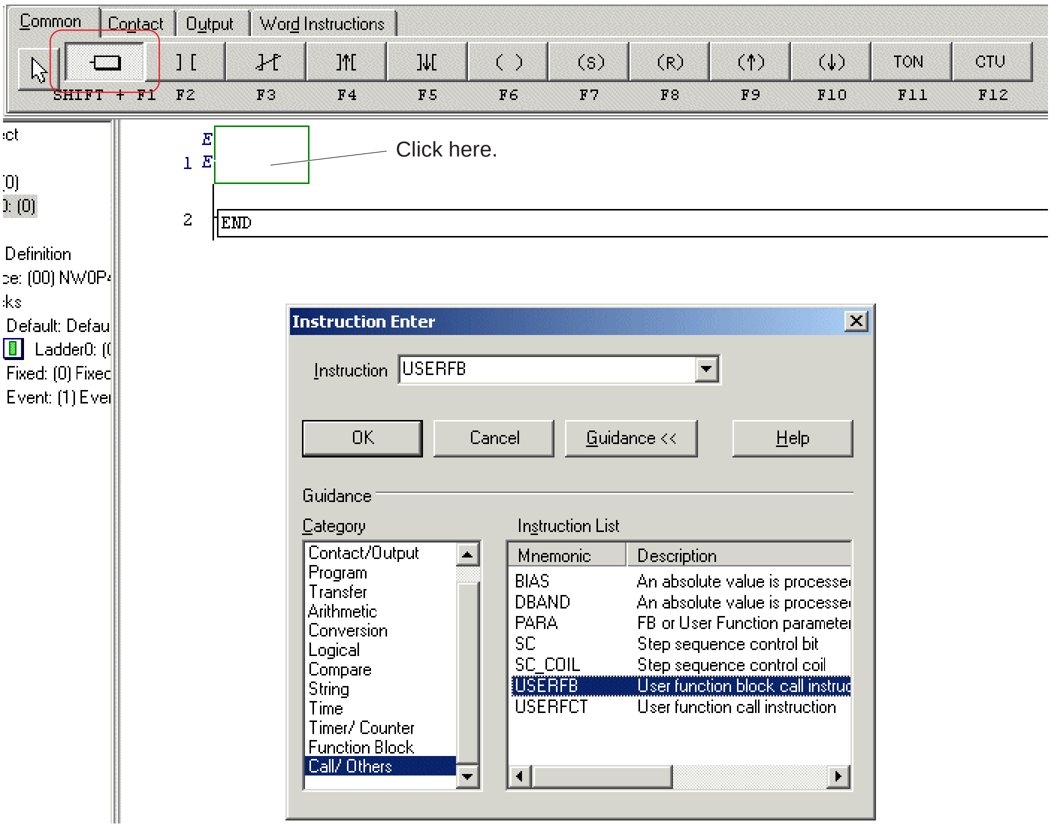
<Operation of FB>

Specify the flicker ON/OFF pulse time in input parameter “DV0001”. The flicker signals are output to output parameter “V0002”.

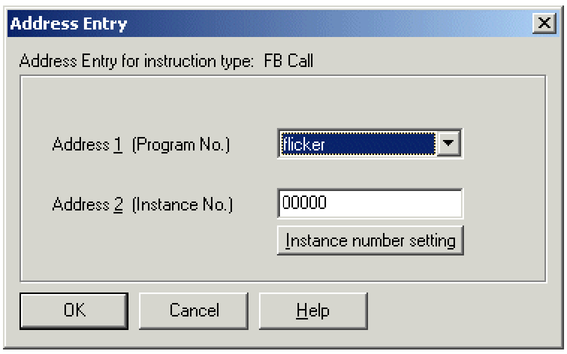
(2) Calling user FBs

This paragraph explains how to use the user FB in a program of the project where the FB was created.

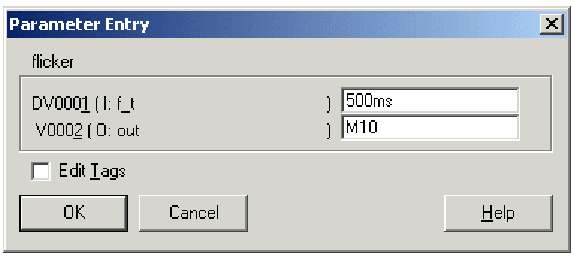
◆ Click the position where you want to insert the user FB with the [Any Instruction] button selected. The [Instruction Enter] dialog appears.



◆ On the [Instruction Enter] dialog, select [Call/Others] from the category and [USERFB] from the instruction list, and then click the [OK] button. The dialog shown below appears. After selecting a user FB to be used and setting the instance No., click the [OK] button.

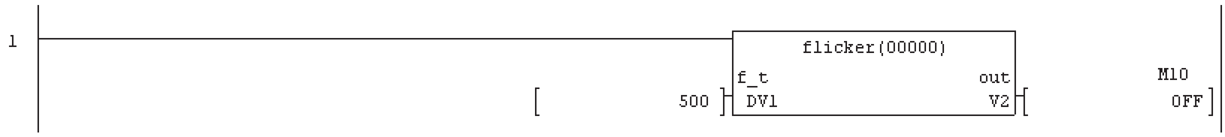


◆ The [Parameter Entry] dialog appears. Enter a device or a constant for each parameter, click the [OK] button.



Appendix 4 Procedure for Creating User FBs

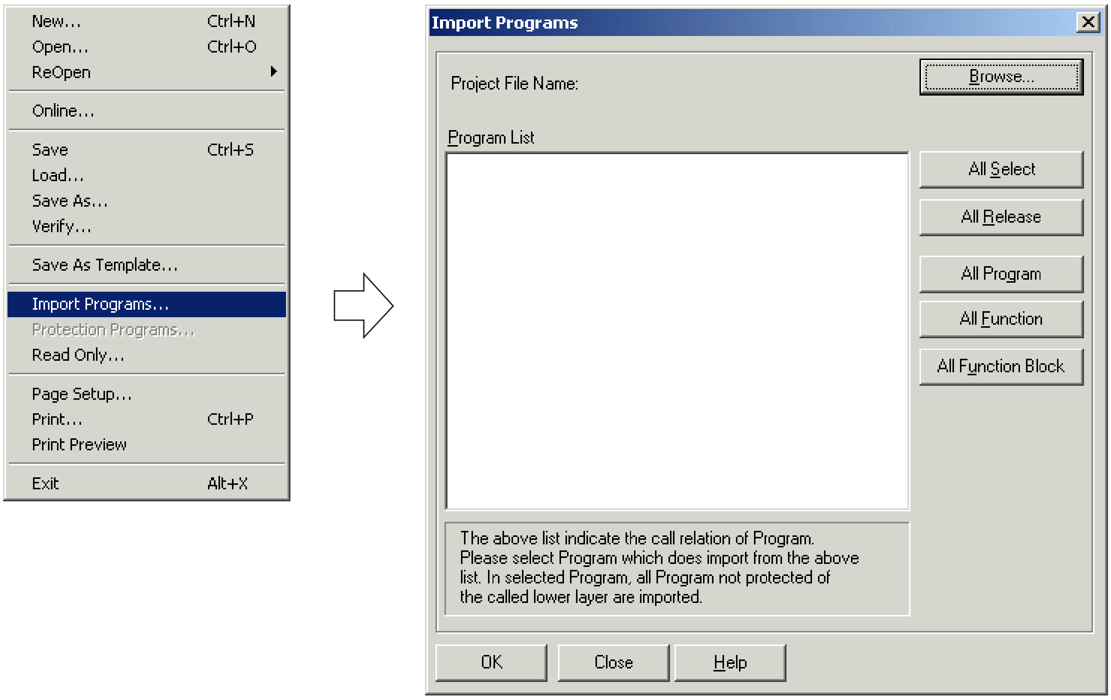
◆ The user FB is displayed as shown below.



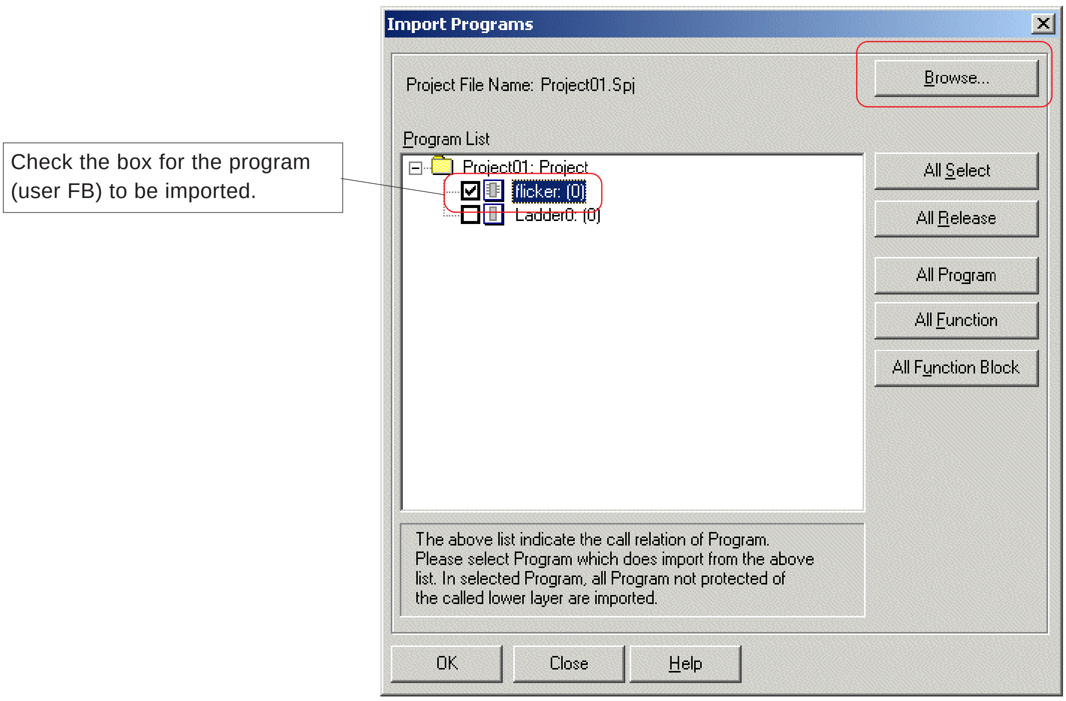
(3) Using user FBs created in other project

This paragraph explains how to use user FBs created in other project.

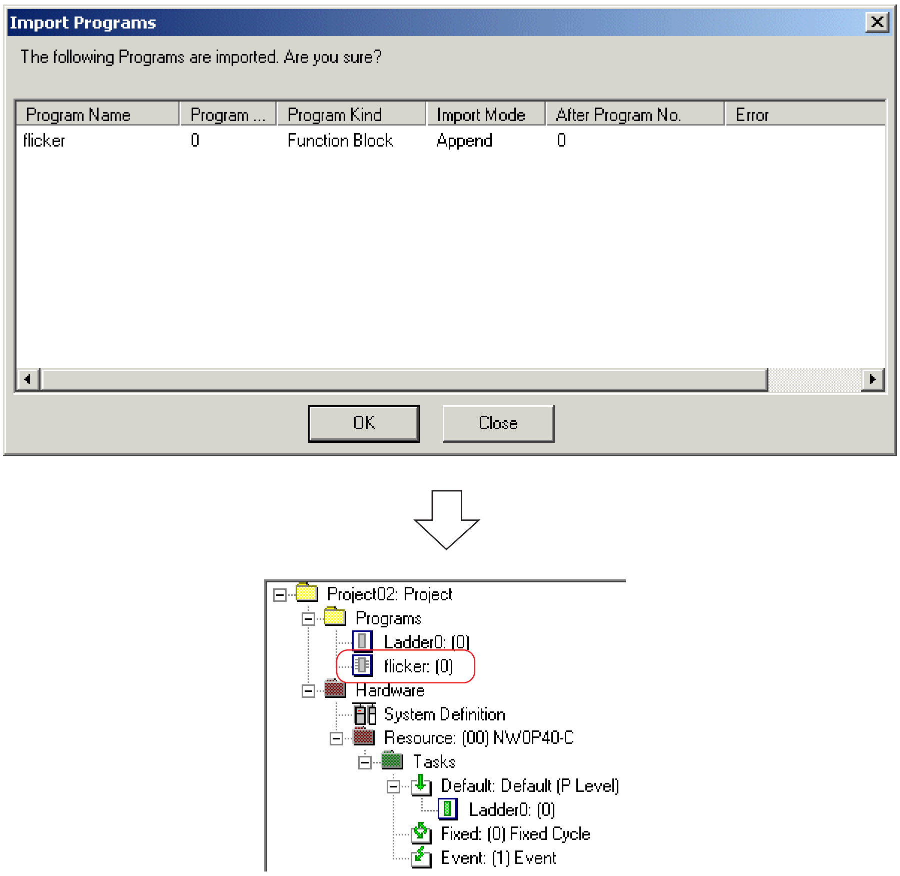
◆ Execute the [Import Programs...] command in the [File] menu to display the [Import Programs] dialog.



◆ Click the [Browse...] button and select the source project for import.



◆ After selecting the program to be imported, click the [OK] button to display the confirmation dialog shown below. Clicking the [OK] button imports the selected program.



* Use the imported user FB following the same procedure as (2).

Appendix 5 Setting Character String Data

Appendix 5 Procedures for Creating User FBs

Character strings are represented in Fuji Electric's original code which is based on the Shift JIS coding system. Whereas Shift JIS strings are mixtures of 8- and 16-bit codes, this coding system extends 8-bit code into 16-bit code so that the length of a single character is fixed at 16 bits. Strings have a variable length and can be as long as 64 characters. A NULL code (0000 (hex)) is automatically appended to the end of each string. Consequently, when a character string is declared, a memory space equal to the length of the character string plus one character, 65 words, is reserved.

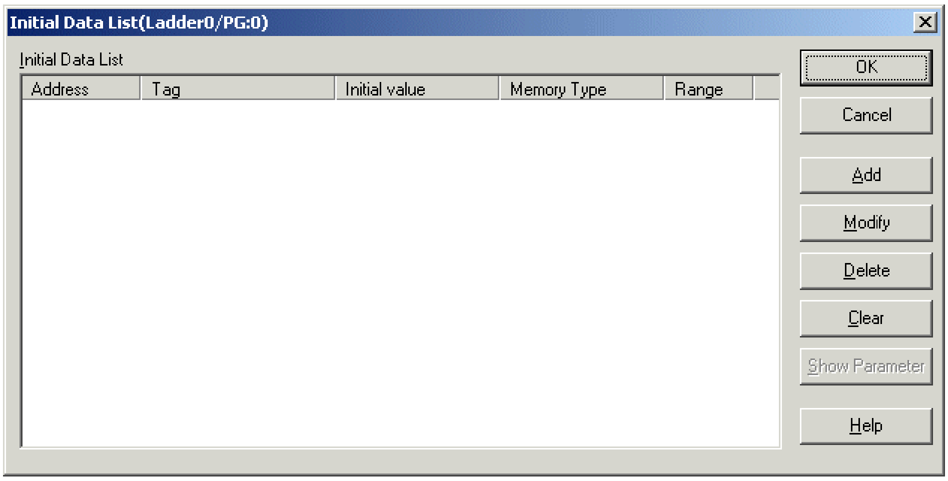
Example: For character string "ABCDEF," the Shift JIS code and string data type are represented as follows:

Character string		Shift JIS code	Character string data type	
A	→	41	00 ¦ 41	
B	→	42	00 ¦ 42	
C	→	43	00 ¦ 43	
D	→	44	00 ¦ 44	
E	→	45	00 ¦ 45	
F	→	46	00 ¦ 46	
			00 ¦ 00	←NULL code

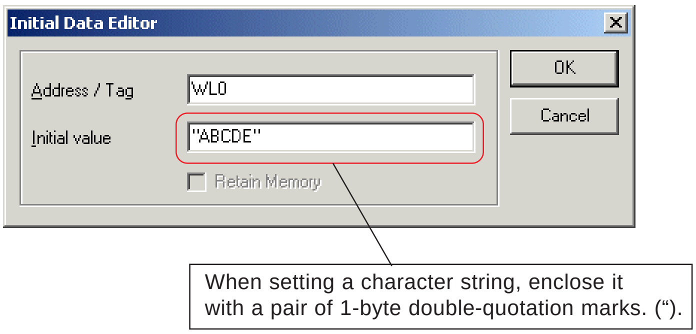
<Setting character string data with SX-Programmer Standard>

If character string data is to be used in application programs, it will be helpful to set the data in a device as initial values. It can be set in the procedure given below.

◆ Execute [Initial data list...] from the [PLC functions] menu to display the [Initial Data List] dialog.

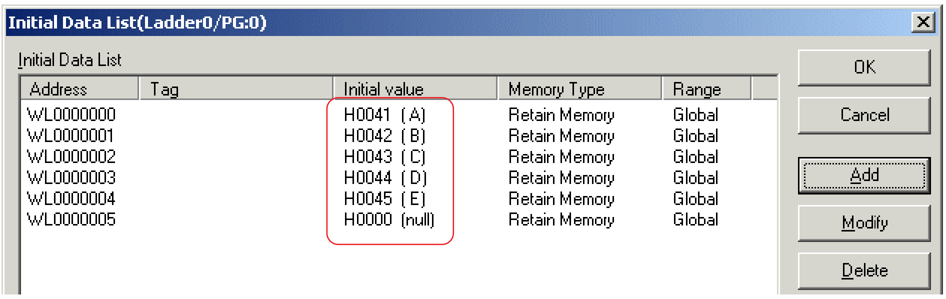


◆ Click the [Add] button to display the [Initial Data Editor] dialog.
From this dialog, enter the device (heading address) where the initial value is to be set and the character string initial value.

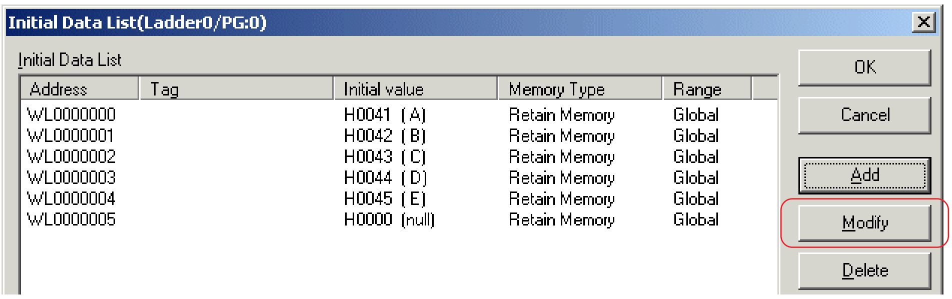


Appendix 5 Procedures for Creating User FBs

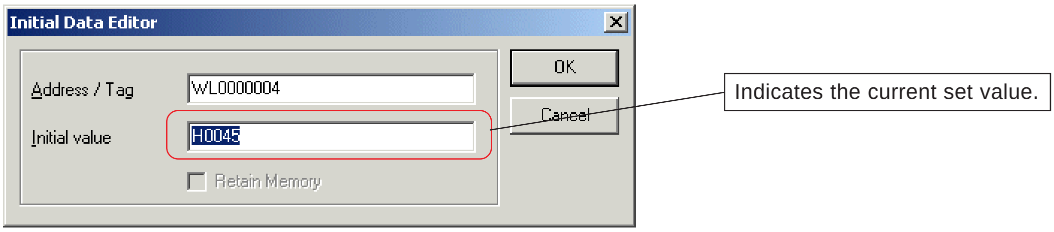
◆ After setting the device and character string, click the [OK] button to write the set character string to the [Initial Data List].



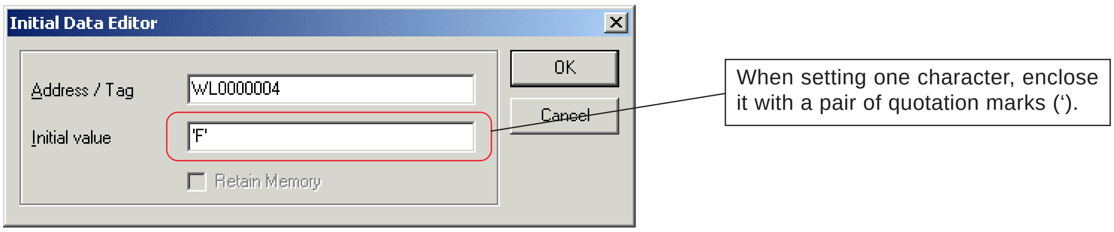
◆ To change a character in the set character string, select that character and click the [Modify] button.



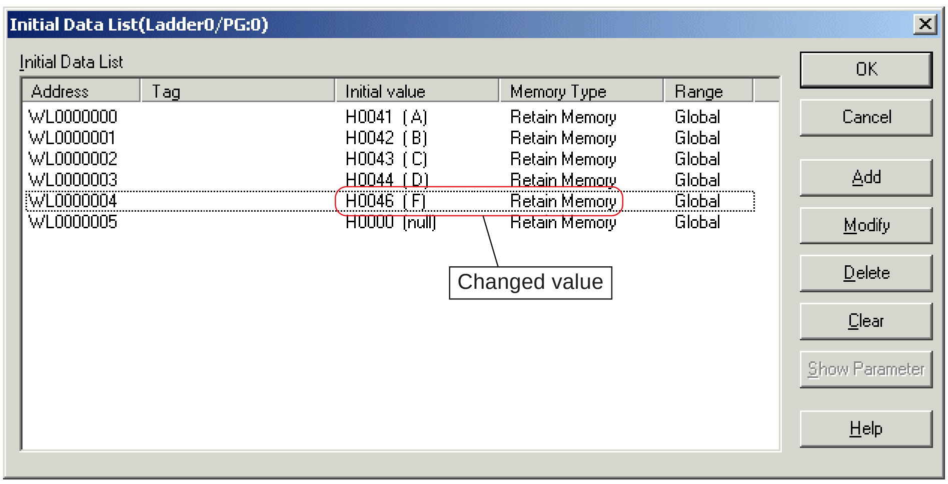
◆ The [Initial Data Editor] dialog appears.



◆ Enter a character enclosed in a pair of 1-byte single-quotation marks (‘).



◆ After setting, click the [OK] button. The change is reflected to the list.

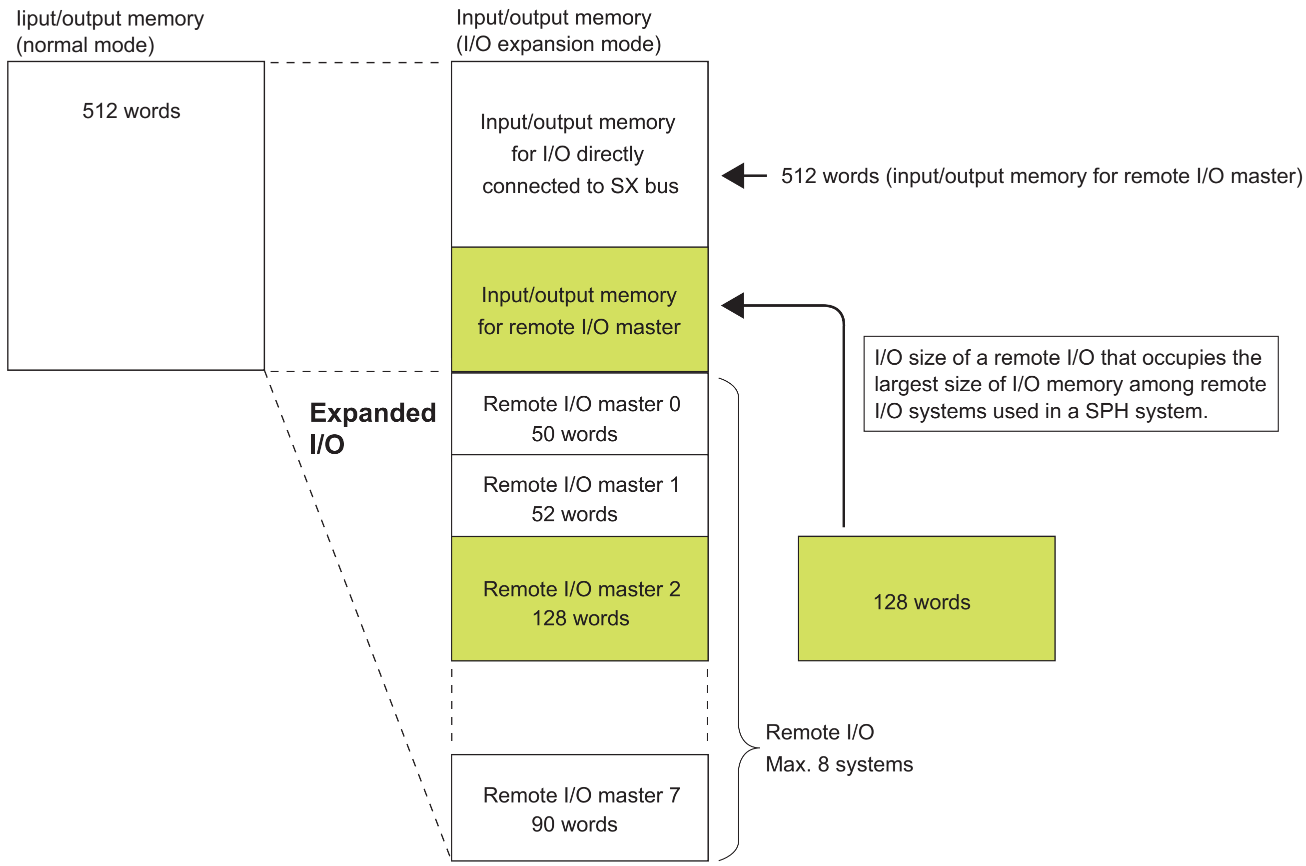


Appendix 6 I/O Expansion Function

Appendix 6 I/O Extension Function

Appendix 6-1 Overview

The I/O expansion function expands input/output memory for remote I/O systems when there are multiple remote I/O systems (max. 8 systems) in one SPH system (one configuration). This function is useful when units that occupy large number of words for input/output memory are used on multiple remote I/O systems and the total number of words occupied for input/output memory by units directly connected to the SX bus and all remote I/O systems on one configuration exceeds 512 words.



* The number of input/output words of a system that occupies the largest number of words among multiple remote I/O systems used in a SPH system is reserved as “input/output memory for remote I/O master” in the input/output memory (512 words). (This is automatically calculated by the loader according to a created system definition.)
  The “input/output memory for remote I/O master” subtracted from 512 words is the “input/output memory for I/O directly connected to the SX bus”.
* For using the I/O expansion function, you need to use a CPU module, remote I/O master module and the Standard loader that support this function. You cannot use this function with a CPU module that is not on the list below, a high-performance CPU board and a standard CPU module

Product name	Type	Supported version
High-performance CPU	NP1PS-74	V2364 or later
	NP1PS-74R NP1PS-117 NP1PS-117R	V2464 or later
	NP1PS-245R	V2064 or later
Remote I/O master module	NP1L-TL1	V2347 or later
SX-Programmer Standard	NP4H-SWN	V2.2.2.0 or later

* The address assignment conventions in the “I/O expansion mode” are the same as those in the “normal mode mode”. For information on the address assignment conventions, refer to “2-3 Input/output Address Assignment” in this manual.

Appendix 6 I/O Extension Function

Appendix 6-2 Specifications

Appendix 6-2-1 Expandable input/output memory size

(1) Maximum expandable input/output memory size

Type	Input/output memory size	Max. memory size of one remote I/O master	Max. number of connectable remote I/O master modules
NP1PS-74/74R	Max. 1856 words	Max. 192 words	Max. 8
NP1PS-117/117R NP1PS-245R	Max. 4096 words	Max. 512 words	Max. 8

(2) Sizes of input/output memory for I/O directly connected to the SX bus and for remote I/O master

When using the I/O expansion function, the **"input/output memory size for remote I/O master"** subtracted from **512 words of input/output memory** is the input/output memory size that can be used by modules and units directly connected to the SX bus. **"Input/output memory size for remote I/O master"** is the input/output memory size of a remote I/O system that occupies the largest number of input/output words among multiple remote I/O systems used in a SPH system.

<Sample calculation of input/output memory size for I/O directly connected to the SX bus>

Memory split structure	Image of input/output memory area	Max. input/output memory size
Pattern 1 (Normal mode)	512 words Both I/O directly connected to SX bus and remote I/O exist.	Total = 512 words
Pattern 2 (I/O expansion mode)	When memory size for remote I/O is 128 words: 384 words 128 words Size of I/O directly connected to SX bus = 512 words - 128 words = 384 words	Total = 384 + 128 x 8 = 1408 words
	When memory size for remote I/O is 512 words: 512 words Size of I/O directly connected to SX bus = 512 words - 512 words = 0 word	Total = 0 + 512 x 8 = 4096 words

Appendix 6-2-2 Takt time

In a system using the I/O expansion function, set the "SX bus takt time" as explained below.

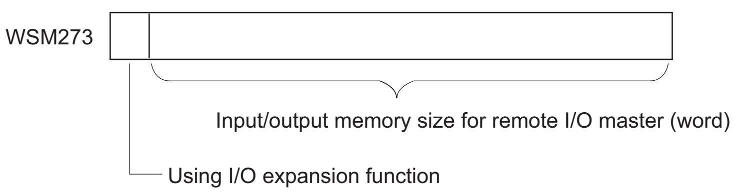
Total number of input/ourput words *	SX bus takt time
1024 words or less	2.0ms or more
2048 words	2.5ms or more
3072 words	3.0ms or more
3584 words	3.5ms or more
4096 words	4.0ms or more
4608 words	4.5ms or more
5120 words	5.0ms or more
5632 words	5.5ms or more
6144 words	6.0ms or more
6656 words	6.5ms or more
7158 words	7.0ms or more
7680 words	7.5ms or more
8192 words	8.0ms or more
8704 words	8.5ms or more
9216 words	9.0ms or more
9728 words	9.5ms or more
10240 words	10.0ms

* Total number of input/output words means the total number of input/output words assigned to all the tasks (default task, 0 level to 3 level) of the SPH system. Because input/output can be assigned to multiple tasks, the total number of input/output words can exceed 4096 words. However, the number of input/output words of a module or a unit that is indicated as "no equipment" is not counted.

Notes: 1) Be sure to set the takt time at 2 ms or more. A value less than 2 ms cannot be set. (An error occurs in the loader.)
2) When configuring a system, ensure that the total number of input/output words is 10240 or less. If it is beyond the range shown in the table above, a system operation definition error (SM222 is set to ON) occurs and the system cannot operate.
3) If a takt period monitoring error (SM39F is set to ON) occurs even though the total number of input/output words is within the range shown in the table above, increase the SX bus tack time.

Appendix 6-2-3 System memory concerning I/O expansion

System memory concerning I/O expansion function is shown below:



- **Using I/O expansion function**
  This is set to ON if the target system is operating in I/O expansion mode.
- **Input/output memory size for remote I/O master**
  The input/output memory size of a remote I/O system that occupies the largest number of words for input/output memory among multiple remote I/O systems in a SPH system is indicated in units of words.

Appendix 6-2-4 Refresh time of input/output memory

For refresh of input/output of remote I/O masters in a system using the I/O expansion function, one remote I/O system is refreshed per takt. Therefore, the refresh time is obtained by the following formula.

(Input/output refresh time of remote I/O master) = (SX bus takt time) x (No. of remote I/O masters)

* Input/output memory for I/O directly connected to SX bus is refreshed in every takt.

Notes:
1) The above formula is used to calculate the refresh time between the remote I/O master modules and the input/output memory area in the CPU.
For information on a communications cycle time between a remote I/O master module and a remote I/O slave station, refer to a user's manual for each remote I/O master module.
2) On a remote I/O, do not mount a device that requires high-speed input/output responsivity.
3) For the refresh time of input/output of remote I/O masters, only I/O remote masters that are actually connected to the system are refreshed.
Example 1) A remote I/O master registered as "no equipment" is skipped (not refreshed).
Example 2) If a remote I/O master is disconnected during operation, the disconnected remote I/O master is skipped (not refreshed). For example, if there are four remote I/O systems, each remote I/O master is refreshed once every four tasks. However, if one of the remote I/O masters is disconnected, every other remote I/O master is refreshed once every three tasks. When the disconnected remote I/O master is restored, each remote I/O master is refreshed once every four tasks as before.

Appendix 6-2-5 Special notes

(1) The I/O expansion function is not available in a redundant system and multi-CPU system.
(2) When using the I/O extension funcation, apply processing speed from "Appendix 7 Instruction Process Speed Chart" in this manual for the processing speeds of commands using addresses assigned to remote I/O.
(3) It is not possible to contain both a remote I/O system using the I/O expansion function and a remote I/O system not using it in one configuration. (The loader blocks such a mixed configuration from being set.)

Appendix 7   Instruction Processing Speed Chart

Appendix 7 Instruction Processing Speed Chart

<MICREX-SX SX-Programmer Standard instruction processing speed chart>

SX-Programmer Standard Instruction			No. of steps	Instruction processing speed (μs)		
				SPH300	SPH200	SPH2000
Basic instructions						
	Normal open contact	LD, AND, OR	1	0.02 to 0.06	0.07	0.03 to 0.18
	Normal close contact	LDI, ANI, ORI	1	0.02 to 0.06	0.07	0.03 to 0.18
	Rising edge differential normal open contact	LD+, AND+, OR+	8	0.16	32	0.59 to 2.06
	Falling edge differential normal open contact	LD-, AND-, OR-,	8	0.16	32	0.59 to 2.06
	Rising edge differential normal close contact	LDI+, ANI+, ORI+,	8	0.16	32	0.59 to 2.06
	Falling edge differential normal close contact	LDI-, ANI-, ORI-	8	0.16	32	0.59 to 2.06
	Coil	OUT	1	0.02 to 0.06	0.14	0.07 to 0.39
	Inverted coil	OUTI	1	0.02 to 0.06	0.14	0.08 to 0.43
	Set coil	SET	1	0.04 to 0.12	0.14	0.07 to 0.39
	Reset coil	RST	1	0.04 to 0.12	0.14	0.07 to 0.39
	Rising edge differential coil	OUT+	10	0.16	32	0.64 to 2.31
	Falling edge differential coil	OUT-	10	0.16	32	0.64 to 2.31
	Invert	NOT	1	0.06	0.35	0.05 to 0.29
	Step control	OUTSC	7	0.08 to 0.16	0.56	0.04 to 0.22
Program control instructions						
	Jump end	JEND	1	0.02	0.14	0
	Unconditional jump	JMP	2	0.08	0.21	0.02 to 0.11
	Conditional jump	JMPC	2	0.1	0.14 to 0.21	0.03 to 0.18
	Negative conditional jump	JMPCN	2	0.1	0.14 to 0.21	0.03 to 0.18
	Loop	LOOP	14	0.46 to 0.86	2.03 to 2.17	0.52 to 2.7
	Loop end	CONT	3	0.06	0.07	0.02 to 0.11
	Unconditional return	RET	3	User FCT: 3.50 User FB: 3.00	User FCT: 15.14 User FB: 18.14	User FCT: 0.63 to 0.92 User FB: 0.49 to 0.73
	Conditional return	RETC	3			
	Negative conditional return	RETCN	3			
	Master control set	MC	7	0.14 to 0.30	0.84	0.3 to 1.65
	Master control reset	MCR	6	0.12 to 0.28	0.56	0.09 to 0.51
	Section	SECTION	3	0.06	0.72	0
Conversion instructions						
	BIN to BCD	BCD	6	0.12 to 0.64	12.35 to 25.49	0.25 to 0.53
	BCD to BIN	BIN	6	0.12 to 0.66	12.35 to 21.49	0.33 to 0.57
	Integer to real (Signed integer 16 bits)	I_TO_R	6	0.34 to 0.42	14.42	0.17 to 0.47
	Integer to real (Signed integer 32 bits)	DI_TO_R	6	0.10 to 0.18	17.49	0.17 to 0.47
	Integer to real (Unsigned integer 16 bits)	UI_TO_R	6	0.34 to 0.42	12.42	0.19 to 0.47
	Integer to real (Unsigned integer 32 bits)	UDI_TO_R	6	0.46 to 0.54	14.49	0.19 to 0.47
	Real to integer (Signed integer 16 bits)	R_TO_I	6	0.70 to 0.78	12.42	0.2 to 0.51
	Real to integer (Signed integer 32 bits)	R_TO_DI	6	0.10 to 0.18	23.49	0.2 to 0.51
	Real to integer (Unsigned integer 16 bits)	R_TO_UI	6	0.66 to 0.74	12.42	0.2 to 0.51

Appendix 7 Instruction Processing Speed Chart

SX-Programmer Standard	Instruction	No. of steps	Instruction processing speed (μs)		
			SPH300	SPH200	SPH2000
Conversion instructions (Cont.)					
Real to integer (Unsigned integer 32 bits)	R_TO_UDI	6	0.76 to 0.84	21.49	0.2 to 0.51
Real to integer (Signed integer 16 bits, fractional part truncation)	TRUNC_I	6	0.70 to 0.78	12.42	0.19 to 0.51
Real to integer (Signed integer 32 bits, fractional part truncation)	TRUNC_DI	6	0.10 to 0.18	21.49	0.19 to 0.51
Real to integer (Unsigned integer 16 bits, fractional part truncation)	TRUNC_UI	6	0.66 to 0.74	12.42	0.2 to 0.51
Real to integer (Unsigned integer 32 bits, fractional part truncation)	TRUNC_UDI	6	0.76 to 0.84	21.49	0.2 to 0.51
Decode	DECODE	6	0.06 to 0.16	8.35 to 16.49	0.06 to 0.37
Encode	ENCODE	6	0.06 to 0.16	10.35 to 18.49	0.06 to 0.48
Bit count	BITCOUNT	6	1.28 to 3.50	10.35 to 13.42	0.06 to 0.44
Arithmetic operation instructions					
Addition (Signed)	ADD	6	0.12 to 0.24	0.56	0.07 to 0.4
Addition (Unsigned)	ADD_UAI	6	0.12 to 0.24	0.56	0.07 to 0.4
Addition (Real)	ADD_R	6	0.12 to 0.24	0.56	0.29 to 0.62
Subtraction (Signed)	SUB	6	0.08 to 0.20	0.21	0.07 to 0.4
Subtraction (Unsigned)	SUB_UAI	6	0.08 to 0.20	0.21	0.07 to 0.4
Subtraction (Real)	SUB_R	6	0.08 to 0.20	0.21	0.30 to 0.62
Multiplication (Signed)	MUL	6	0.14 to 0.26	20.42	0.08 to 0.48
Multiplication (Unsigned)	MUL_UAI	6	0.14 to 0.26	20.42	0.08 to 0.48
Multiplication (Real)	MUL_R	6	0.14 to 0.26	20.42	0.27 to 0.63
Division (Signed)	DIV	6	1.12 to 1.24	10.14	0.25 to 0.59
Division (Unsigned)	DIV_UAI	6	1.12 to 1.24	10.14	0.25 to 0.59
Division (Real)	DIV_R	6	1.12 to 1.24	10.14	0.34 to 0.67
Division remainder (Signed)	MOD	6	1.12 to 1.24	10.14	0.25 to 0.59
Division remainder (Unsigned)	MOD_UAI	6	1.12 to 1.24	10.14	0.25 to 0.59
Exponent	EXPT	6	16.86 to 16.98	2206.49	0.65 to 1.04
Absolute	ABS	6	0.5	9.35 to 11.49	0.07 to 0.48
Absolute (Real)	ABS_R	6	0.1	7.49	0.05 to 0.29
Square root	SQRT	6	4.24 to 4.32	613.49	2.06 to 2.35
Natual logarithm	LN	6	5.44 to 5.52	1700.49	2.09 to 2.46
Common logarithm	LOG	6	5.84 to 5.92	1726.49	2.3 to 2.63
Exponent	EXP	6	16.86 to 16.98	244.49	0.77 to 1.08
Sine	SIN	6	6.24 to 6.32	3856.49	15.2 to 15.6
Cosine	COS	6	6.24 to 6.32	3856.49	15.2 to 15.7
Tangent	TAN	6	11.04 to 11.12	7606.49	31.2 to 32.3
Arcsine	ASIN	6	11.84 to 11.92	3606.49	16.1 to 16.5
Arccosine	ACOS	6	11.84 to 11.92	3706.49	16.3 to 16.7
Arctangent	ATAN	6	6.84 to 6.92	2806.49	24.7 to 26.1
32 bits addition with carry	ADC	9	1.02 to 1.18	20.19	0.74 to 1.81
32 bits addition carry	ADCO	9	1.16 to 1.32	20.19	0.82 to 1.72

Appendix 7 Instruction Processing Speed Chart

SX-Programmer Standard	Instruction	No. of steps	Instruction processing speed (µs)		
			SPH300	SPH200	SPH2000
Arithmetic operation instructions (Cont.)					
32 bits subtraction with borrow	SBB	9	1.04 to 1.20	21.19	0.74 to 1.82
32 bits subtraction borrow	SBBO	9	1.16 to 1.32	21.19	0.83 to 1.73
64 bits multiplication (Lower-order digit)	MULL	8	1.14 to 1.26	31.19	0.61 to 1.39
64 bits multiplication (Higher-order digit)	MULU	8	1.14 to 1.26	31.19	0.61 to 1.39
64 bits division (Lower-order digit)	DIVL	9	4.60 to 4.76	42.26	1.12 to 2.04
64 bits division (Higher-order digit)	DIVU	9	4.60 to 4.76	42.26	0.95 to 1.88
Transfer instructions					
Move	MOVE	7	0.08 to 0.16	0.56	0.04 to 0.22
Negation	NEG	6	0.06 to 0.14	7.49	0.05 to 0.29
Negation (Real)	NEG_R	6	0.06 to 0.14	7.49	0.01 to 0.07
Block move	BMOV	7	1.0 + 0.2 x (No. of move)	82.77 + 0.14 x (No. of move)	0.63 + 0.03 x (No. of move)
File move	FMOV	19	0.4 + 0.52 x (No. of move)	0.21 + 2.1 x (No. of move)	0.04 + 0.17 x (No. of move) to 0.22 + 1.21 x (No. of move)
Exchange	XCH	11	0.12 to 0.36	0.63	0.12 to 0.66
Indirect put (Block move)	BDMPX	12	1.02 + 0.2 x (No. of move) to 1.08 + 0.2 x (No. of move)	131.46 + 0.14 x (No. of move)	0.64 + 0.03 x (No. of move) to 0.93 + 0.03 x (No. of move)
Indirect get (Block move)	BMPX	12	1.02 + 0.2 x (No. of move) to 1.08 + 0.2 x (No. of move)	131.46 + 0.14 x (No. of move)	0.64 + 0.03 x (No. of move) to 0.93 + 0.03 x (No. of move)
Bit string operation instructions					
Logical AND	AND_AW	6	0.12 to 0.24	0.21	0.07 to 0.4
Logical OR	OR_AW	6	0.12 to 0.24	0.21	0.07 to 0.4
Logical exclusive OR (Bit)	XOR_B	6	0.12 to 0.24	0.21	0.13 to 0.75
Logical exclusive OR (Word, double-word)	XOR_AW	6	0.12 to 0.24	0.21	0.07 to 0.4
Logical exclusive NOR (Bit)	XORN_B	6	0.12 to 0.24	0.21	0.13 to 0.75
Logical exclusive NOR (Word, double-word)	XORN_AW	6	0.12 to 0.24	0.21	0.07 to 0.4
Logical NOT	NOT_AW	6	0.06 to 0.14	0.35 to 0.49	0.05 to 0.29
Rotation right	ROR	6	0.08 to 0.20	9.42 to 10.56	0.08 to 0.51
Rotation left	ROL	6	0.08 to 0.20	9.42 to 10.56	0.08 to 0.51
Shift right	SHR	6	0.08 to 0.20	9.42 to 9.56	0.07 to 0.44
Shift left	SHL	6	0.08 to 0.20	9.42 to 9.56	0.07 to 0.44
Set bit	SBIT	6	0.16 to 0.28	0.49 to 0.63	0.08 to 0.44
Reset bit	RBIT	6	0.16 to 0.28	0.49 to 0.63	0.08 to 0.44
Test bit	TBIT	6	0.14 to 0.26	0.56 to 0.63	0.12 to 0.4
Shift left 32 bits with carry	SLC	8	1.14 to 1.26	18.84	0.62 to 1.52
Shift right 32 bits with carry	SRC	8	1.14 to 1.26	18.84	0.62 to 1.53
Shift left 32 bits carry	SLCO	7	1.08 to 1.16	17.56	0.54 to 0.99
Shift right 32 bits carry	SRCO	7	1.08 to 1.16	17.56	0.54 to 0.98

Appendix 7 Instruction Processing Speed Chart

SX-Programmer Standard Instruction		No. of steps	Instruction processing speed (μs)		
			SPH300	SPH200	SPH2000
Selection/comparison instructions					
Comparison (S1 > S2)	LD>, LD_UAI>, LD_R>, LD_B>, LD_AW>, LD_TOD>, LD_DT>, LD_D>, LD_T>	3	0.14 to 0.24	0.56	0.1 to 0.29
Comparison (S1 = S2)	LD=, LD_UAI=, LD_R=, LD_B=, LD_AW=, LD_TOD=, LD_DT=, LD_D=, LD_T=	3	0.14 to 0.24	0.56	0.1 to 0.29
Comparison (S1 ≠ S2)	LD<>, LD_UAI<>, LD_R<>, LD_B<>, LD_AW<>, LD_TOD<>, LD_DT<>, LD_D<>, LD_T<>	3	0.08 to 0.20	0.91	0.1 to 0.29
Comparison (S1 >= S2)	LD>=, LD_UAI>=, LD_R>= LD_B>=, LD_AW>=, LD_TOD>=, LD_DT>=, LD_D>=, LD_T>=	3	0.14 to 0.24	0.56	0.1 to 0.29
Comparison (S1 < S2)	LD<, LD_UAI<, LD_R<, LD_B<, LD_AW<, LD_TOD<, LD_DT<, LD_D<, LD_T<	3	0.14 to 0.24	0.56	0.1 to 0.29
Comparison (S1 <= S2)	LD<=, LD_UAI<=, LD_R<=, LD_B<=, LD_AW<=, LD_TOD<=, LD_DT<=, LD_D<=, LD_T<=	3	0.14 to 0.24	0.56	0.1 to 0.29
Select	SEL	12	0.30 to 0.50	1.05 to 1.19	0.15 to 0.84
Select	SEL_B	12	0.30 to 0.50	1.12	0.27 to 1.54
Select	SEL_UAI	12	0.30 to 0.50	1.05 to 1.19	0.15 to 0.84
Select	SEL_R	12	0.30 to 0.50	1.19	0.15 to 0.84
Select	SEL_AW	12	0.30 to 0.50	1.05 to 1.19	0.15 to 0.84
Select	SEL_T	12	0.30 to 0.50	1.19	0.15 to 0.84
Select	SEL_STR	12	0.30 to 0.50	87.47 + 0.64 x (Total No. of characters)	1.94 + 0.09 x (No. of characters) to 3.67 + 0.09 x (No. of characters)
Maximum value	MAX	6	0.42 to 0.54	0.84 to 1.19	0.04 to 0.22
Maximum value	MAX_UAI	6	0.42 to 0.54	0.84 to 1.19	0.04 to 0.22
Maximum value	MAX_R	6	0.42 to 0.54	29.05	0.04 to 0.22
Minimum value	MIN	6	0.42 to 0.54	0.84 to 1.19	0.04 to 0.22
Minimum value	MIN_UAI	6	0.42 to 0.54	0.84 to 1.19	0.04 to 0.22
Minimum value	MIN_R	6	0.42 to 0.54	29.05	0.04 to 0.22
Limit	LIMIT	9	0.88 to 1.00	0.98 to 1.26	0.04 to 0.22
Limit	LIMIT_UAI	9	0.88 to 1.00	0.98 to 1.26	0.04 to 0.22
Limit	LIMIT_R	9	0.88 to 1.00	29.12	0.04 to 0.22

Appendix 7 Instruction Processing Speed Chart

SX-Programmer Standard Instruction			No. of steps	Instruction processing speed (μs)		
				SPH300	SPH200	SPH2000
Character string instructions						
	Get length	LEN	6	0.84 + 0.30 x (No. of chr.) to 0.84 + 0.34 x (No. of chr.)	53.42 + 0.5 x (No. of chr.)	0.54 + 0.04 x (No. of chr.) to 0.97 + 0.04 x (No. of chr.)
	Get left sub-string	LEFT	8	1.44 + 0.48 x (No. of chr. extracted) to 1.48 + 0.56 x (No. of chr. extracted)	83.98 + 0.5 x (No. of chr.) + 0.14 x (No. of chr. extracted)	1.8 + 0.1 x (No. of chr. extracted) to 2.82 + 0.11 x (No. of chr. extracted)
	Get right sub-string	RIGHT	8	1.60 + 0.16 x (No. of input chr.) + 0.18 x (No. of chr. extracted)	85.98 + 0.5 x (No. of chr.) + 0.14 x (No. of chr. extracted)	1.94 + 0.04 x (No. of input chr.) + 0.1 x (No. of chr. extracted) to 2.98 + 0.04 x (No. of input chr.) + 0.1 x (No. of chr. extracted)
	Get middle sub-string	MID	9	1.66 + 0.48 x (No. of chr. extracted) to 1.70 + 0.56 x (No. of chr. extracted)	87.26 + 0.5 x (No. of chr.) + 0.14 x (No. of chr. extracted)	1.9 + 0.41 x (No. of chr. extracted) to 3.1 + 0.41 x (No. of chr. extracted)
	Concatenate	CONCAT	9	2.80 + 0.48 x ((No. of first input chr.) + (No. of second input chr.)) to 2.80 + 0.56 x ((No. of first input chr.) + (No. of second input chr.))	49.19 + 40 x (No. of input str.) + 0.5 x (Total No. of input chr.) x (No. of input str.) + 0.14 x (No. of output chr.)	1.9 + 0.09 x ((No. of first input chr.) + (No. of second input chr.)) to 3.06 + 0.09 x ((No. of first input chr.) + (No. of second input chr.))
	Insert string	INSERT	9	2.04 + 1.06 x (No. of input str.) + 0.48 x (No. of chr. inserted) to 2.16 + 1.22 x (No. of input str.) + 0.56 x (No. of chr. inserted)	138.19 + 0.64 x (No. of output chr.)	2.41 + 0.09 x (No. of input str.) + 0.01 x (No. of chr. inserted) to 3.54 + 0.11 x (No. of input str.) + 0.01 x (No. of chr. inserted)
	Delete string	DELETE	9	2.04 + 1.06 x ((No. of input str.) - (No. of chr. deleted)) + 0.36 x (No. of chr. deleted) to 2.16 + 1.22 x ((No. of input str.) - (No. of chr. deleted)) + 0.56 x (No. of chr. deleted)	110.76 + 0.5 x ((No. of input chr.) + (No. of chr. deleted)) + 0.14 x (No. of output chr.)	2.0 + 0.09 x ((No. of input str.) - (No. of chr. deleted)) + 0.05 x (No. of chr. deleted) to 3.7 + 0.09 x ((No. of input str.) - (No. of chr. deleted)) + 0.05 x (No. of chr. deleted)
	Replace string	REPLACE	10	2.42 + 1.06 x ((No. of input str.) - (No. of chr. replaced)) + 0.60 x (No. of chr. replaced) to 2.70 + 1.22 x ((No. of input str.) - (No. of chr. replaced)) + 0.68 x (No. of chr. replaced)	139.47 + (No. of input chr.) + 0.5 x ((No. of str. replaced) - (No. of chr. replaced)) + 0.14 x (No. of output chr.)	2.15 + 0.094 x ((No. of input str.) - (No. of chr. replaced)) + 0.201 x (No. of chr. replaced) to 4.15 + 0.09 x ((No. of input str.) - (No. of chr. replaced)) + 0.132 x (No. of chr. replaced)
	Find string	FIND	7	0.96 + (0.26 + 0.54 x (No. of input str.)) x (No. of chr. searched) to 0.96 + (0.30 + 0.58 x (No. of input str.)) x (No. of chr. searched)	92.63 + 1.5 x ((No. of first input chr.) + (No. of second input chr.))	0.73 + 0.05 x (No. of chr. searched) to 1.44 + 0.05 x (No. of chr. searched)
	Compare string	LD_STR>, LD_STR>=, LD_STR<=, LD_STR<	4	1.24 + 0.58 x (No. of chr.) to 1.32 + 0.66 x (No. of chr.)	99.56 + 2.3 x (Position where comparison result is output) + 0.5 x (Total No. of input chr.)	0.86 + 0.04 x (No. of chr.) to 1.57 + 0.04 x (No. of chr.)
	Compare string	LD_STR=	4	1.24 + 0.58 x (No. of chr.) to 1.32 + 0.66 x (No. of chr.)	92.56 + 2.3 x (Position where comparison result is output) + 0.5 x (Total No. of input chr.)	0.86 + 0.04 x (No. of chr.) to 1.57 + 0.04 x (No. of chr.)
	Compare string	LD_STR<>	4	1.14 + 0.3 x (No. of chr.)	92.56 + 2.3 x (Position where comparison result is output) + 0.5 x (Total No. of input chr.)	0.82 + 0.05 x (No. of chr.) to 1.1 + 0.05 x (No. of chr.)
	Move string	MOVE_STR	8	13.86	83.78	2.0 to 3.72
	Convert string to number	STR_TO_UI	6	0.62 + 0.76 x (No. of chr.) to 0.66 + 0.80 x (No. of chr.)	63.35 + 6.5 x (No. of input chr.)	0.55 + 0.09 x (No. of chr.) to 0.89 + 0.09 x (No. of chr.)
	Convert number to string	UI_TO_STR	7	1.36 + 1.82 x (No. of chr.) to 1.36 + 1.94 x (No. of chr.)	61.77 + 0.14 x (No. of output chr.)	2.42 + 0.16 x (No. of chr.) to 4.05 + 0.16 x (No. of chr.)
	Convert shift-JIS to string	SJ_TO_STR	8	2.76 + 1.98 x (No. of chr.) to 3.02 + 2.10 x (No. of chr.)	71.98 + 7 x (No. of output chr.)	1.83 + 0.14 x (No. of chr.) to 3.5 + 0.14 x (No. of chr.)
	Convert stirng to shift-JIS	STR_TO_SJ	7	1.46 + 0.94 x (No. of chr.) to 1.52 + 1.02 x (No. of chr.)	68.70 + 5 x (No. of output chr.)	1.87 + 0.07 x (No. of chr.) to 3.34 + 0.07 x (No. of chr.)
	Byte length	BYTE_LEN	6	0.66 + 0.46 x (No. of chr.) to 0.70 + 0.50 x (No. of chr.)	56.35 + 0.5 x (No. of input chr.)	0.53 + 0.07 x (No. of chr.) to 1.07 + 0.08 x (No. of chr.)

Appendix 7 Instruction Processing Speed Chart

SX-Programmer Standard Instruction			No. of steps	Instruction processing speed (μs)		
				SPH300	SPH200	SPH2000
Time instructions						
	Add time	ADD_T_T	8	0.12 to 0.24	0.98	0.07 to 0.41
	Add time (Duration + Time)	ADD_TD_T	15	3.00 to 3.18	30.68	0.68 to 1.89
	Add time (Date and Time + Time)	ADD_DT_T	10	1.22 to 2.06	16.47	0.97 to 1.19
	Subtract time	SUB_T_T	8	0.12 to 0.24	0.98	0.07 to 0.41
	Subtract time (Date - Date)	SUB_D_D	10	0.18 to 0.34	19.19	0.1 to 0.63
	Subtract time (Duration - Time)	SUB_TD_T	13	3.00 to 3.18	30.96	0.72 to 1.75
	Subtract time (Duration - Duration)	SUB_TD_TD	10	0.18 to 0.34	19.19	0.1 to 0.63
	Subtract time (Date and Time - Time)	SUB_DT_T	10	3.00 to 3.18	16.47	0.44 to 1.19
	Subtract time (Date and Time - Date and Time)	SUB_DT_DT	10	0.18 to 0.34	19.19	0.1 to 0.63
	Multiply time	MUL_T_UDI	8	0.50 to 0.62	18.91	0.08 to 0.48
	Multiply time (Time x real)	MUL_T_R	9	0.54 to 0.66	81.05	0.58 to 1.2
	Divide time	DIV_T_UDI	8	2.10 to 2.36	15.91	0.25 to 0.69
	Divide time (Time / real)	DIV_T_R	9	2.10 to 2.36	131.05	0.65 to 1.27
	Concatenate time	CONCAT_D_D	8	0.12 to 0.24	0.98	0.07 to 0.41
	Date and time - Duration conversion	DT_TO_TOD	9	1.16 to 1.28	14.91	0.24 to 0.56
	Date and time - Date conversion	DT_TO_D	10	1.76 to 2.06	39.98	0.27 to 0.78
Analog or other instructions						
	Dead band	DBAND	8	1.34 to 1.46	19.70 to 21.91	0.72 to 1.46
	Daed band (Real)	DBAND_R	8	1.46 to 1.58	33.91	0.71 to 1.46
	Bias	BIAS	8	1.14 to 1.26	22.70 to 24.91	0.7 to 1.44
	Bias (Real)	BIAS_R	8	1.20 to 1.32	28.91	0.89 to 1.46
	Step sequence coil	SC_COIL	9	1.12 to 1.28	18.98	0.8 to 2.1
	Step sequence bit	SC	9	1.12 to 1.28	19.05	0.86 to 2.03
Timer, Counter						
	Up counter	CTU	13	0.22	35 to 39	0.7 to 2.11
	Down counter	CTD	13	0.22	35 to 39	0.69 to 2.11
	Pulse	TP	10	0.44	33 to 45	0.78 to 2
	On-delay timer	TON	10	0.44	37 to 42	0.78 to 1.99
	Off-delay timer	TOF	10	0.44	32 to 42	0.67 to 1.91
	Real-time clock	RTC	10	1.76	39	10.8 to 12.2
	Ring counter	RCT	13	0.22	36 to 42	0.7 to 2.11
	Integrating timer	TMR	14	0.44	34 to 44	0.8 to 2.27
	Retriggerable timer	MR	10	0.44	38 to 47	0.79 to 2.01
Function blocks						
	Set reset flip-flop	SR	9	0.16	30	0.56 to 1.58
	Reset set flip-flop	RS	9	0.16	30	0.56 to 1.58
	Rising edge trigger	R_TRIG	7	0.16	32	0.45 to 1.27
	Falling edge trigger	F_TRIG	7	0.16	32	0.45 to 1.27
	Open channel	M_OPEN	32	1.6	210	1.15 to 4.34
	Send message	M_SEND	23	1.76	190 + 0.14 x (No. of send words)	0.99 to 3.04
	Receive message	M_RECEIVE	23	1.92	169	1.01 to 3.18
	Direct read	READ_W	24	1.5	211 + 3 x (No. of receive request words)	1.12 to 3.46
	Direct read (BOOL)	READ_B	20	1.5	213 + 4 x (No. of receive request bits)	1.14 to 3.47

Appendix 7 Instruction Processing Speed Chart

SX-Programmer Standard		Instruction	No. of steps	Instruction processing speed (µs)		
				SPH300	SPH200	SPH2000
Function blocks (Cont.)						
	Direct write	WRITE_W	24	1.3	217 + 0.14 x (No. of send words)	1.12 to 3.46
	Direct write (BOOL)	WRITE_B	20	1.3	230 +7 x (No. of send words)	1.14 to 3.48
	Remote data read	R_READ	32	1.5	232	1.25 to 4.45
	Remote data write	R_WRITE	32	1.3	251 + 0.14 x (No. of send words)	1.26 to 4.5
	File data read	F_READ	28	1.5	168	1.18 to 3.9
	File data write	F_WRITE	26	1.3	278 + 0.14 x (No. of send words)	1.14 to 3.69
	Extension test & set	EXT_T_S	19	3	-	1.06 to 3.03
	Sequential file store	FFST	24	3.08 + 0.12 x (No. of words)	81 + 0.14 x (No. of words)	1.7 to 4.44
	Sequential file load first	FIFO	24	2.68 + 0.12 x (No. of words)	81 + 0.14 x (No. of words)	1.14 to 3.37
	Sequential file load last	FILO	24	2.68 + 0.12 x (No. of words)	81 + 0.14 x (No. of words)	1.14 to 3.77
	Filter	FILTER_DI	12	4.32	101 to 217	2.78 to 3.91
	Filter (Real)	DILTER_R	12	2.98	281	5.8 to 7.05
	Integrate	INT_DI	20	4.6	102 to 216	2.28 to 4.25
	Integrate (Real)	INT_R	20	4.02	239	2.28 to 4.25
	Differentiate	DIF_DI	12	3.24	84	3.94 to 4.81
	Differentiate (Real)	DIF_R	12	3.48	259	3.25 to 4.42
	Pulse count	PULSE_CNT	16	2.8	46 to 49	1.1 to 2.42
	Pulse output	PULSE_OUT	16	2.28	45 to 57	0.72 to 2.16
	Modulate pulse width	PWM	13	2.48	44 to 49	0.6 to 1.89
	Hardware RTC	HW_RTC	13	1.2	33 to 75	10.7 to 12.1
	Test & set	T_S	12	0.86	47	0.77 to 1.94
	Change bank	BANK_CHG	19	4.72	61 to 154	2.06 to 3.97

Fuji Electric Systems Co.,Ltd.

Gate City Ohsaki, East Tower, 11-2, Osaki 1-chome, Shinagawa-ku, Tokyo, 141-0032,Japan
Phone: +81-3-5435-7280
Fax: +81-3-5435-7425
URL http://www.fesys.co.jp/eng/