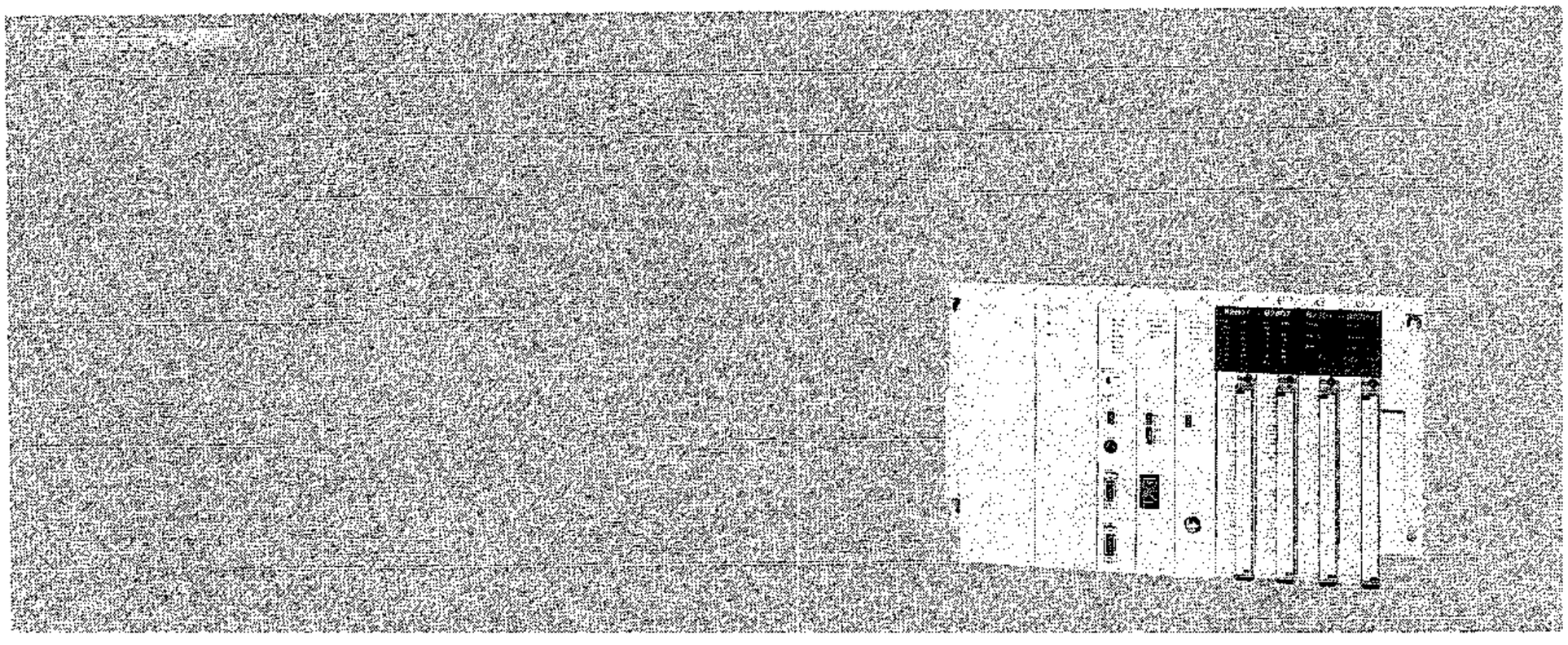
YASKAWA

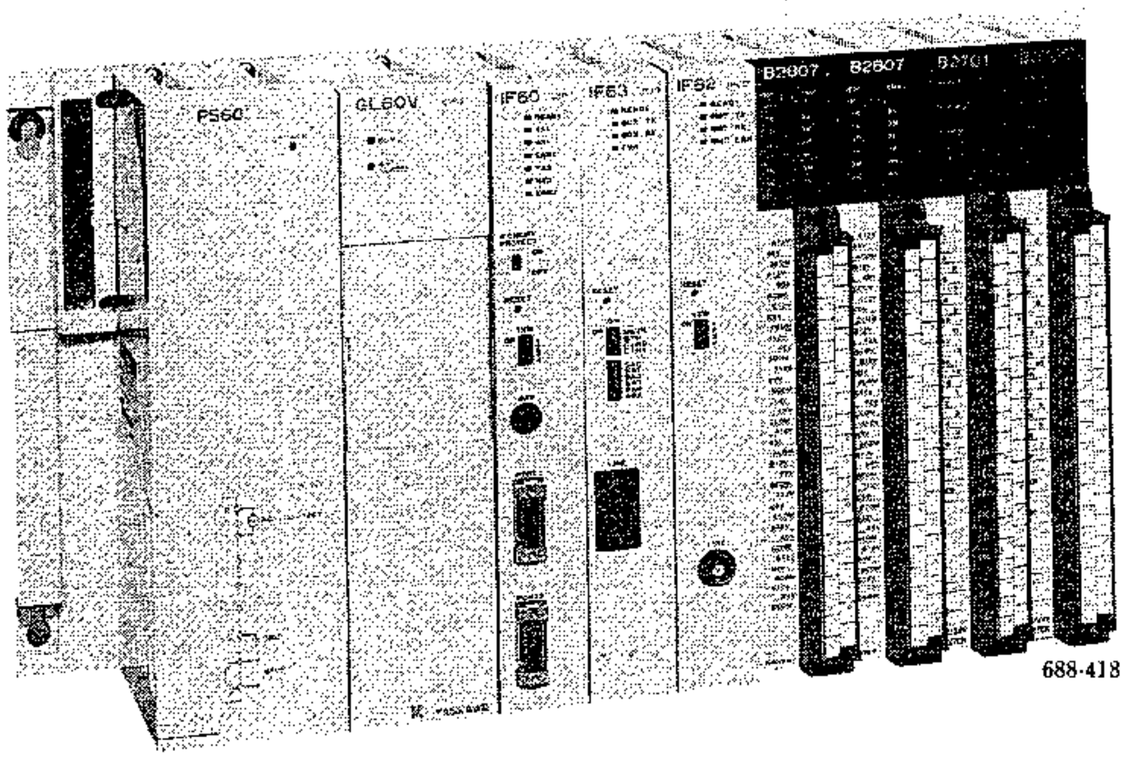
CONTROL PACK CP-3300
DESIGNER'S MANUAL



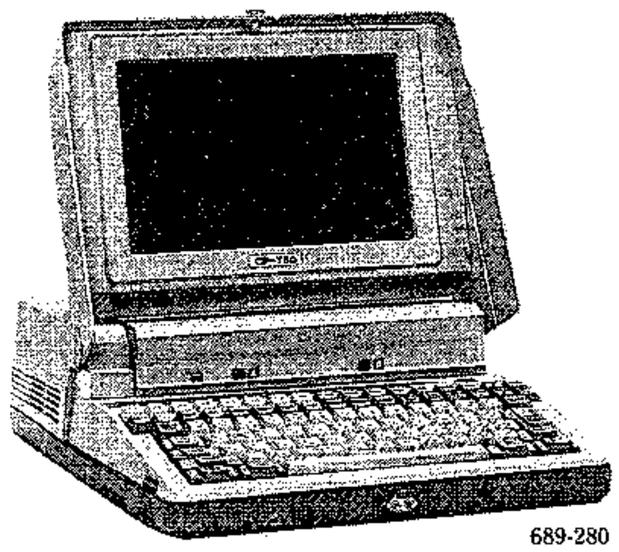
YASKAWA

MANUAL NO. SIE-C873-20.2D

The system controller **Control Pack CP-3300** (hereafter called CP-3300) is used for hierarchical and distributed control of a variety of industrial plant applications and equipment. It is most suitable as a main controller of a system calling for high-speed operation, including a drive system.
This handbook contains requirements for CP-3300 system design, hardware design, and software design.
Refer also to the following publications on CP-3300:



688-418


689-280

Control Pack CP-3300

[References]

KAE-C873-20 ··· **Control Pack CP-3300** ‹Catalog›
SIE-C873-20.3 ··· **Control Pack CP-3300**
Programming Panel Operator's Manual
SIE-C873-20.4 ··· **Control Pack CP-3300**
Maintenance Manual
SIE-C815-13.3 ··· 2000 Series I/O Modules

CONTENTS

CONTENTS

Page

Page

1 CP-3300 CONFIGURATION

This section explains the CP-3300 configuration.

It illustrates an example of a typical system configuration using the CP-3300 and a system diagram of modules configured in the CP-3300.

CONTENTS PAGE

1.1 SYSTEM CONFIGURATION

The CP-3300 is a programmable controller whose object is to create a hierarchical and distributed control system. It provides transmission functions (including I/O) as shown in Fig. 1.1.

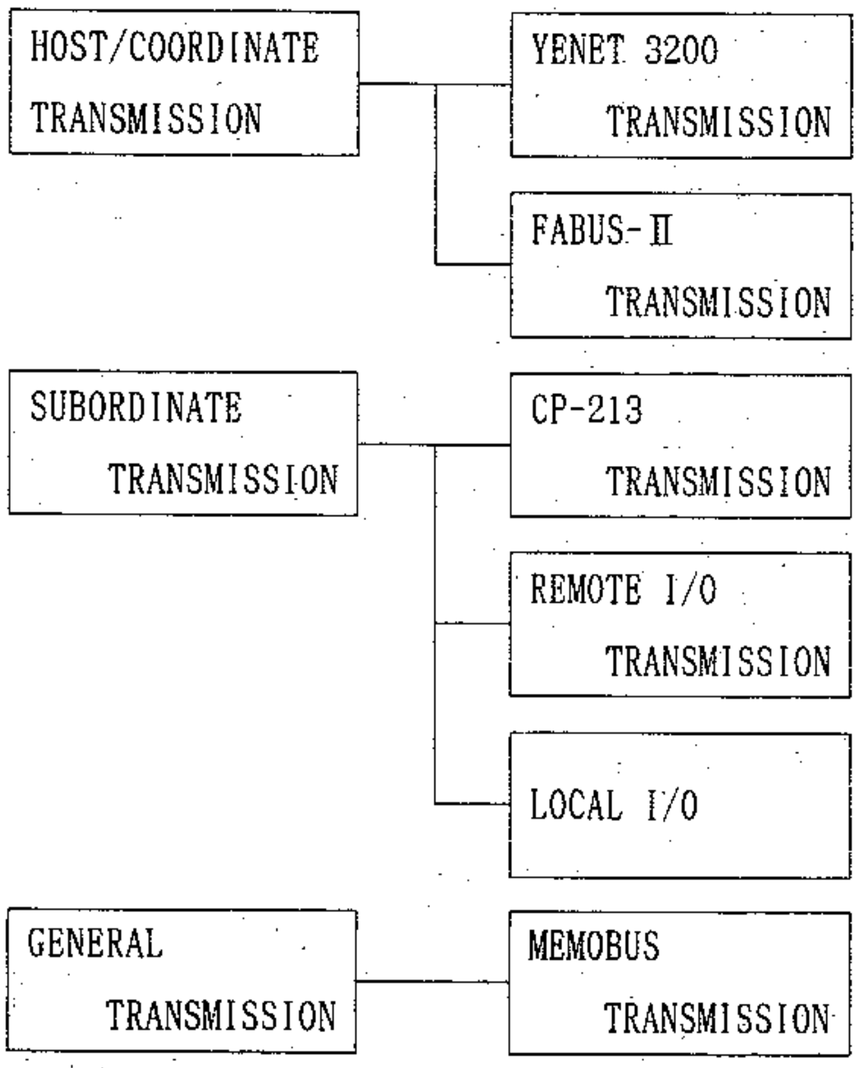


Fig. 1.1 CP-3300 transmission system

Fig. I.2 shows a typical example of system configuration of CP-3300. The following description of CP-3300 system is based on this figure.

The CPU module at the heart of the CP-3300 system is housed on the first rack of the local channel. The CPU module mounts a memory with battery back up for holding user programs and data and a microprocessor for executing the user programs.

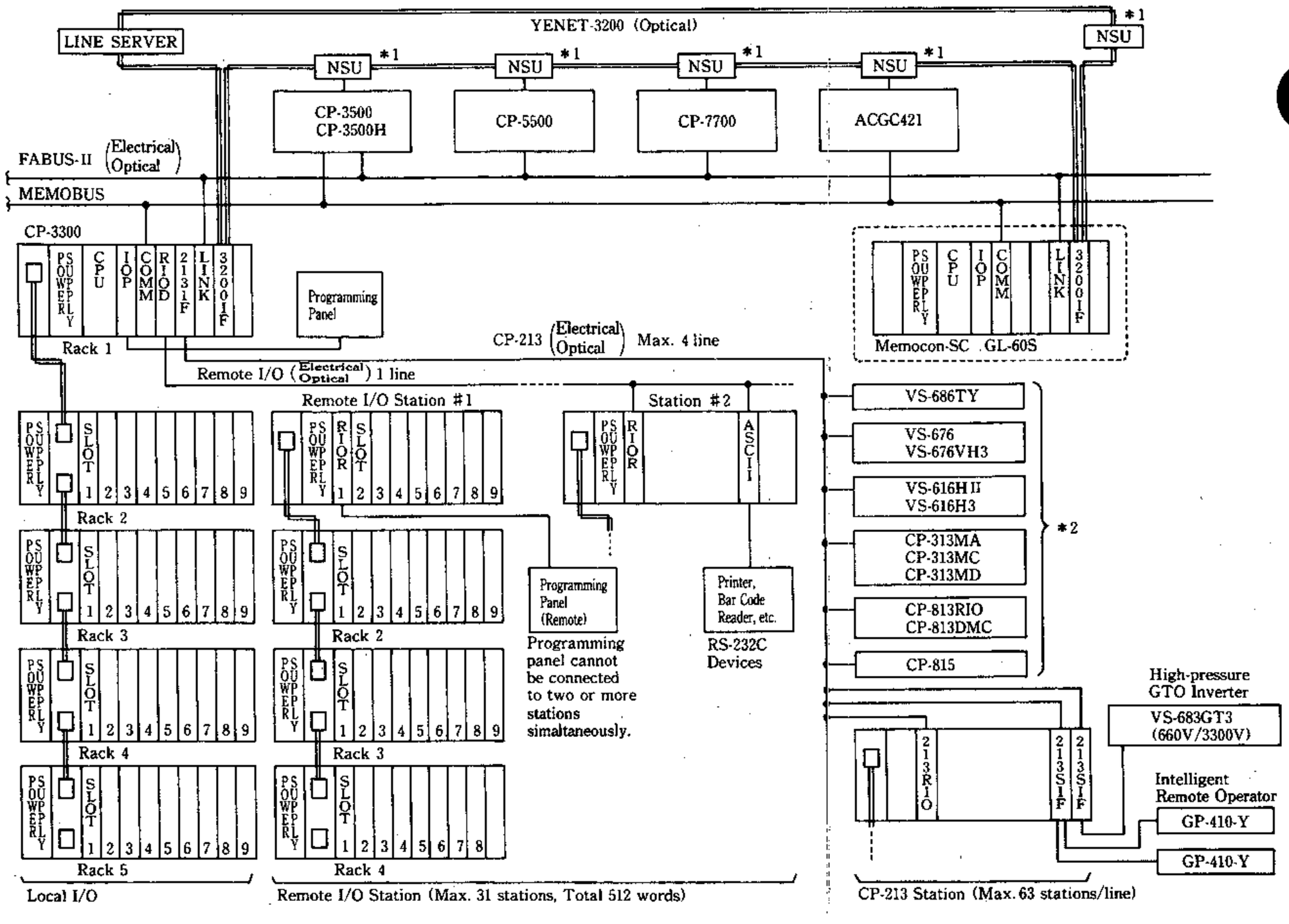


Fig. 1.2 CP-3300 system configuration

Note: In local I/O rack 1, if a transmission module is not needed a maximum of 6 slots can be allocated to I/O.

*1 NSU denotes a network service unit.

*2 FDS (a flexible drive system) devices.

- Inverter
  VS-686TY
  VS-676
  VS-676VH3
  VS-616H II
  VS-616H3
- Small size controller
  CP-313MA
  CP-313MC
  CP-313MD
- Intelligent I/O
  CP-813RIO
  CP-813DMC
  CP-815
  CP-213RIO
  CP-213SIF

The local I/O consists of the second through the fifth rack of the local channel. If there are unused slots in the first rack, that space can be used for local I/O. A 2000 series I/O module is used for an I/O module of local I/O. Racks are interconnected vertically with specialized I/O cables. The local I/O supports a maximum of 512 words for both input and output respectively.

Next to the CPU module there is an IOP module. The IOP module provides the following functions:

- Local I/O interface
- MEMOBUS interface

The MEMOBUS interface section is equivalent to the COMM module. The CP-750 programming panel is usually connected to the MEMOBUS interface connector PORT1. In the system configuration example of Fig. 1.2, MEMOBUS transmission is used for data trasmission with ACGC-421.

The CP-3300 is connected to a host system CP-3500, a CP-5500, other CP-3300 on the same level as itself, and a Memocon-SC GL60S via an FABUS-II interface module. The transmission speed is 4M bps and the transmission path is a coaxial cable.

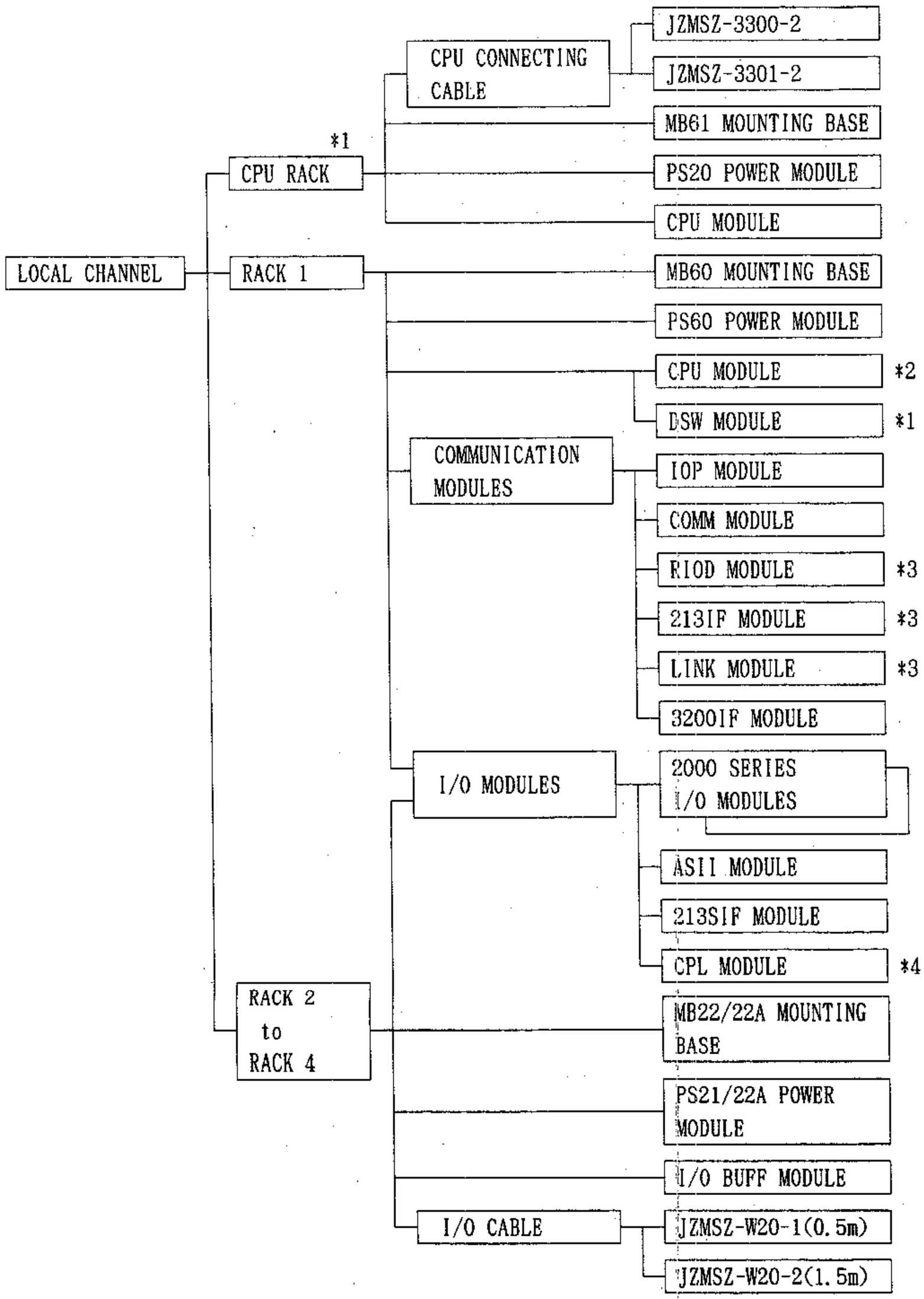
In CP-3300, up to 31 remote channels (maximum 512 words) can be connected via a remote I/O line. A RIOD module is provided for this interface. As a transmission path, a coaxial cable is used to connect the bus. A distance of up to 1 km can be covered by such transmission.

The remote channel is configured the same as the local channel except that the RIOR module is mounted on the first rack.

The CP-3300 can be connected to FDS equipment via CP-213 transmission. Up to four 213IF modules providing CP-213 transmission interface can be mounted. Namely, up to four lines are allowed in the CP-213 transmission path with a maximum of 63 stations per line(except for master station).
Allocate FDS equipment to stations from 1 to 31.
Allocate CP-3300 or CP-813 remote I/O to stations from 32 to 63.

1.2 MODULE CONFIGURATION

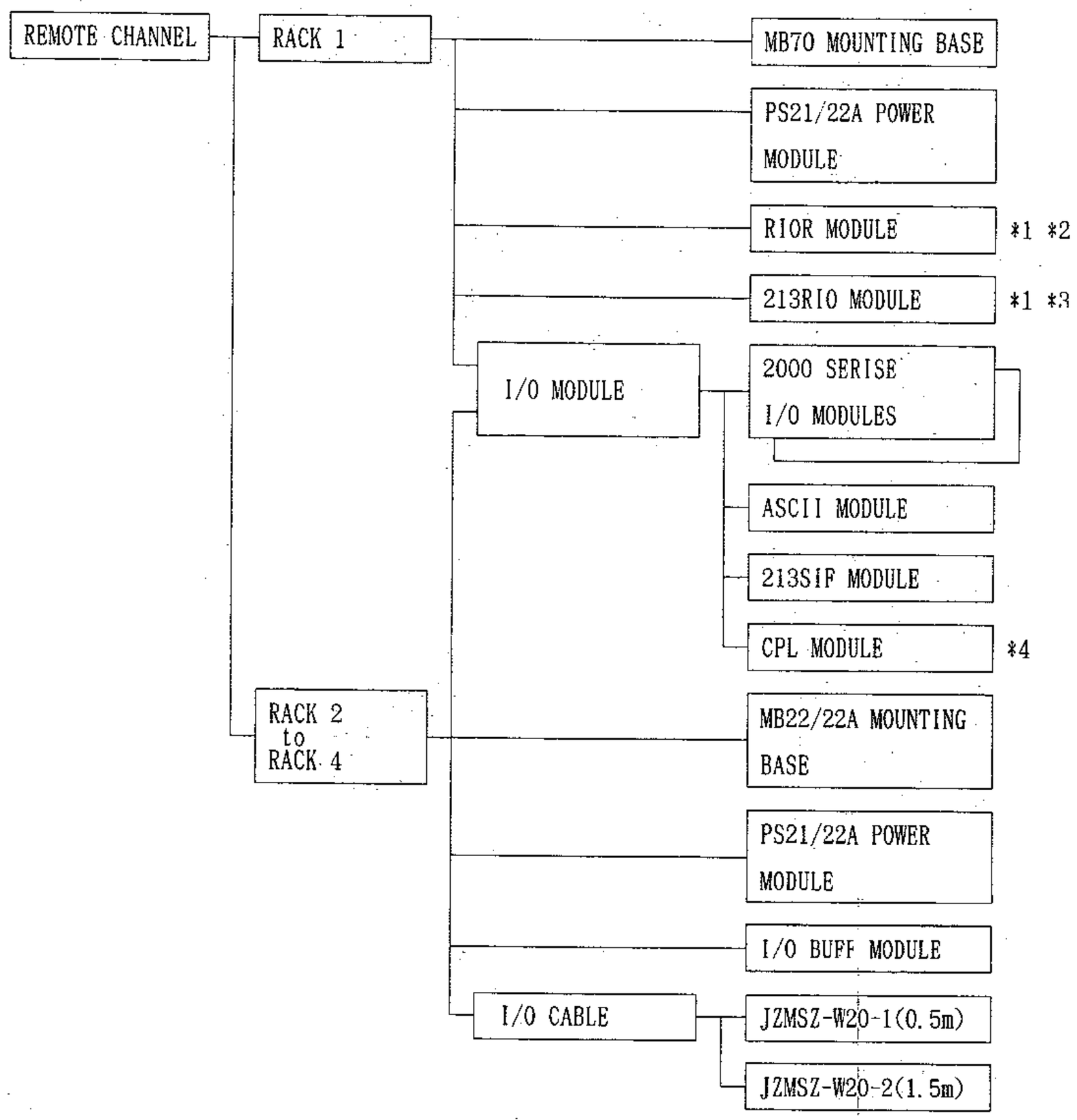
The CP-3300 is constructed with various modules as building blocks. Fig. 1.3 shows these modules in systematic configuration. The system designer will choose whichever are necessary of them to build a system desired.



*1 Required upon dual CPU system
*2 Not Required upon dual CPU system
*3 Select electrical or optical transmission.
*4 Required when the topology is star or duplex star in optical transmission system.

Fig. 1.3 (a) CP-3300 module system diagram (local channel)

MODULE CONFIGURATION



*1 Select electrical or optical transmisson.
*2 Used on remote I/O line connection.
*3 Used on CP-213 line connection.
*4 Required when the topology is star or duplex star in optical transmission system.

Fig. 1.3 (b) CP-3300 module system diagram (remote channel)

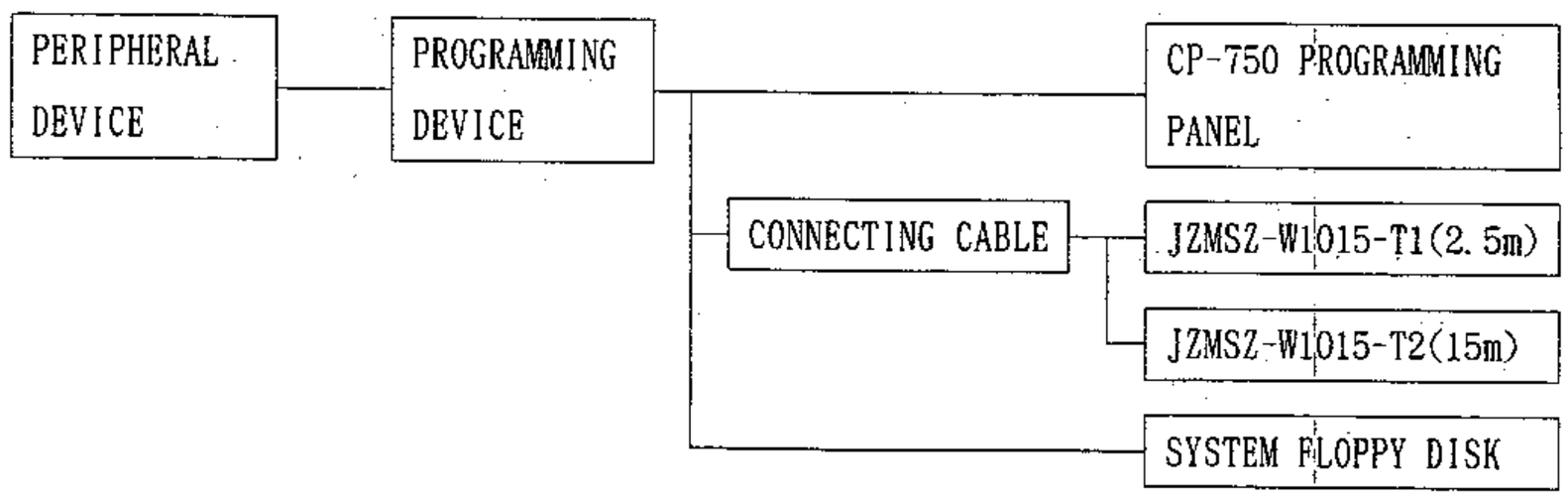


Fig. 1.3 (c) CP-3300 module system diagram (Peripheral device)

■ Number of optional modules mounted on racks

Optional modules (e.g. transmission module, I/O module) are mounted on each rack in CP-3300 system. However, the number of optional modules and the total current consumption have limits.

Be sure to calculate the total current consumption for mounted optional modules, referring to Fig. 1.4.

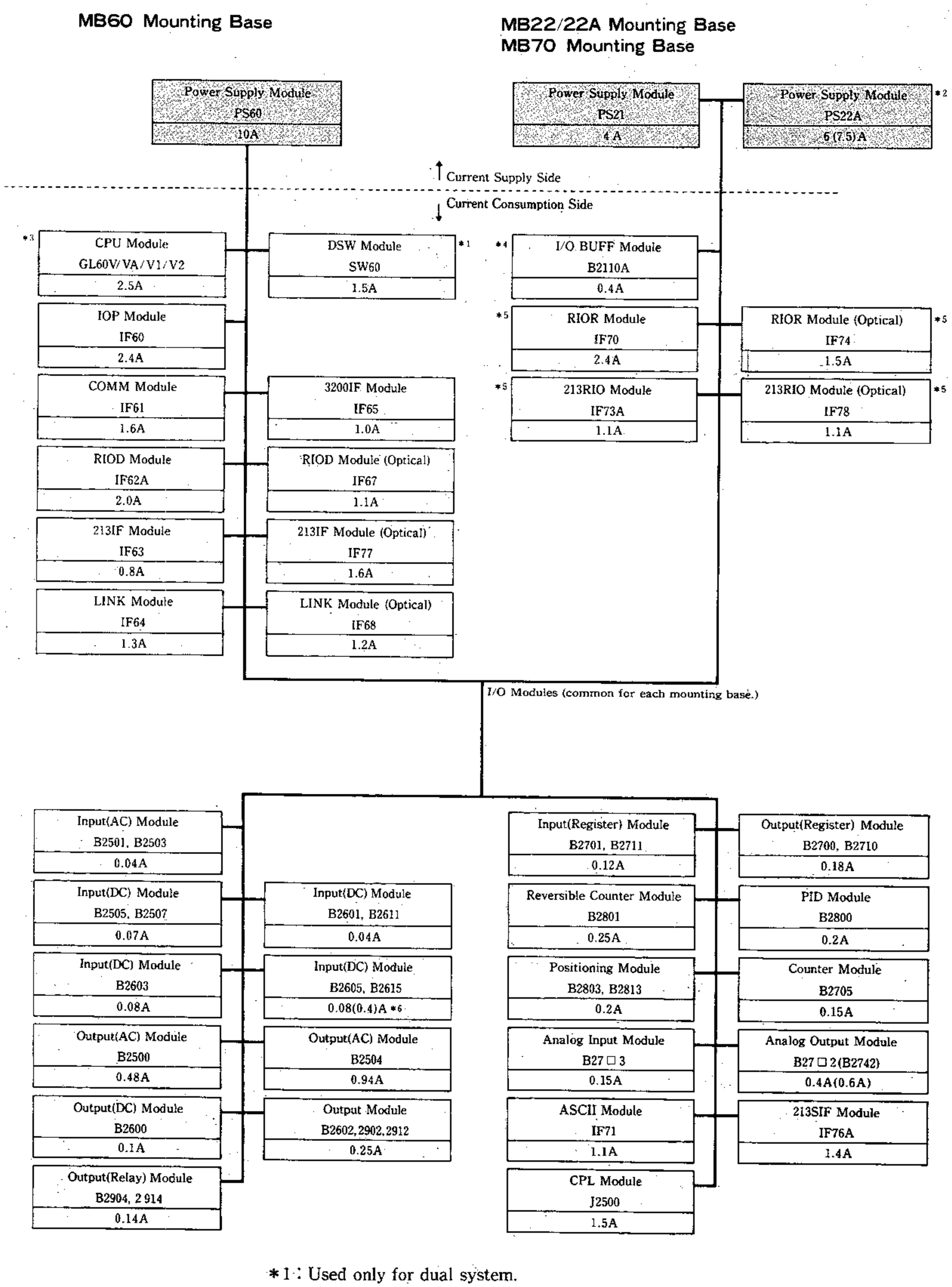
For I/O modules not described in Fig. 1.4, refer to the instruction manual.

Table 1.1 Number or Optional Modules Mounted on Racks

Rack	Mounting Base to be Used	Max. Number of Optional Modules Mounted
CPU rack	MB61	0*
Rack 1 in local I/O	MB60	6
Rack 1 in remote I/O	MB70	8
Racks 2 to 5 in local I/O Racks 2 to 4 in remote I/O	MB22 MB22A	9

* Only CPU module is mounted on MB61 mounting base.

MODULE CONFIGURATION



*1 : Used only for dual system.
*2 : 6A in MB22 mounting base, 7.5A in MB22A mounting base.
*3 : GL60V/VA is used for dual CPU system,
GL60V1/V2 is used for single CPU system.
*4 : Mounted on MB22/22A mounting base.
*5 : Mounted on MB70 mounting base.
*6 : 0.4A when all points are converted simultaneously.

Fig. 1.4 Module Current Consumption

2 CP-3300 SPECIFICATIONS

This section describes the major specifications of the CP-3300.

The general, hardware, and software specifications are listed separately.

CONTENTS PAGE

2.1 GENERAL SPECIFICATIONS

Table 2.1 General specifications

Item	Specification	Remarks
Input power	Single-phase, 85 to 132 VAC, 47.5 to 63 Hz	
Power consumption	150 VA	
Rush current	Less than 30 A	
Power failure handling	Outage handling performed if a voltage less than 80 VAC lasts 10 ms. (A duration not more than 0.5 cycle is not considered an outage.)	
	Data are held 5000 hours or more for both PM and DM. (Battery backup)	
Ground	Ground resistance less than 100 ohms (Specialized)	
Operating ambient temperature	0 to+55℃	Peripheral device excluded.
Storage temperature	−20 to +85℃	Lithium cell excluded.
Humidity	30 to 95%RH (non condensing)	
Vibration resistance	Conform to JIS *C0911.	Peripheral device excluded.
Shock resistance	JIS *C0912 (max. 98 m/s^2)	Peripheral device excluded.
Dielectric strength	1500 VAC for one minute between all power input terminals and the case	
Insulation resistance	1 MΩ (500 VDC) between all power input terminals and the cace	
Noise immunity	Noise voltage of 1500 Vpp, width of 1 μs, rise and fall times of 1 ns (by noise simulator)	
Atmosphere	No corrosive or inflammable gases	
Cooling system	Forced cooling (air flow speed of 1.5 m/s or more)	

* Japanese Industrial Standard

2.2 HARDWARE SPECIFICATIONS

Table 2.2 Hardware specifications

Item	Specification		
Program memory	Source program		380 k bytes
	Object program		272 k bytes (including constant registers max. 512 words/DWG and local variables max. 8 k words/DWG)
Data memory	System variable	S	768 words
	Input variable	I	4 k words
	Output variable	O	4 k words
	Global variable	M	16 k words
File memory	Trace buffer		4 k words ×2
I/O	Local I/O		1 channel, 512 words
	Remote I/O		1 line, 31 stations, 512 words
	CP-213		4 lines, 31 stations/line, 2 k words
	Link I/O		1 line, 31 stations, 1 k words
	I/O Equipment	General	2000 or 1000 series I/O
		FDS Equipment	CP-313M, CP-313MA, CP-313MB, CP313MC, CP-313MD, CP-813RIO-01M, CP-813DMC-02M, CP-815, 213RIO, 213SIF
		Inverter	VS-686TY, VS-676, VS-676VH3, VS-616HⅡ, VS-616H3, VS-683GT3 (660V/3300V)
Programming Panel CP-750	Max. 3 units connectable (each one unit for IOP, COMM, RIOR)		
Transmission Function	MEMOBUS		1200/2400/4800/9600/19200pbs
	FABUS-Ⅱ		4 Mbps
	YENET		2 Mbps
	CP-213		1 Mbps

2.3 SOFTWARE SPECIFICATIONS

Table 2.3 Software specifications

No.	Item	Specification
1	Program execution method	Two-level (high/low speed) constant period scanning Scan setting: 3 to 300 ms (in 1 ms steps) Interrupt processing drawing Batch processing drawing
2	Basic instruction set Sequence instruction Operation instruction Control instruction Built-in function DDC instruction SFC instruction System standard function User defined function	 7 types 33 types 10 types 10 types 12 types 6 types 20 types 100 types can be defined.
3	Basic data type	Relay, integer, real number
4	Variable specification method	Specifying a relay/register number Specifying a symbol (up to 8 alphanumeric characters)
5	Number of drawings Number of steps/DWG DWG type DWG hierarchy Operation error drawing	Max. 300 DWGs 300 steps/DWG (equivalent to functions) A: Start; H: High speed; L: Low speed; B: Batch; I: Interrupt 3 layers (×99.99) Detailed drawing number for each drawing type = 00
6	Basic instruction execution time Sequence instruction Arithmetic operation	 Approx 0.3 μs Approx 0.375 to 3.5 μs (integer); Approx 7.875 to 27.5 μs(real)

3 COMPONENT MODULES

This section briefly explains the specifications for the modules making up the CP-3300 as well as their functions and switch settings.

The component modules are the power, CPU, communication, and I/O modules.

CONTENTS PAGE

3.1 POWER MODULES

CP-3300 power modules come in three types. PS60 is used for a CPU base in single system, PS20 for CPU bases in dual system, and PS21 for an expanded I/O base.

3.1.1 PS60 main power module

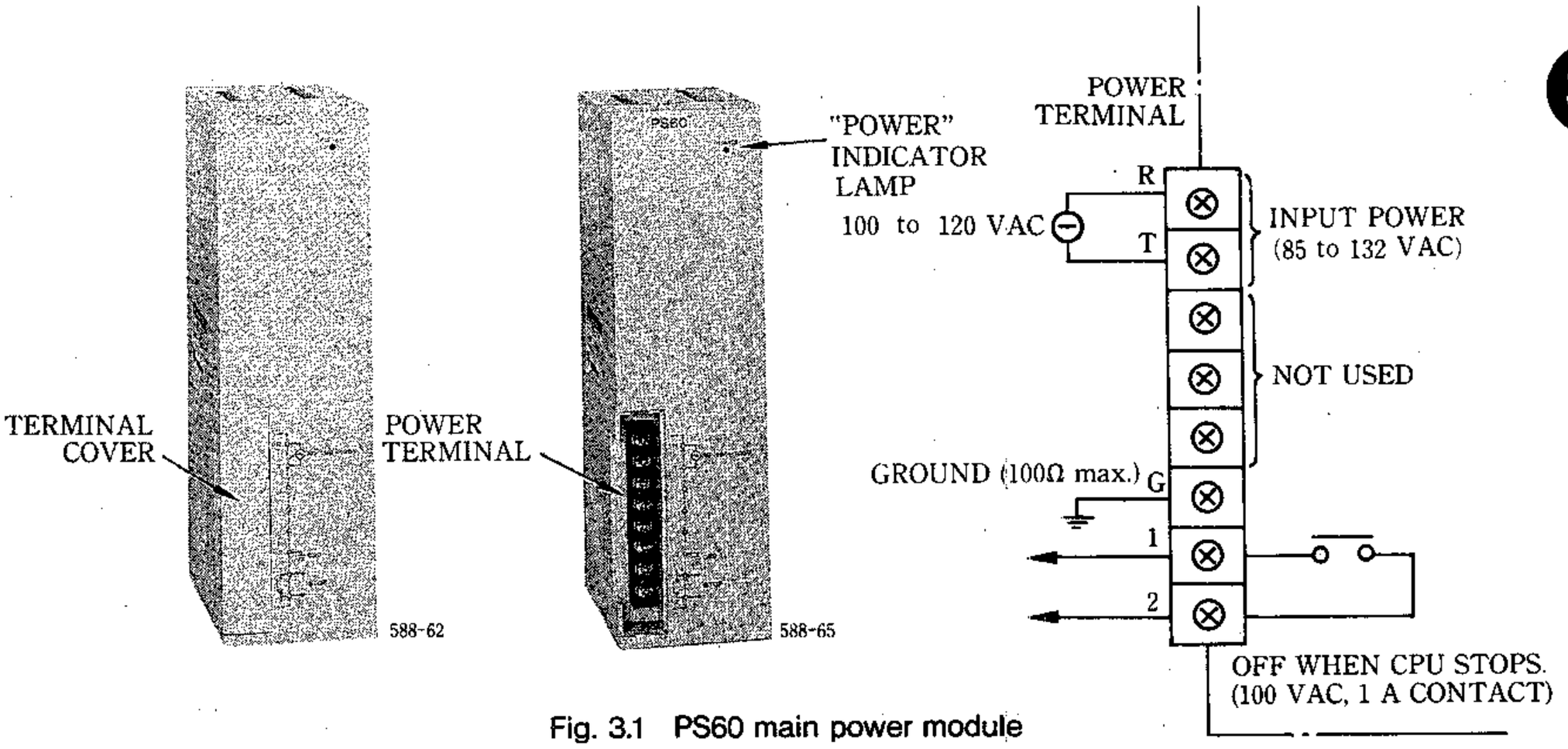


Fig. 3.1 PS60 main power module

Table 3.1 PS60 main power module specifications

Item	Specification
Type	JRMSP-PS60V
Function	Supplies necessary DC power to all modules mounted on a CPU base. It also monitors AC power and detects any power failure.
Input power	Single-phase, 85 to 132 VAC, 47.5 to 63 Hz, 150 VA
Transient input voltage	0 to 154 VAC (10 ms)
Rush current	Less than 30 A (peak)
Leak current	Less than 1 mA
Fuse	Normal melting type glass tube fuse 5A
Indicator lamp	POWER: Turned ON when power is normal.
Monitor contact	STOP: A contact is ON when CPU operates and OFF when CPU stops. 100 VAC, 1 A
Output current	5 VDC, 10A
Mounting base	MB60 mounting base (CPU base)
Overall size	75 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass	0.9 kg

3.1.2 PS21/22A auxiliary power module

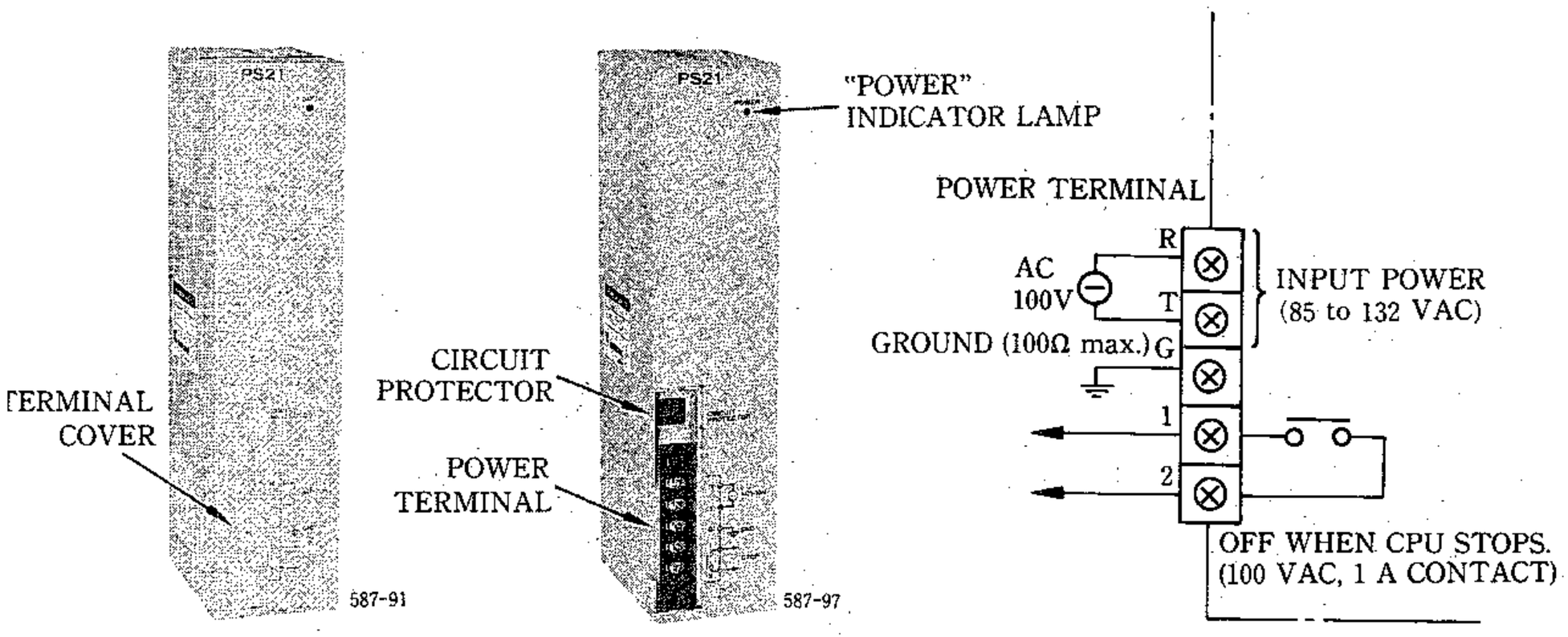


Fig. 3.2 PS21 auxiliary power module

Table 3.2 PS21/22A auxiliary power module specifications

Item	Specification		
Type	JRMSP-PS21V, JRMSP-PS22AV		
Function	Supplies necessary DC power to the modules (I/O BUFF, RIOR, 213R10, and I/O).		
Input power	Single-phase, 85 to 132 VAC, 47 to 63 Hz, 70 VA (for PS21), 100VA (for PS22A)		
Transient input voltage	0 to 154 VAC (10 ms)		
Rush current	Less than 30 A (peak)		
Leak current	Less than 0.2 mA		
Fuse	Circuit protector 3A		
Indicator lamp	POWER: Turned ON when power is normal.		
Output current		Mounting Base	
	Module Types	MB22	MB22A
	PS21	5VDC, 4A	5VDC, 4A
	PS22A	5VDC, 6A	5VDC, 7.5A
Monitor contact	STOP: A contact that is ON when CPU operates and OFF when CPU stops. 100 VAC, 1 A		
Mounting base	MB22, MB22A, or MB70 mounting base		
Overall size	60mm (w) ×250 mm (h) ×94 mm (d)		
Approximate mass	0.7 kg		

3.1.3 PS20 auxiliary power module

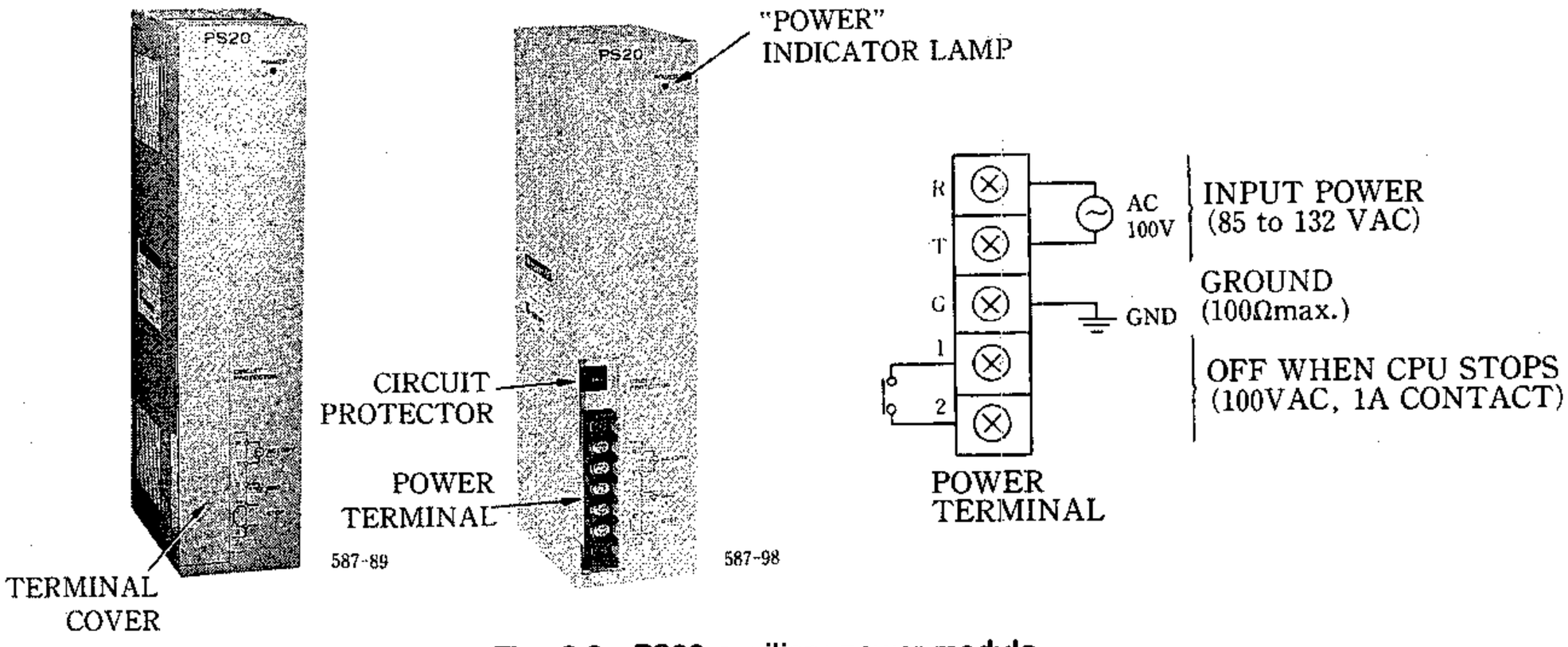
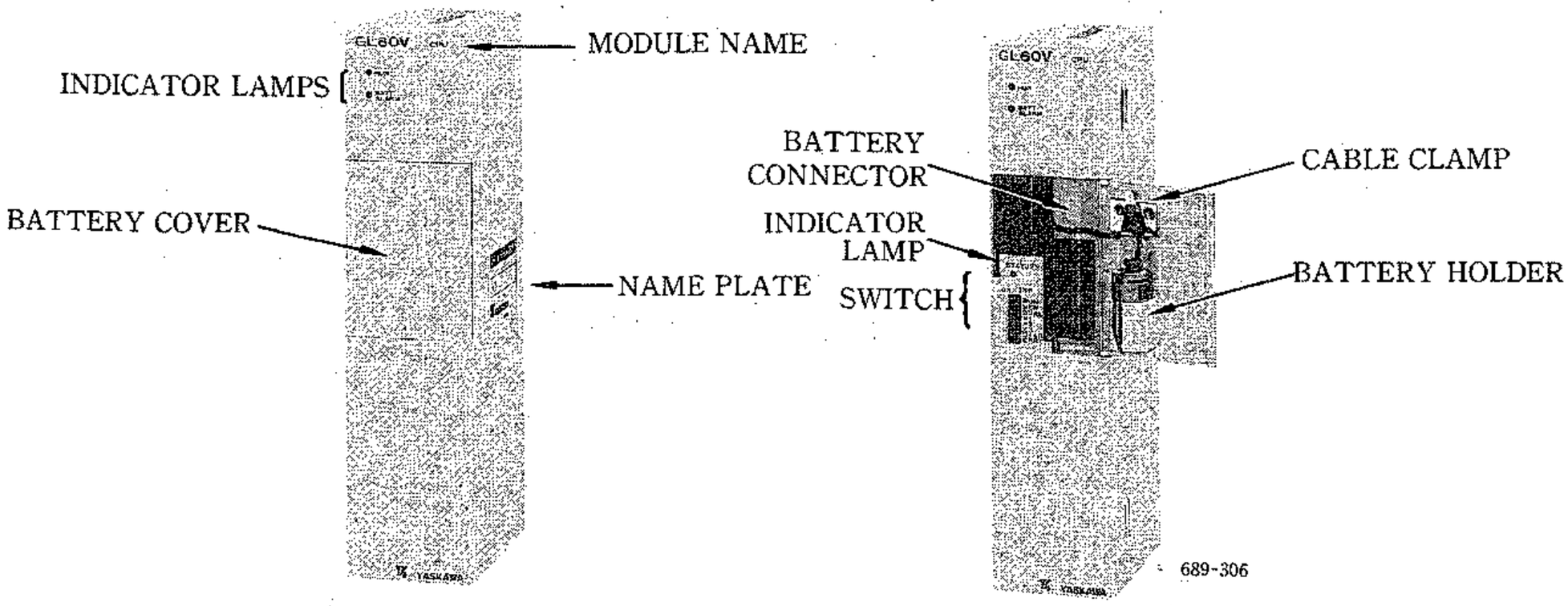


Fig. 3.3 PS20 auxiliary power module

Table 3.3 PS20 auxiliary power module specifications

Item	Specification
Type	JRMSP-PS20V
Function	Supplies necessary DC power to the CPU on the MB61 mounting base.
Input power	Single-phase, 85 to 132 VAC, 47 to 63 Hz, 100 VA
Transient input voltage	0 to 154 VAC (10 ms)
Rush current	Less than 30 A (peak)
Leak current	Less than 1 mA
Fuse	Circuit protector 3A
Indicator lamp	POWER: Turned ON when power is normal.
Monitor contact	STOP: A contact that is ON when CPU operates and OFF when CPU stops.
Output Current	5VDC, 5.5A
Mounting base	MB61 mounting base
Overall size	60 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass	1.0 kg

3.2 CPU MODULE



(1) WHEN BATTERY COVER IS CLOSED

(2) WHEN BATTRY COVER IS OPENED

Fig. 3.4 CPU module

Table 3.4 CPU module specifications

Item		Specification	
Type		DDSCR-GL60V, -GL60VA	Single CPU system
		DDSCR-GL60V1, -GL60V2	Dual CPU system
Function		A module at the heart of the CP-3300 that performs user program execution, I/O control, and programming panel services. It contains program and data memory.	
Memory	Program memory	272 k bytes	Backed up by battery
	Data memory	56.8 k bytes	
	Source program	380 k bytes	
Program execution method		2-level constant period scanning (scan time: 3 to 300 ms, set up in 1 ms steps) Interrupt and batch processing also supported.	
Operation function	Sequence instruction	7 types	
	Numeric value operation instruction	33 types	
	Control instruction	10 types	
	Built-in function	10 types	
	DDC instruction	12 types	
	SFC instruction	6 types	
	System standard function	20 types	
	User defined function	100 types are registered.	
Data type		Relay type : ON/OFF	
		Integer type : −32,768 to +32,767	
		Real number type : $\pm(1.17 \times 10^{-38}$ to $3.40 \times 10^{38})$ and 0	

Table 3.4 CPU module specifications (Cont'd)

Item		Specification
Watchdog timer time limit		Not less than 420 ms
Power requirement	Main power	5 VDC ± 3%, 2.5 A (typ)
	Backup power	Not less than 2VDC, 410 μA (typ), 810 μA (max.)
Mounting base		MB60 (in single CPU system), MB61 (in dual CPU system)
Overall size		60 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass		0.8 kg

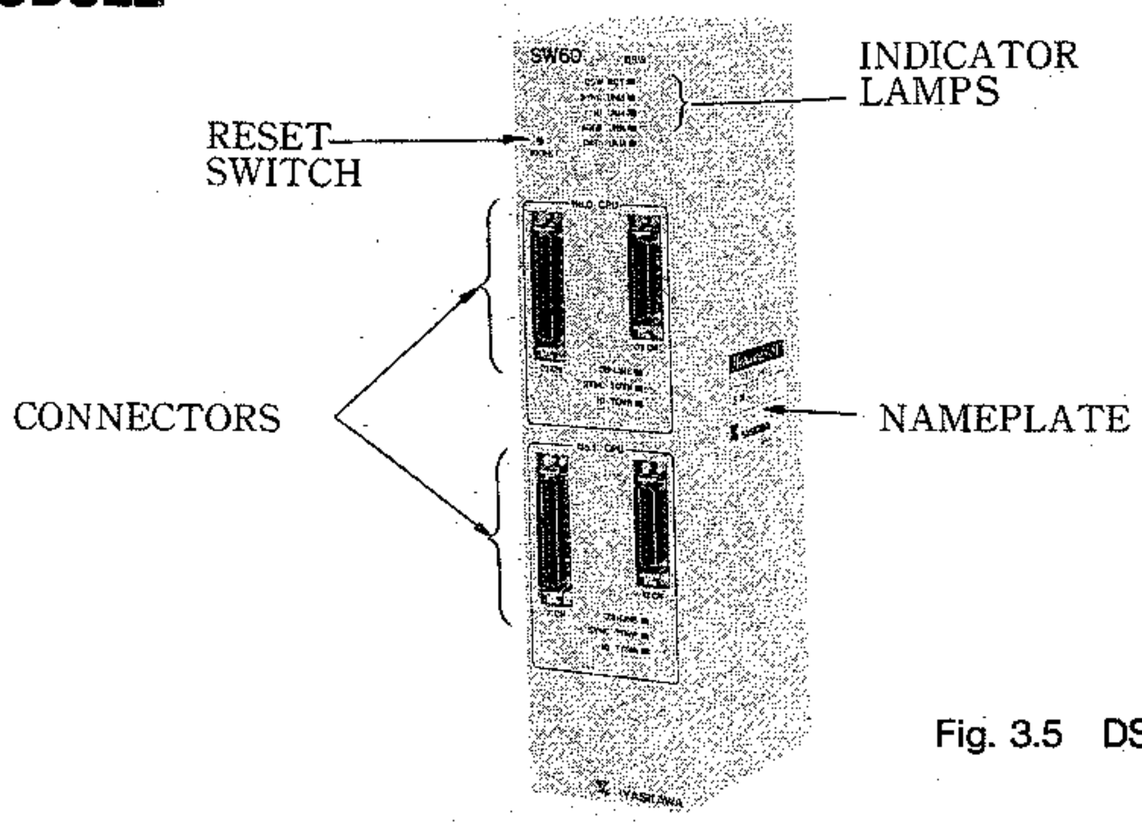
Table 3.5 Indicator lamps

Name	Normal status	Description	Remarks
RUN	Lit	Lights when CP-3300 system is operating normally.	—
BATT ALARM	OFF	Lights if the battery voltage is dropped. Replace the battery (within one month).	—
STATUS	OFF	Lights or blinks if an error occurs.	This indicator lamp can be seen by opening the battery cover.

Table 3.6 Dip switch 1SW

Name	Standard	Function	
		Setting	Setting Description
RUN	ON	ON	Operates the system.
		OFF	Stops the system.
NEW	Depending on the system	ON	Starts new operation.
		OFF	Starts continued operation.
DUAL	Depending on the system	ON	Dual CPU system
		OFF	Single CPU system
X1	OFF	ON	Not used. (Set to OFF.)
		OFF	
X2	OFF	ON	Resets CPU.
		OFF	—
X3	OFF	ON	Initializes memory when the CPU is started after turning this switch ON.
		OFF	
DIAG	OFF	ON	Executes self diagnosis.
		OFF	Does not execute self diagnosis. (Set to OFF during system operation.)
ERST	OFF	ON	To reset an error, turn this switch ON once and then turn it OFF.
		OFF	

3.3 DSW MODULE



692-38

Fig. 3.5 DSW module

Table 3.7 DSW module specifications

Item	Specification
Type	JAMSC-SW60V
Function	Performs synchronism detection of two CPU modules in dual CPU system.
Current consumption	5VDC, 1.5A
Mounting base	MB60 mounting base (CPU base)
Overall size	60 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass	0.6 kg

Table 3.8 Switch

Name	Function
RESET	Resets the DSW module and all the modules on the MB60 mounting base. Normally, do not depress this switch.

Table 3.9 Connectors

Name		Function
No. 0 CPU	01CN	Connects to the MB61 mounting base for No. 0 CPU by using cable JZMSZ-3301-2.
	02CN	Connects to the MB61 mounting base for No. 0 CPU by using cable JZMSZ-3300-2.
No. 1 CPU	11CN	Connects to the MB61 mounting base for No. 1 CPU by using cable JZMSZ-3301-2.
	12CN	Connects to the MB61 mounting base for No. 1 CPU by using cable JZMSZ-3300-2.

Table 3.10 Indicator lamps

Name		Normal Status	Function
DSW RDY		Lit	Lights when DSW module normal.
SYNC UNM		OFF	Lights when synchronous levels are not matched between both CPUs.
ID UNM		OFF	Lights when slave modules selected by both CPUs are not matched at I/O operation.
ADDR UNM		OFF	Lights when addresses from both CPUs are not matched at I/O operation.
DATA UNM		OFF	Lights when data from both CPUs are not matched at I/O operation.
No. n CPU	ON-LINE	Lit	Lights when No. n CPU is under online operation.
	SYNC TOVR	OFF	Lights when level synchronization request has not come from the other CPU within a certain time.
	ID TOVR	OFF	Lights when slave module selection request has not come from the other CPU within a certain time.

Note: n=0 or 1, corresponding to CPU No.

3.4 COMMUNICATION MODULES

3.4.1 IOP module

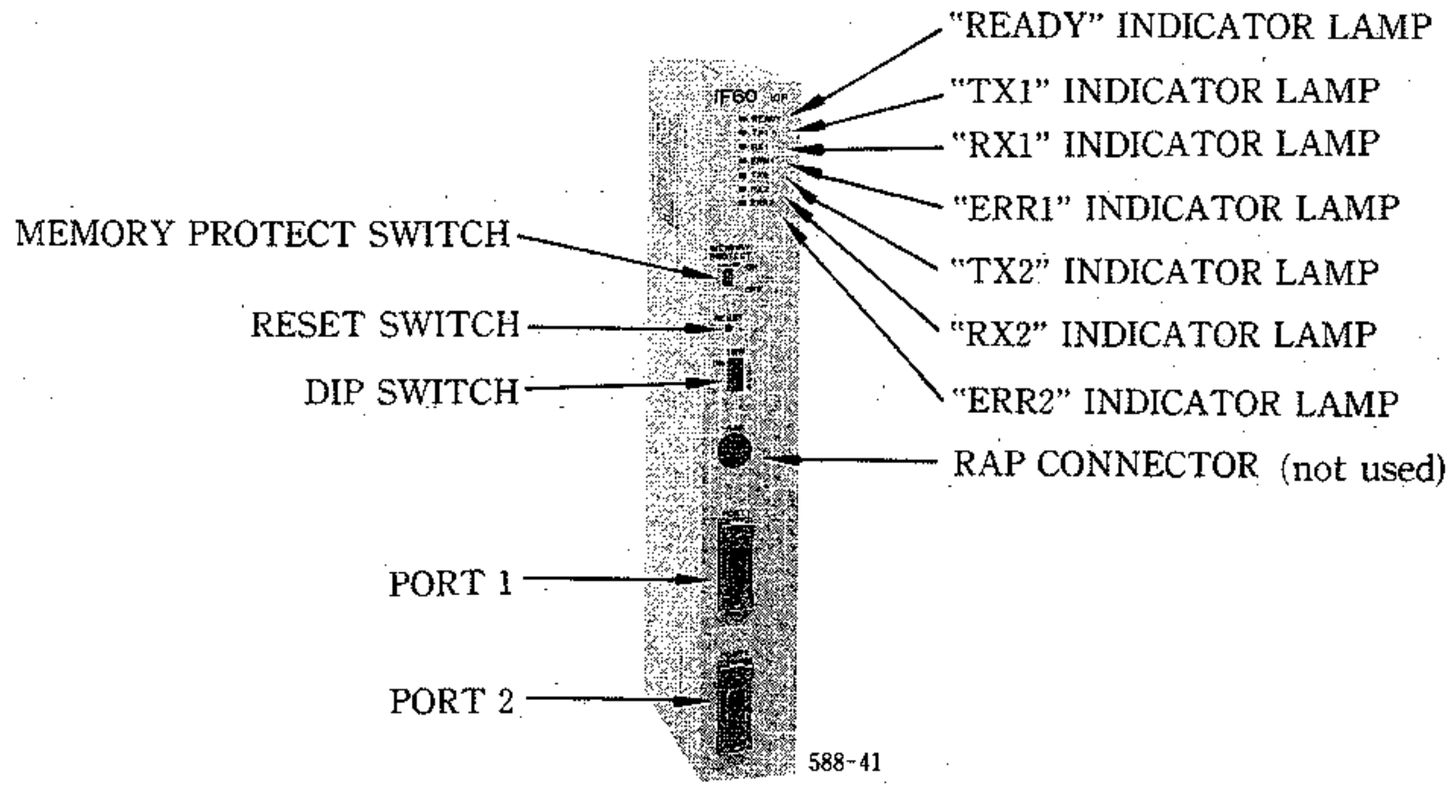


Fig. 3.6 IOP module

Table 3.11 IOP module specifications

Item		Specification
Type		JAMSC-IF60V
Function		Functionally, this module is the COMM module to which a local I/O driver and a register access panel interface are added. The module has two MEMOBUS ports (slave) for communication with a PP and host computer. The register access panel cannot be used in the CP-3300 system.
Communication port	Number of ports	2
	Transmission specification	EIA RS-232C
	Transmission speed	19,200/9,600/4,800/2,400/1,200/600/300/150 bps*
	Data bit	7 or 8 bits
	Parity check	Odd/even/none
	Stop bit	1 or 2
	Transmission procedure	MEMOBUS protocol
	Transmission check	CRC-16 or LRC
	Connector	D-SUB (9 pins)

* Transmission speed for programming panel should be 1,200 bps or more. (standard : 9,600 bps)

Table 3.11 IOP module specifications (Cont'd)

Item		Specification
Indicator lamps	READY	ON when IOP is normal.
	TX1	ON when port 1 is sending.
	RX1	ON when port 1 is receiving.
	ERR1	ON when port 1 communication error occurs.
	TX2	ON when port 2 is sending.
	RX2	ON when port 2 is receiving.
	ERR2	ON when a port 2 communication error occurs.
Current consumption		5VDC, 2.4A
Mounting base		MB60 mounting base (CPU base)
Overall size		37.5 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass		0.6 kg

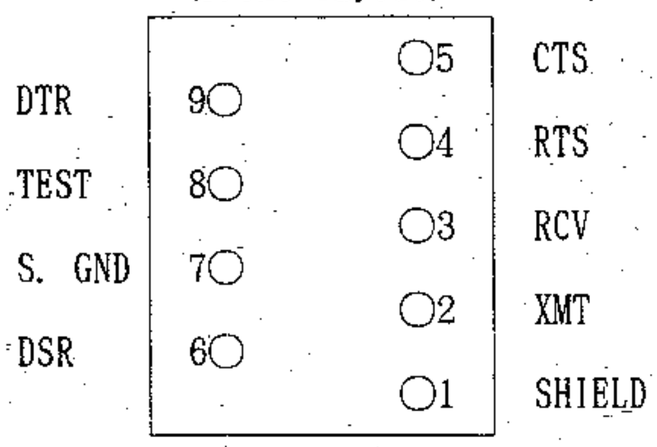
Table 3.12 Indicator lamp combinations

LED indicator			Error content
READY	ERR1	ERR2	
Blinking	Blinking	ON	ROM error
Blinking	ON	Blinking	RAM error
Blinking	Blinking	OFF	Common memory error
Blinking	Blinking	Blinking	Watchdog error

Table 3.13 Switches

Name		Standard	Function	
			Setting	Setting Description
MEMORY PROTECT		ON	ON	Prohibits writing to program memory.
			OFF	Permits writing to program memory.
RESET		——	——	Depressing this pushbutton switch initializes this module.
ISW	1	OFF	ON	Holds output status of the discrete output module when the CPU is stopped.
			OFF	Sets output status of all discrete output modules to OFF when the CPU is stopped.
	2	Depending on the system	ON	Sets port 1 to the transparent mode.
			OFF	Sets port 1 to the MEMOBUS mode.
	3	Depending on the system	ON	Sets port 2 to the transparent mode.
			OFF	Sets port 2 to the MEMOBUS mode.
	4	OFF	ON	Self diagnostic mode
			OFF	Normal operation mode

Fig. 3.7 PORT1, PORT2 connector
(front layout)



3.4.2 COMM module

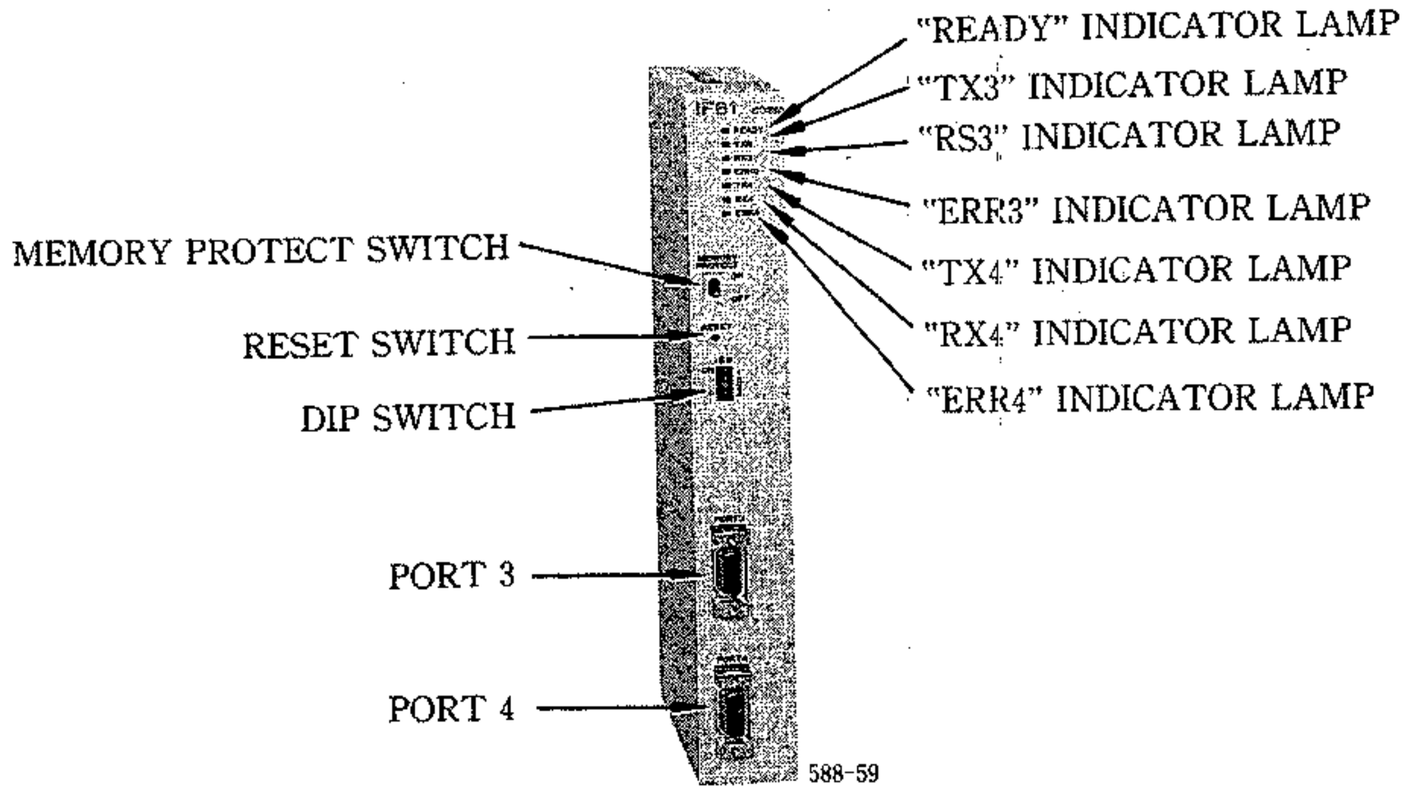


Fig. 3.8 COMM module

Table 3.14 COMM module specifications

Item		Specification
Type		JAMSC-IF61V
Function		Has two MEMOBUS ports (slave) for communication with a programming panel and a host computer.
Communication port	Number of ports	2
	Transmission specification	EIA RS-232C
	Transmission speed	19,200/9,600/4,800/2,400/1,200/600/300/150 bps*
	Data bit	7 or 8 bits
	Parity check	Odd/even/none
	Stop bit	1 or 2
	Transmission procedure	MEMOBUS protocol
	Transmission check	CRC-16 or LRC
	Connector	D-SUB (9 pins)

* Transmission speed for programming panel should be 1,200 bps or more.
(Standard : 9,600 bps)

Table 3.14 COMM module specifications (Cont'd)

Item		Specification
Indicator lamps	READY	ON when COMM is normal.
	TX3	ON when port 3 is sending.
	RX3	ON when port 3 is receiving.
	ERR3	ON when port 3 communication error occurs.
	TX4	ON when port 4 is sending.
	RX4	ON when port 4 is receiving.
	ERR4	ON when port 4 communication error occurs.
Current consumption		5VDC, 1.6A
Mounting base		MB60 mounting base (CPU base)
Overall size		37.5 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass		0.5 kg

Table 3.15 Indicator lamp combinations

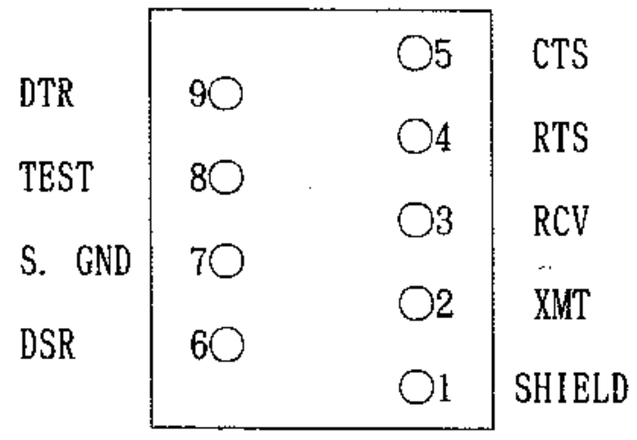
LED indicator			Error content
READY	ERR3	ERR4	
Blinking	Blinking	ON	ROM error
Blinking	ON	Blinking	RAM error
Blinking	Blinking	OFF	Common memory error
Blinking	Blinking	Blinking	Watchdog error

Table 3.16 Switches

Name		Standard	Function	
			Setting	Setting Description
MEMORY PROTECT		ON	ON	Prohibits writing to program memory.
			OFF	Permits writing to program memory.
RESET		——	——	Depressing this pushbutton switch initializes this module.
1SW	1	OFF	ON	Holds output status of the discrete output module when the CPU is stopped.
			OFF	Sets output status of all discrete output modules to OFF when the CPU is stopped.
	2	Depending on the system	ON	Sets port 3 to the transparent mode.
			OFF	Sets port 3 to the MEMOBUS mode.
	3	Depending on the system	ON	Sets port 4 to the transparent mode.
			OFF	Sets port 4 to the MEMOBUS mode.
	4	OFF	ON	Self diagnostic mode
			OFF	Normal operation mode

Fig. 3.9 PORT3, PORT4 connector

(front layout)



3.4.3 RIOD module (for electrical transmission)

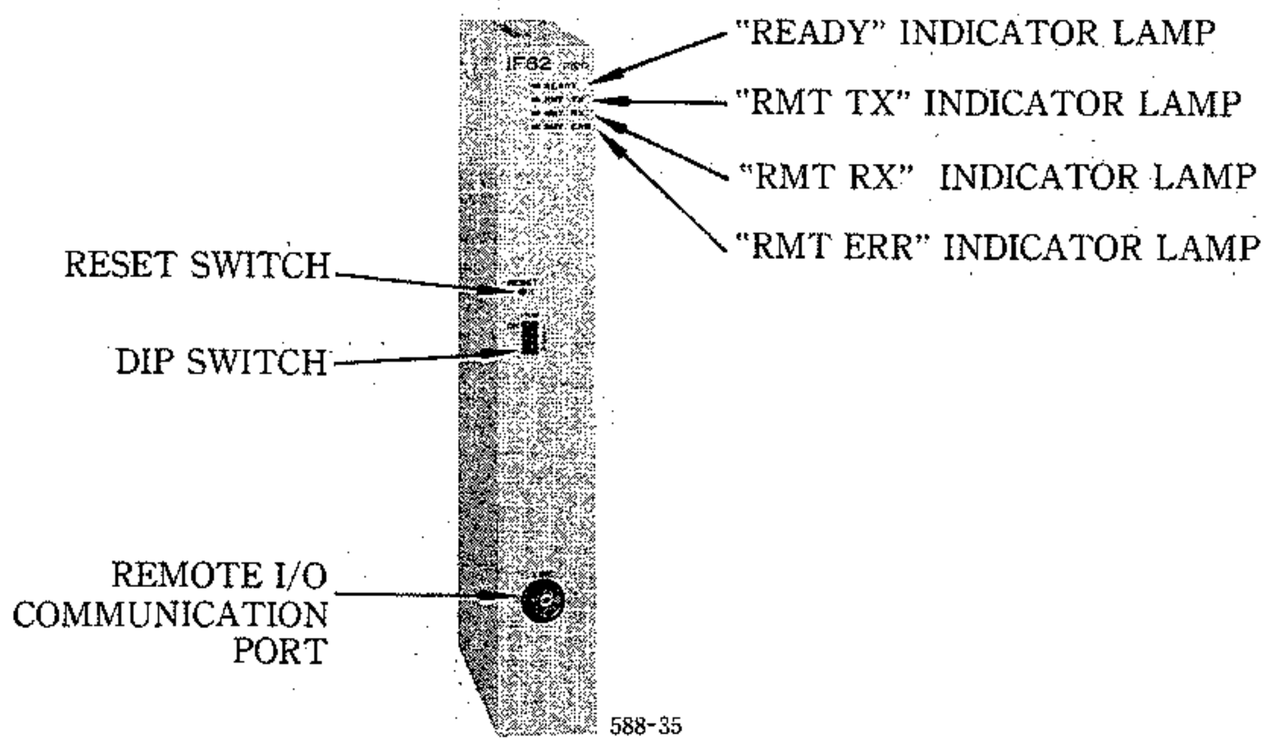


Fig. 3.10 RIOD module

Table 3.17 RIOD module specifications

Item			Specification
Type			JAMSC-IF62AV
Function			Required when setting up I/O modules at a remote site. Functions as a master station in a remote communication line. Also required when using an ASCII module.
Remote line	Topology (net status)		Bus
	Transmission medium		Coaxial cable
	Transmission method		Baseband (Manchester code)
	Transmission speed		0.5/1/2/4Mbps
	Max. cable length		1 km (when 11C-FB is used)
	Max. number of stations		31
	Troubleshooting		Automatic disconnect of a station in trouble, automatic reactivation after error recovery
Indicator lamp		READY	ON when RIOD normal.
		RMT TX	ON when remote sending in progress.
		RMT RX	ON when remote receiving in progress.
		RMT ERR	ON when remote communication error occurs.
Current consumption			5DVD, 2.0A
Mounting base			MB60 mounting base (CPU base)
Overall size			37.5 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass			0.5 kg

Table 3.18 Dip switch 1SW

<table>
<tr><th rowspan="2">Switch</th><th rowspan="2">Standard Setting</th><th colspan="2">Function</th></tr>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>1</td><td rowspan="2">ON</td><td>ON</td><td>Remote line 1</td></tr>
<tr><td>2</td><td rowspan="2">OFF</td><td>OFF</td><td>Remote line 2 (Remote I/O connection impossible)</td></tr>
<tr><td>3</td><td rowspan="2">Depending on the system</td><td>ON</td><td rowspan="2">Not used. (Set to OFF.)</td></tr>
<tr><td>4</td><td>OFF</td></tr>
</table>

				Setting			
Switch	3	OFF	ON	OFF	ON
	4	OFF	OFF	ON	ON
Transmission speed (Mbps)		0.5	1	2	4

3.4.4 RIOD module (for optical transmission)

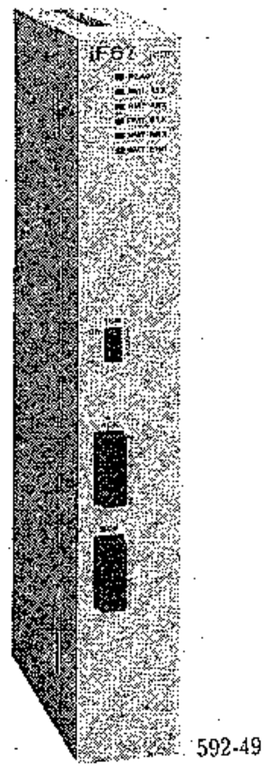


Fig. 3.11 RIOD module

Table 3.19 RIOD module specifications

Item		Specification
Type		JAMSC-IF67V
Function		Master station on remote communication line
Remote communication line	Topology	Cascade, star* , duplex star* (* CPL module is needed.)
	Transmission media	Optical fiber cable (Model H-PCF made by Sumitomo Electric Industries, Ltd.)
	Connector for optical transmission	Model DL-92 or DL-92H made by Sumitomo Electric Industries, Ltd. (Complies with JIS C 5977 F08 type.)
	Transmission method	Baseband (Manchester code)
	Transmission speed	4 Mbps (fixed)
	Transmission distance	12 km max. (850m for between stations)
	Number of stations	31 max.
	Troubleshooting	Automatic disconnect of a station in trouble, automatic reactivation after error recovery, double transmission paths
Indicating lamp	READY	Lights while RIOD module (for optical transmission) is normal.
	RMT ATX	Lights at bit stream of data sent from ACN port.
	RMT ARX	Lights at bit stream of data received in ACN port.
	RMT BTX	Lights at bit stream of data sent from BCN port.
	RMT BRX	Lights at bit stream of data received in BCN port.
	RMT ERR	Lights at remote transmission error such as timeout, incorrect data.
Current consumption		5VDC, 1.1A
Mounting base		MB60 mounting base
Overall size		37.5 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass		0.5 kg

Table 3.20 Dip switch 1SW

Switch	Standard setting	Function
1	ON	ON : Remote line 1 OFF: Remote line 2 (Remote I/O cannot be connected)
2	OFF	ON : Self-diagnosis mode (Normally, do not set to this mode.) OFF: Operation mode
3	*	ON : Duplex star mode OFF: Cascade mode
4	OFF	Not used. (Set to OFF.)

* : Set depending on the system.

3.4.5 213IF module (for electrical transmission)

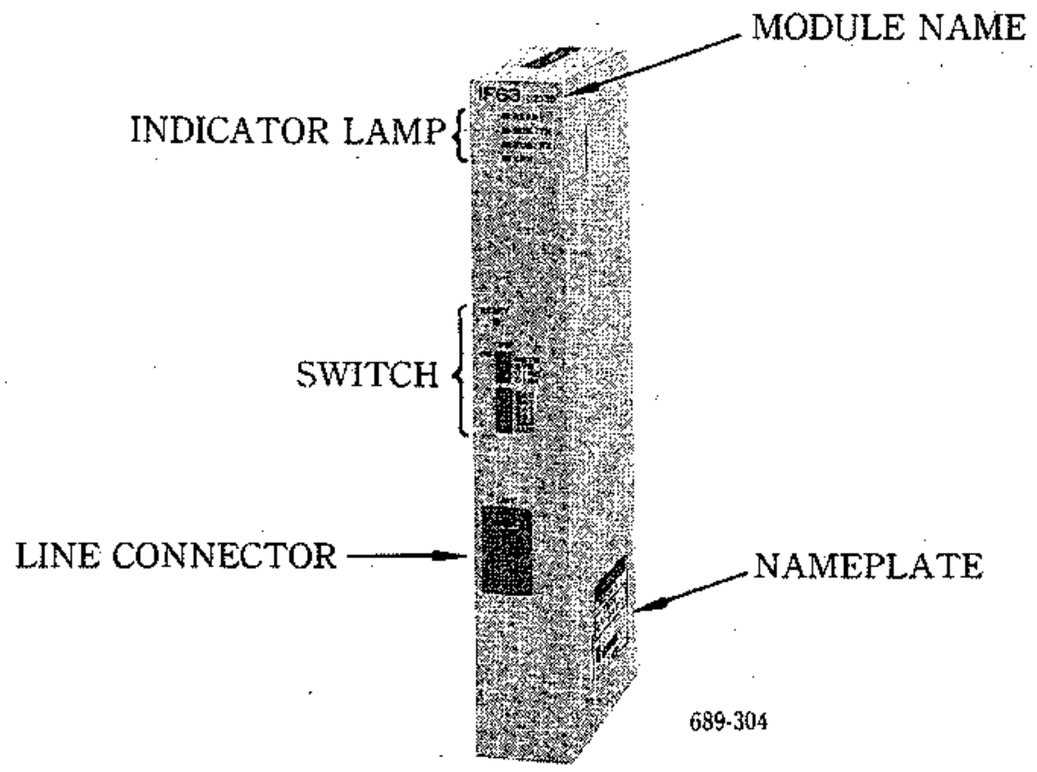


Fig. 3.12 213IF module

Table 3.21 213IF module specifications

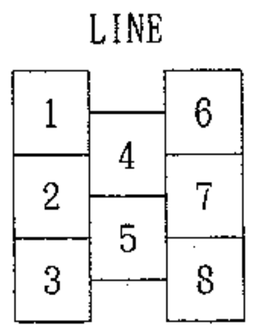
Item	Specification
Type	JAMSC-IF63V
Function	A CP-213 transmission interface of the CP-3300 system. It can be a master or slave station. As a master station, it provides both single-wire and three-wire interfaces. Using an external optical converter (CP-290), it supports building of an optical loop system. Up to four 213IF modules can be attached to the CPU base.
Number of lines	1
Connecting method	Electric bus (EIA RS-485) Optical loop (At outside, CP-290 is required.)
Mating connector	MR-8MG (connector), MR-8L (vertical case)
Transmission path	Three-wire bus : KV 0.75 mm² Single-wire bus : KPEV-S 1.25 mm² Multi-component optical fiber : OPCB
Transmission distance	Three-wire bus: max. 30 m Single-wire bus: max. 300 m Optical loop: max. 300 m (between stations)
Transmission speed	1 Mbps
Data exchange	1 : N
Transmission mode	Broadcasting, control transmission, message transmission
Number of attachable	As a master station, it can serve a total of 63 stations. As a slave station, up to 31 station number can be set up.
Mounting base	MB60 mounting base (CPU base)
Current consumption	5 VDC ± 3%, 0.82 A (typ)
Overall size	37.5 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass	0.5 kg

Table 3.22 Indicator lamps

Name	Normal status	Description
READY	Lit	Indicates this module is normal and ready to accept an operation command from the CPU.
BUS TX	Lit when sending	Lights or blinks during signal transmission to a CP-213 line.
BUS RX	Lit when receiving	Lights or blinks during signal reception from a CP-213 line.
ERR	OFF	Lights when this module is faulty, an error is found in the received signal, or no response comes from a receiving station.

Table 3.23 LINE connector connection

Connector terminal number	Signal name	Function
1	DATA+	Send/receive data line.
8	DATA−	
2	I/O +	Switches send and receive operation. Used for a three-wire system and when connecting an external repeater.
5	I/O −	
4	SCLK+	A clock signal for CP-213 system.
7	SCLK−	Used for a three-wire system.
3		Not used. (Do not connect anything.)
6		



(FRONT VIEW)

Table 3.24 Switches (module front)

<table>
<tr><th rowspan="2">Type</th><th rowspan="2">Name</th><th rowspan="2">Standard Setting</th><th colspan="2">Function</th></tr>
<tr><th>Setting</th><th>Description</th></tr>
<tr><td>Pushbutton switch</td><td>RESET</td><td>——</td><td>——</td><td>Depress this switch to initialize this module. This switch need not normally be depressed.</td></tr>
<tr><td rowspan="11">1SW</td><td rowspan="2">MSTR</td><td rowspan="2">Depending on the system</td><td>ON</td><td>Makes this module a master station.</td></tr>
<tr><td>OFF</td><td>Makes this module a slave station. A station number is determined by SA0 to SA4.</td></tr>
<tr><td rowspan="2">SYN</td><td rowspan="2">ON</td><td>ON</td><td>Synchronizes the operation with CPU scanning.</td></tr>
<tr><td>OFF</td><td>The CP-231 performs synchronous operation independent of CPU scanning.</td></tr>
<tr><td>CIR0</td><td rowspan="2">Depending on the system</td><td rowspan="2">——</td><td rowspan="2">Sets up a line number.
<table><tr><th>Line No.</th><th>0</th><th>1</th><th>2</th><th>3</th></tr><tr><td>CIR0</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr><tr><td>CIR1</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr></table></td></tr>
<tr><td>CIR1</td></tr>
<tr><td>SA0</td><td rowspan="5">Depending on the system</td><td rowspan="5"></td><td rowspan="5">Sets up a station number when the module is used as a slave station.
<table><tr><th>Station No.</th><th>1</th><th>2</th><th>3</th><th>……</th><th>31</th></tr><tr><td>SA0</td><td>ON</td><td>OFF</td><td>ON</td><td></td><td>ON</td></tr><tr><td>SA1</td><td>OFF</td><td>ON</td><td>ON</td><td></td><td>ON</td></tr><tr><td>SA2</td><td>OFF</td><td>OFF</td><td>OFF</td><td></td><td>ON</td></tr><tr><td>SA3</td><td>OFF</td><td>OFF</td><td>OFF</td><td></td><td>ON</td></tr><tr><td>SA4</td><td>OFF</td><td>OFF</td><td>OFF</td><td></td><td>ON</td></tr></table>Notes:
1. SA0 to SA4 are invalid when the MSTR switch is ON.
2. Setting all SA0 to SA4 to OFF is not allowed.</td></tr>
<tr><td>SA1</td></tr>
<tr><td>SA2</td></tr>
<tr><td>SA3</td></tr>
<tr><td>SA4</td></tr>
<tr><td>AUX</td><td>OFF</td><td>ON
OFF</td><td>Self diagnosis. Starting up with this switch at ON performs self diagnosis.</td></tr>
</table>

3.4.6 213IF module (for optical transmission)

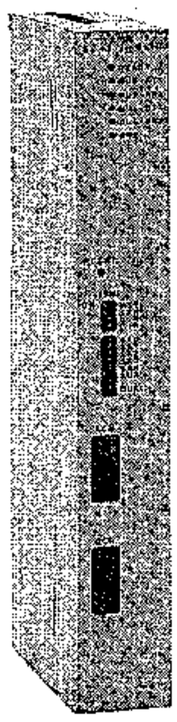

693-227

Fig. 3.13 213IF module

Table 3.25 213IF module specifications

Item		Specification
Type		JAMSC-IF77V
Function		This module becomes either a master station or a slave station as CP-213 optical transmission interface in CP-3300 system. Four 213 IF modules are mounted in a CPU base mounting.
Number of lines		1
Topology		Cascade, star* , duplex star* (* : CP-213 optical star coupler is needed.)
Transmission media		Optical fiber cable (model H-PCF made by Sumitomo Electric Industries Ltd.)
Connector for optical transmission		Model CF-2001H or CF-2011 made by Sumitomo Electric Industries Ltd. (Complies with JIS C5976 F07 type.)
Transmission distance		Between stations : 1000 m when using connector model CF-2001H 700 m when using connector model CF-2011 Total : 12 km
Transmission speed		1 Mbps
Data exchange		1 : N
Transmission mode		Broadcasting, control, message
Number of connectable modules		31 modules max. as a master station. Station addresses 1 to 31 can be set as a slave station.
	Cascade	18 modules max. 32 modules max. when can CP-213 optical star coupler is used. (8 modules max. / branch at one port)
	Star or Duplex star	32 modules max. (CP-213 optical star coupler is needed.)
Mounting base		MB 60 mounting base
Current consumption		5 VDC ± 3%, 1.6A
Overall size		37.5 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass		0.6 kg

Table 3.26 Indicating lamps (LED)

Name	Normal status	Description
READY	Lit	This module is normal, so it can receive operation commands from CPU.
BUS ATX	Lit or blinking	Signal is being sent to CP-213 line of ACN port.
BUS ARX	Lit or blinking	Signal is being received from CP-213 line of ACN port.
BUS BTX	Lit or blinking	Signal is being sent to CP-213 line of BCN port.
BUS BRX	Lit or blinking	Signal is being received from CP-213 line of BCN port.
ERR	OFF	Lights at module fault, receiving signal error, or no response from receiving station.

Table 3.27 Switches (module front)

Switch	Name	Standard setting	Function
Push button	RESET	—	Initialize this module by depressing this switch. Normally, do not depress this switch.
1SW	MSTR	*	ON : Makes this module a master station. OFF: Makes this module a slave station. The station number is decided by SA0 to SA4.
	SYN	ON	ON : Transmits synchronously with CPU scan setting. OFF: Transmits independently of CPU scan setting.
	CIR0 CIR1	*	Sets line number connecting this modules. Switch \ Line No. / 0 / 1 / 2 / 3 CIR 0 / OFF / ON / OFF / ON CIR 1 / OFF / OFF / ON / ON
	SA0 SA1 SA2 SA3 SA4	*	Sets station number as slave station. Switch \ ST No. / 1 / 2 / 3 / … / 31 SA0 / ON / OFF / ON / … / ON SA1 / OFF / ON / ON / … / ON SA2 / OFF / OFF / OFF / … / ON SA3 / OFF / OFF / OFF / … / ON SA4 / OFF / OFF / OFF / … / ON Notes : 1. When all MSTR and SA0 to SA4 switches are set to ON, it is a long distant mode. Other settings are standard distant mode. 2. Do not set all SA0 to SA4 to OFF.
	AUX	OFF	ON : Self-diagnosis mode (Normally, do not set to this mode.) OFF: Operation mode
	…	OFF	Not used (Set to OFF.)
	DUAL	*	ON : Duplex star mode OFF: Cascade mode

* Set depending on the system.

3.4.7 LINK module (for electrical transmission)

Fig. 3.14 LINK module

Table 3.28 LINK module specifications

Item	Specification
Type	JAMSC-IF64V
Function	Provides data and message transmission between PCs.
Topology (network form)	Bus (party line)
Media access control method	Token passing bus (n: n communication)
Transmission media	Coaxial cable (75Ω)
Transmission method	Baseband method (Manchester code)
Transmission speed	0.5, 1, 2, 4 Mbps (set up by DIP switch on the module front) Transmission speed selectable according to transmission distance.
Max. transmission distance	1 km (depending on the line speed and cable used)
RAS	Automatic unlinking of an error station Automatic relinking
Synchronization	Frame synchronization
Frame format	HDLC
Insulation	Pulse transformer
Number of stations	Max. 32 stations
Station address	1 to 32. Set up by rotary switch on the module front
Connector	BNC connector
Current consumption	5 VCDC, 1.3 A
Mounting base	MB60 mounting base (CPU base)
Overall size	37.5 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass	0.6 kg

Table 3.29 Indicator lamps

Name	Meaning	Color
READY	Indicates the IF64 is operating normally.	Green
PP TX	Goes ON when sending data from a PP port. This LED is turned ON by a send data bit stream.	Green
PP RX	Goes ON when receiving data from a PP port. This LED is turned ON by a receive data bit stream.	Green
PP ERR	Goes ON for about 10 ms when an error occurs in the PP port transmission (receiving). A parity error, overrun error, framing error, CRC error, illegal data received.	Red
LINK TX	Goes ON when sending data from a link port. This LED is turned ON by a send data bit stream.	Green
LINK RX	Goes ON when receiving data from a link port. This LED is turned ON by a receive data bit stream.	Green
LINK ERR	Goes ON for about 10 ms when an error occurs in the link port communication (receiving). A timeout error, illegal data received.	Red

All the indicators above are lit at power-on or reset time. Only READY is indicated after module self diagnosis. Other indicators are lit according to the communication condition.

Table 3.30 Dip switch

SW No.	Setting		
1	Be sure to turn this OFF. (Cannot operate normally if ON.)		
2	ON	Hold mode. Holds the link data of down station (*) in the preceding condition.	
	OFF	Clear mode. Turns the link data of down station (*) to "OFF" or "0".	
3, 4	3	4	Transmission speed
	ON	ON	4 Mbps
	OFF	ON	2 Mbps
	ON	OFF	1 Mbps
	OFF	OFF	0.5 Mbps

* A station unlinked from a transmission path due to some error such as power off.

3.4.8 LINK module (for optical transmission)

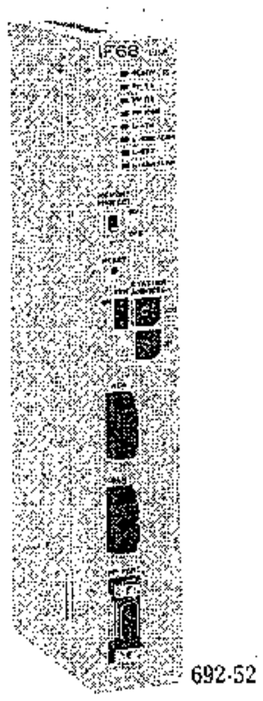


Fig. 3.15 LINK module

Table 3.31 LINK module specifications

Item		Specification
Type		JAMSC-IF68V
Function		Connects between PCs by using a special reference.
Transmission line	Topology	Cascade, star*, duplex star* (* CPL module is needed.)
	Media access	Token passing bus (n:n communication)
	Transmission media	Optical fiber cable (model H-PCF made by Sumitomo Electric Industries Ltd.)
	Connector for optical transmission	Model DL-92 or DL-92H made by Sumitomo Electric Industries Ltd. (Complies with JIS C5977 F08 type.)
	Transmission method	Baseband (Manchester code)
	Frame format	HDLC (High-level Data Link Control)
	Transmission speed	4 Mbps (fixed)
	Transmission distance	12 km max. (850 m for between stations)
	Number of stations	32 max.
	Troubleshooting	Automatic disconnect of a station in trouble, automatic reactivation after error recovery, double transmission paths.
Programming panel port (Note)	Transmission specification	Complies with EIA RS-232C.
	Synchronization	Half-duplex communication synchronization
	Transmission speed	9600 bps
	Character length	8 bits
	Stop bit	1 bit
	Parity check	Even parity
	Protocol	MEMOBUS
	Device address	Same as station address
	Connector	D-SUB (9 pins)
	Connectable device	Programming panel CP-750

Note: A programming panel port in LINK module cannot be used for CP-3300.

Table 3.31 LINK module specifications (Cont'd)

Item		Specification
Indicating lamps	READY	Lights while LINK module (optical) is normal.
	PP TX	Lights at bit stream of data sent from a programming panel port.
	PP RX	Lights at bit stream of data received in a programming panel port.
	PP ERR	Lights momentarily at receiving data error in a programming panel port.
	L-ATX	Lights at bit stream of data sent from ACN port.
	L-ARX/ERR	Green light indicates bit stream of data received in ACN port. Red light indicates receiving data error.
	L-BTX	Lights at bit stream of data sent from BCN port.
	L-BRX/ERR	Green light indicates bit stream of data received in BCN port. Red light indicates receiving data error.
Current consumption		5 VDC, 1.2A
Mounting base		MB60 mounting base
Overall size		37.5 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass		0.6 kg

Table 3.32 STATION ADDRESS rotary switch

Setting range		Station address range
×10	×1	
0	0	Cannot be set
	1 to 9	1 to 9
1 or 2	0 to 9	10 to 29
3	0 to 2	30 or 32
	3 to 9	Cannot be set
4 to 9	—	Cannot be set

Table 3.33 Switches

Switch		Standard Setting	Function
MEMORY PROTECT		OFF	Not used. (Set to OFF.)
RESET		—	Reset this module by depressing this switch. Normally, do not depress this switch.
1SW	1	OFF	ON : Self-diagnosis mode (Normally, do not set to this mode.) OFF: Operation mode
	2	OFF	ON : Hold mode for output OFF: Clear mode for output
	3	*	ON : Duplex star mode OFF: Cascade mode
	4	OFF	Not used. (Set to OFF.)

* : Set depending on the system.

3.4.9 3200 IF module

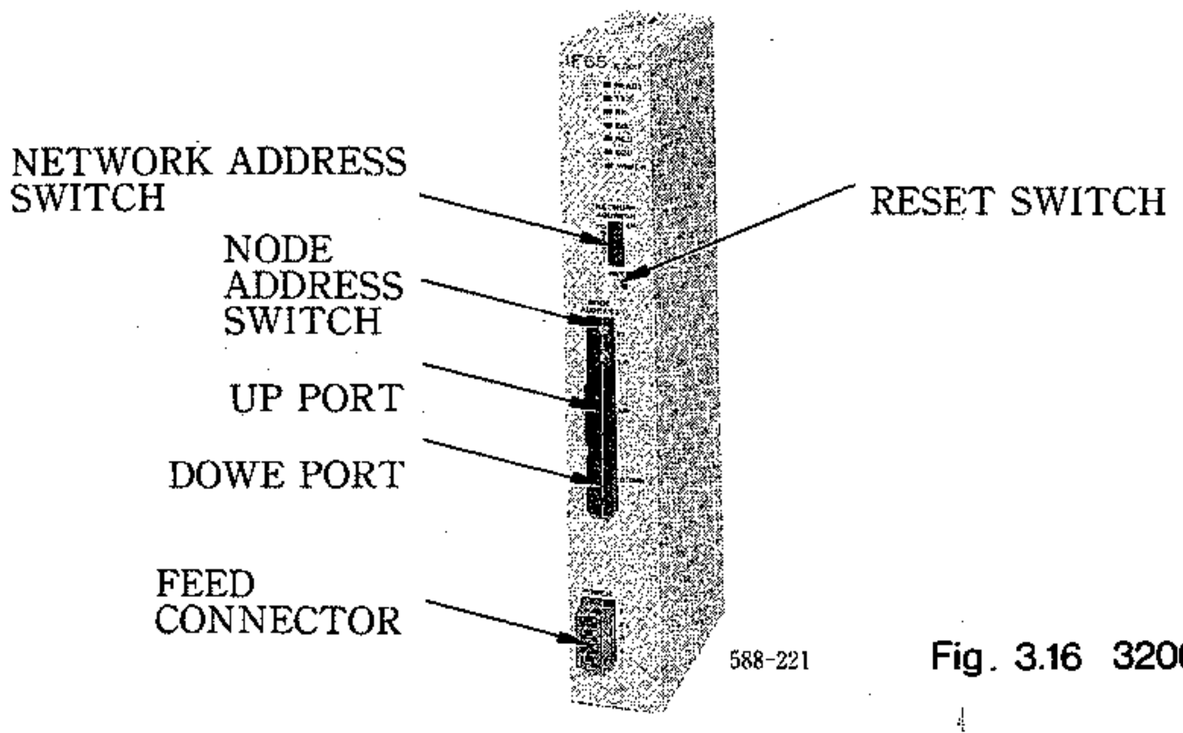
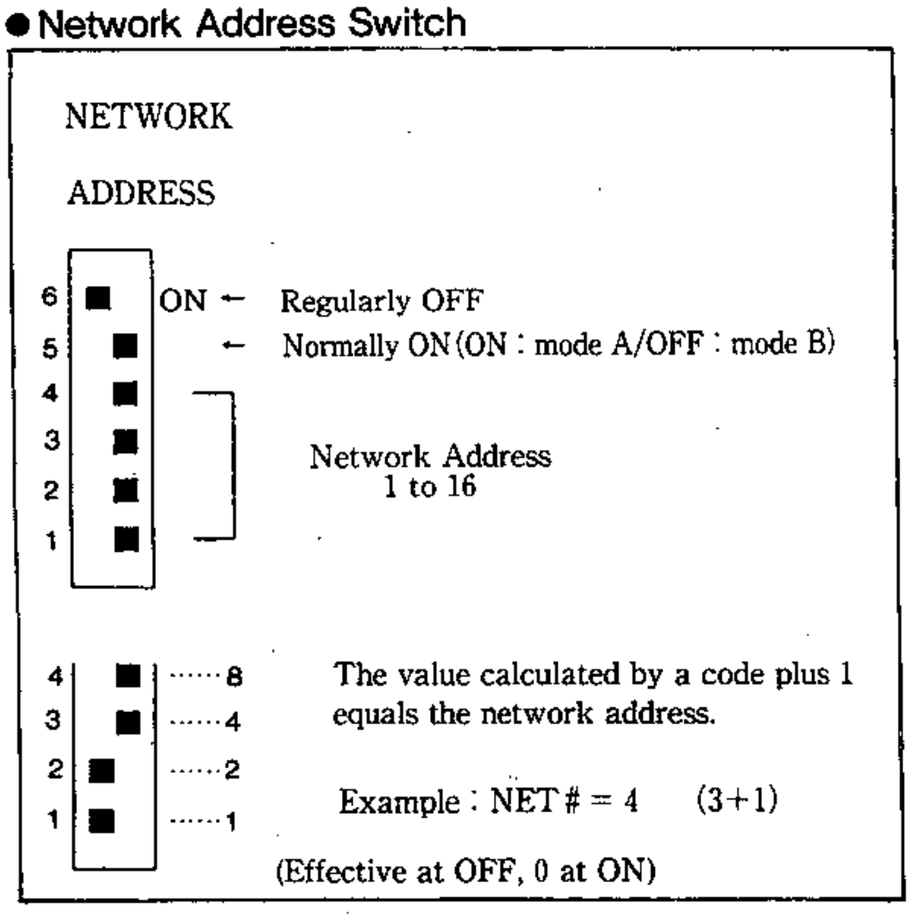


Fig. 3.16 3200 IF module

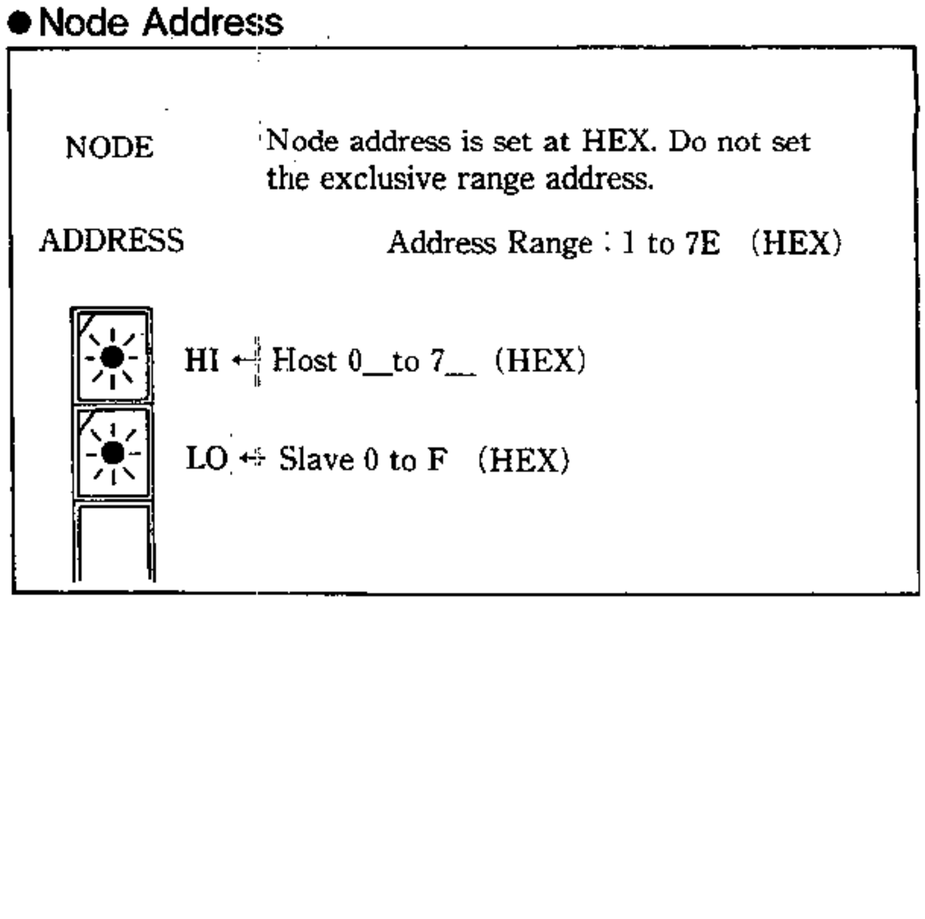
Table 3.34 3200 IF module specifications

Item	Specification
Type	JAMSC-IF65V
Function	Connected to the YENET by an optical fiber for communication with CP series, MEMOCON series and a host computer.
Send/receive buffer	32 k bytes (sending 1/receiving 15 buffers) 1 buffer = 2 k bytes
Indicator lamp (green)	READY : 3200 IF is operating normally. TX : Sending in progress. RX : Receiving in progress. INS : Communication available NCD, BCD : Normal when both are ON. Upstream loop back is in progress when only BCD is OFF. Downstream loop back is in progress when only NCD is OFF. Loop error occurs when both are OFF. POWER : Feeding from the feed connector.
Power consumption	5VDC, 1.0A
Mounting base	MB60 mounting base (CPU base)
Overall size	37.5 mm (w) × 250 mm (h) × 94 mm (d)
Approximate mass	0.6 kg

● Network Address Switch



● Node Address



3.4.10 RIOR module (for electrical transmission)

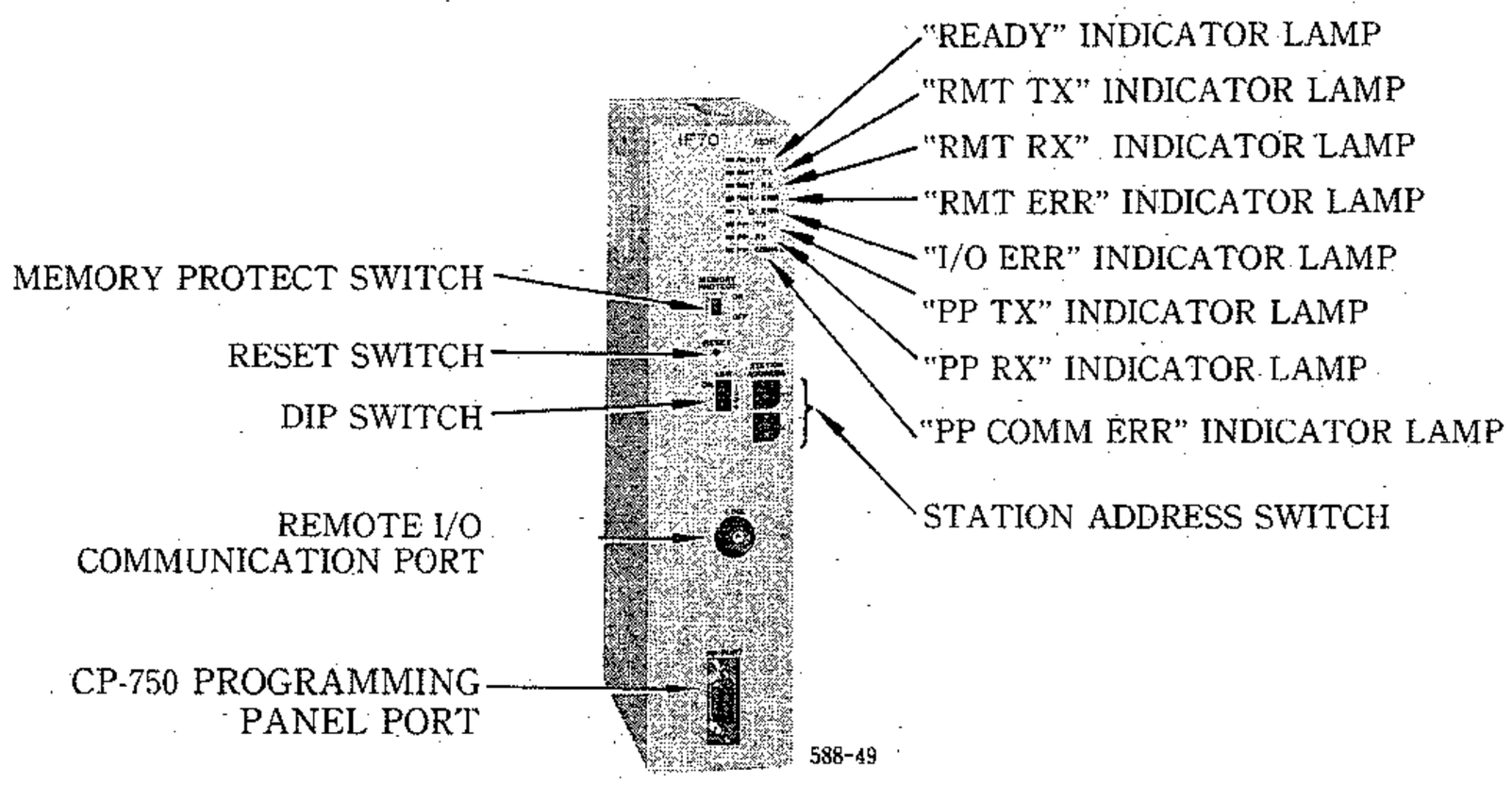


Fig. 3.17 RIOR module

Table 3.35 RIOR module specifications

Item		Specification
Type		JAMSC-IF70V
Function		Required when installing an I/O module at a remote site. Functions as a slave station on a remote communication line. Drives an I/O module via a local I/O bus. Has one port for programming panel. Can monitor the CPU by connecting a programming panel.
Remote line	Topology (net status)	Bus
	Transmission medium	Coaxial cable
	Transmission method	Baseband (Manchester code)
	Transmission speed	0.5/1/2/4 Mbps
	Max. cable length	1 km (when 11C-FB is used)
	Max. number of stations	31
	Troubleshooting	Automatic breakaway of an error station, automatic reactivation after error recovery
Communication port	Number of ports	1
	Transmission specification	EIA RS-232C
	Transmission speed	9600 bps
	Device address	1
	Data bit	8 bits
	Parity check	Even parity
	Connector	D-SUB (9 pins)
	Connected equipment	Programming panel CP-750

Table 3.35 RIOR module specifications (Cont'd)

Item		Specification
Indicator lamps	READY	ON when RIOR normal.
	RMT TX	ON when remote sending in progress.
	RMT RX	ON when remote receiving in progress.
	RMT ERR	ON when a remote communication error occurs.
	I/O ERR	ON when an I/O error occurs.
	PP TX	ON when sending to programming panel in progress.
	PP RX	ON when receiving from programming panel in progress.
	PP COMM ERR	ON when a programming panel communication error occurs.
Mounting base		MB70 mounting base
Overall size		60 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass		0.6 kg

Table 3.36 Rotary switch STATION ADDRESS

Setting range		Station address range
×10	×1	
0	0	Cannot be set
	1 to 9	1 to 9
1, 2	0 to 9	10 to 29
3	0, 1	30, 31
	2 to 9	Cannot be set
4 to 9	——	Cannot be set

Table 3.37 Switches

Switch	Standard setting	Function	
		Setup	Description
MEMORY PROTECT	ON	ON	Not used. (Set to ON.)
		OFF	
RESET	——	——	Depressing this switch initializes this module.

Table 3.38 Dip switch 1SW

No.	Standard setting	Function					
1	OFF	ON	Not used. (Set to OFF.)				
		OFF					
2	OFF	ON	Not used. (Set to OFF.)				
		OFF					
3	Depending on the system	——		Setting			
			Switch	3	OFF	ON	OFF	ON
4				4	OFF	OFF	ON	ON
			Transmission speed (Mbps)		0.5	1	2	4

3.4.11 RIOR module (for optical transmission)

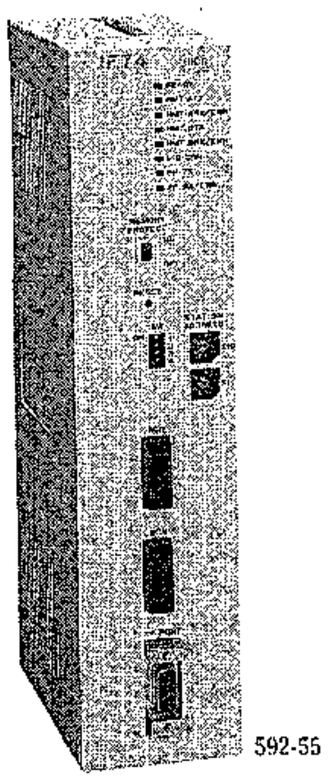


Fig. 3.18 RIOR module

Table 3.39 RIOR module specifications

Item		Specification
Type		JAMSC-IF74V
Function		Functions as a slave station on a remote communication line. Has one port for programming panel. Can monitor the CPU by connecting a programming panel.
Remote communication line	Topology	Cascade, star*, duplex star* (* CPL module is needed.)
	Transmission media	Optical fiber cable (model H-PCF made by Sumitomo Electric Industries Ltd.)
	Connector for optical transmission	Model DL-92 or DL-92H made by Sumitomo Electric Industries Ltd. (Complies with JIS C 5977 F08 type.)
	Transmission method	Baseband (Manchester code)
	Transmission speed	4 Mbps (fixed)
	Transmission distance	12 km max. (650 m for between stations)
	Number of stations	31 max.
	Troubleshooting	Automatic disconnect of a station in trouble, automatic reactivation after error recovery ; double transmission paths
Programming panel port	Transmission specification	Complies with EIA RS-232C.
	Synchronization	Half-duplex communication synhronization
	Transmission speed	9600 pbs
	Character length	8 bits
	Stop bit	1 bit
	Parity check	Even parity
	Protocol	MEMOBUS
	Device address	1
	Connector	D-SUB (9 pins)
	Connectable device	Programming panel CP-750

Table 3.39 RIOR module specifications (Cont'd)

Item		Specification
Indicating lamps	READY	Lights while RIOR module (for optical transmission) is normal.
	RMT ATX	Lights at bit stream of data sent from ACN port.
	RMT ARX/ERR	Green light indicates bit stream of data received in ACN port. Red light indicates receiving data error.
	RMT BTX	Lights at bit stream of data sent from BCN port.
	RMT BRX/ERR	Green light indicates bit stream of data received in BCN port. Red light indicates receiving data error.
	I/O ERROR	Lights at I/O bus error.
	PP TX	Lights at bit stream of data sent from a programming panel port.
	PP RX/ERR	Green light indicates bit stream data received in programming panel port. Red light indicates receiving data error.
Current consumption		5 VDC, 1.5A
Mounting base		MB70 mounting base
Overall size		60 mm (w) ×250 mm (h) × 94 mm (d)
Approximate mass		0.6 kg

Table 3.40 STATION ADDRESS switch

Setting range		Station address range
×10	×1	
0	0	Cannot be set
	1 to 9	1 to 9
1 or 2	0 to 9	10 to 29
3	0 or 1	30 or 31
	2 to 9	Cannot be set
4 to 9	—	Cannot be set

Table 3.41 Switches

Switch		Standard Setting	Function
MEMORY PROTECT		OFF	Not used. (Set to OFF.)
RESET		—	Reset this module by depressing this switch. Depress once after changing station address or 1SW settings.
1SW	1	OFF	ON : Hold mode for output OFF: Clear mode for output (Be sure to set to OFF.)
	2	OFF	ON : Self-diagnosis mode (Normally, do not set to this mode.) OFF: Operation mode
	3	*	ON : Duplex star mode OFF: Cascade mode
	4	OFF	Not used. (Set to OFF.)

* : Set depending on the system.

3.4.12 ASCII module

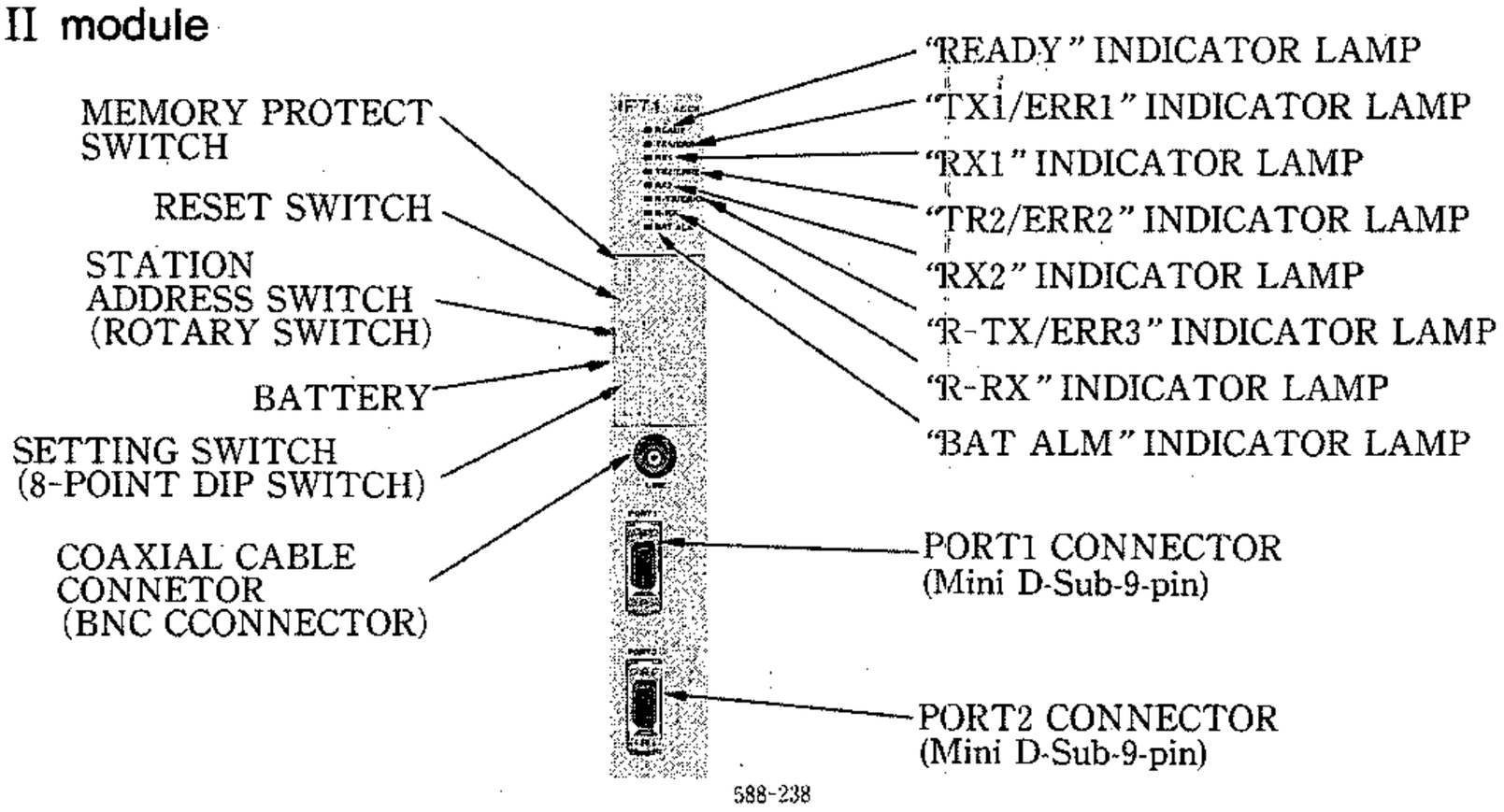


Fig. 3.19 ASCII module

Table 3.42 ASCII module specification

Item		Specification
Type		JAMSC-IF71V
Applicable RIOD		JAMSC-IF62AV (IF62 not applicable)
Number of connecting modules		Max. 8 modules
Memory capacity		64 k bytes / module
Memory backup		1 lithium battery Battery guaranteed for 5 years Total non-energizing memory content are held for 1 year. (25℃)
Message		• Max. 1024 messages / modules • Max. 1024 bytes / messages
Applicable Register		Max. 999 registers / messages
ASCII port	Number of ports	2 ports / module
	Transmission specification	EIA RS-232C half duplex, start-stop transmission.
	Transmission speed	19200/9600/4800/2400/1200/600/300/150 bps
	Data bit	5 to 8 bits
	Parity check	odd/even/none
	Stop bit	1 or 2 bits
	Connector	Mini D-SUB (9 pins) connector
Indicator lamp	READY	ASCII module self diagnosis normal (green)
	TX1/ER1	Port 1 sending in progress (green)
	RX1	Port 1 reciving in progress (green)
	TX2/ER2	Port 2 sending in progress (green)
	RX2	Port 2 reciving in progress (green)
	RTX/ER3	Remote line sending in progress (green)/send or receive error (red)
	RRX	Remote line receiving in progress
	BAT ALM	Battery voltage drop (red)
Self diagnostic function		• ROM check • RAM check • Memory total check • Watchdog timer • Battery voltage check
Mounting base		MB60, MB21, MB70, MB71 mounting base
Overall size		37.5 mm (w) × 250 mm (h) × 94 mm (d)
Approximate mass		1 kg
Current consumption		5 VDC, 1.1A

Table 3.43 ASCII module transmission mode setting

Switch No.	Description			
1, 2	1	2	Mode	Port 1 mode setting
	ON	ON	MEMOBUS RTU mode	
	ON	OFF	MEMOBUS ASCII mode	
	OFF	ON	ASCII equipment mode (1)*1	
	OFF	OFF	ASCII equipment mode (2)*2	
3, 4	3	4	Mode	Port 2 mode setting
	ON	ON	MEMOBUS RTU mode	
	ON	OFF	MEMOBUS ASCII mode	
	OFF	ON	ASCII equipment mode (1)*1	
	OFF	OFF	ASCII equipment mode (2)*2	
5	Not used			
6*3	ON	Self diagnostic mode		
	OFF	Normal operation mode		
7, 8	7	8	Transmission rate	Specify the transmission rate of remote transmission line.
	ON	ON	4 Mbps	
	OFF	ON	2 Mbps	
	ON	OFF	1 Mbps	
	OFF	OFF	0.5 Mbps	

*1 : The data received before READ command is started is unavailable in ASCII equipment mode (1).
*2 : The data received before READ command is started is available in ASCII equipment mode (2).
*3 : Normally, switch 6 should be OFF.

3.4.13 I/O BUFF module

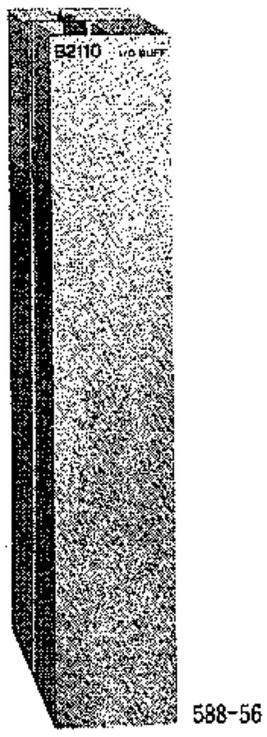


Fig. 3.20 I/O BUFF module

Table 3.44 I/O BUFF module specifications

Item	Specification
Type	JAMSC-B2110V
Function	I/O bus buffer. This module must be mounted on each rack when racks 2 to 5 are used.
Connector	Two connectors for inter-rack connecting cable (W20−1, −2)
Current consumption	5 VDC, 0.4A
Mounting base	MB22 mounting base
Overall size	46.3 mm (w) ×250 mm (h) ×94 mm (d)
Approximate mass	0.4 kg

3.4.14 213 RIO module (for electrical transmission)

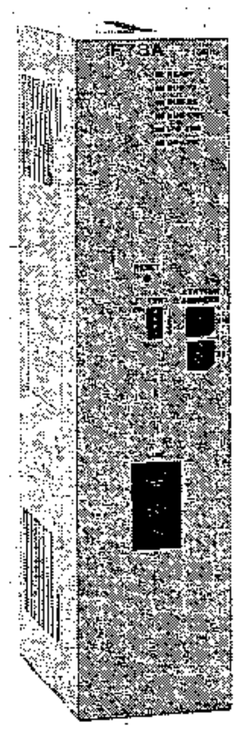

693-236

Fig. 3.21 213 RIO module

Table 3.45 213 RIO module specifications

Item	Specification	
Type	JAMSC-IF73AV	
Function	Connected to CP-213 optical transmission line as the interface. Controls I/O module via local I/O bus. This module provides a momentary power loss protection function.	
Transmission speed	1 Mbps	
Transmission calbe	Electrical bus : KPEV-S 1.25 ㎟	
Transmission distance	Total of distance : 300 m	
Connection method	Bus	
Insulation method	Optical connection	
Number of slots	For setting by programming panel	20 slots/station 1st rack: 8 slots max. 2nd rack: 9 slots max. 3rd rack: 3 slots max.
	For using system standard function ISET-213	35 slots/station 1st rack: 8 slots max. 2nd rack: 9 slots max. 3rd rack: 9 slots max. 4th rack: 9 slots max.
I/O points	Discrete input points : Up to 8 points/slot Discrete output points : Up to 8 points/slot Register input points : Up to 8 points/slot Register output points : Up to 8 points/slot	
Input points	Max. 127 points/station (1 ≦ input points ≦ 127)	
Output points	Max. 127 points/station (0 ≦ output points ≦ 127)	
Current Consumption	5VDC, 1.1A	
Mounting base	MB70 mounting base	
Overall seze	60 mm (w) ×250 mm (h) ×94 mm (d)	
Approximate mass	0.55 kg	

Table 3.46 Indicator lamps

Name	Status	Description
READY	Lit	213RIO module is ready for operation.
	OFF	213RIO module operation fault or hardware fault. The lighting of BUS ERR, I/O ERR, and ON-LINE are invalid.
BUS TX	Lit	When ON-LINE is lit : 2000 I/O input data are being sent to CP-213 master. When ON-LINE is OFF : 2000 I/O input impossible status is being sent to CP-213 master.
	OFF	No data are being sent to CP-213.
BUS RX	Lit	2000 I/O output data or initializing data are being received from CP-213.
	OFF	No 2000 I/O output data or initializing data are being received from CP-213.
BUS ERR	Lit	CP-213 transmission error. No output data are being received for one second.
	Blinking	Error during self-diagnosis.
	OFF	CP-213 transmission is normal.
I/O ERR	Lit	I/O operation error due to 2000 I/O bus error. I/O module fault, or I/O allocattion error.
	OFF	I/O operation is correct under normal 2000 I/O bus.
ON-LINE	Lit	Lights when correct initialization data receiving is completed after the power supply turns ON. Then, this lamp blinks.
	OFF	Correct initialization data receiving is not completed.

Table 3.47 RESET switch

Name	Setting	Description
RESET	ON	213 RIO hardware reset
	OFF	Operation status

Table 3.48 Dip switch 1SW*

No.	Setting	Description
1	ON	Holds output data during CP-213 transmission error.
	OFF	Clears output data during CP-213 transmission error.
2	ON	Holds the lighting of BUS ERR or I/O ERR during CP-213 transmission error or I/O error. 〔213 RIO module is master during self-diagnosis mode (1SW-3 ON).〕
	OFF	Goes out when CP-213 transmission error or I/O error is reset. 〔213 RIO module is slave during self-diagnosis mode (1SW-3 ON).〕
3	ON	Self-diagnosis is executed repeatedly. Do not set normally.
	OFF	I/O operation is executed.
4	ON	Momentary power loss mode. After momentary power loss occured (approx. 1 second or less), the preceding value is output according to I/O allocation. 〔Transmission test is executed during self-diagnosis mode (1SW-3 ON).〕
	OFF	Normal mode 〔No transmission test is executed during self-diagnosis mode (1SW-3 ON).〕

Table 3.49 STATION ADDRESS* rotary switch

Name	Setting	Description	
×10	0 to 3	Sets a figure of tens in station address.	(1 ≦ X10 ×10 + X1 ≦ 31)
×1	0 to 9	Sets a figure of hundreds in station address.	(1 ≦ X10 ×10 + X1 ≦ 31)

* When 1SW or STATION ADDRESS switch setting is changed, power supply should be turn ON again or switch RESET should be set to ON.

3.4.15 213 RIO module (for optical transmission)

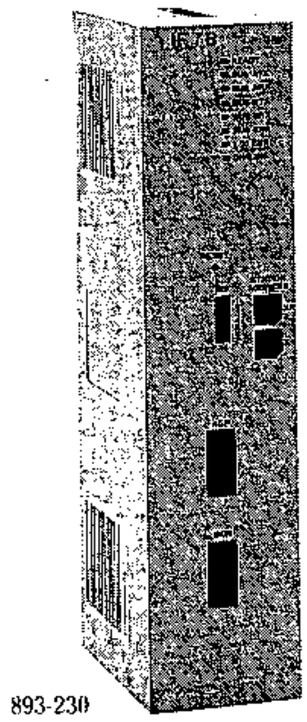


Fig. 3.22 213 RIO module

Table 3.50 213 RIO module specifications

Item		Specification	
Type		JAMSC-IF78V	
Function		Connected to CP-213 optical transmission line as the interface. Controls I/O module via local I/O bus. This module provides a momentary power loss protection function.	
Topology		Cascade, star*, duplex star* (* CP-213 optical star coupler is needed.)	
Transmission media		Optical fiber cable (Model H-PCF made by Sumitomo Electric Industries Ltd.)	
Connector for optical transmission		Model CF-2001H or CF-2011 made by Sumitomo Electric Industries Ltd.	
Transmission distance		Between stations : 1000 m when using connector model CF-2001H 700 m whne using connector model CF-2011 Total : 12 km	
Transmission speed		1 Mbps	
Number of connectable modules		Station addresses 1 to 31 can be set as slave stations.	
	Cascade	18 modules max. including master station (8 modules max./branch at one port) 32 modules max. when one CP-213 optical star coupler is used.	
	Star or Duplex star	32 modules max. including master station (CP-213 optical star coupler is needed.)	
I/O slot		20 slots per station when they are set in programming panel.	1st rack : 8 slots max. 2nd rack : 9 slots max. 3rd rack : 3 slots max.
		35 slots per station when system function ISET-213 is used.	1st rack : 8 slots max. 2nd to 4th racks : 9 slots max. each rack

Table 3.50 213 RIO module specifications (Cont'd)

Item	Specification
I/O points	Discrete input points : Up to 8 points / slot Discrete output points : Up to 8 points / slot Register input points : Up to 8 points / slot Register output points : Up to 8 points / slot
Number of input points	Up to 127 points per station (1 ≦ input points ≦ 127)
Number of output points	Up to 127 points per station (0 ≦ output points ≦ 127)
Current consumption	5 VDC ± 3%, 1.9A
Mounting base	MB70 mounting base
Overall size	60 mm (w) × 250 mm (h) × 94 mm (d)
Approximate mass	0.6 kg

Table 3.51 Indicator lamps

Name	Normal status	Description
READY	Lit	This module is normal.
BUS ATX	Lit or blinking	Lights or blinks at ON-LINE Input data (such as run status, I/O modules) are being sent to CP-213 line of ACN port. OFF at ON-LINE : Ineffective input data for run status are being sent to CP-213 line of ACN port.
BUS ARX	Lit	Signal is being received from CP-213 line of ACN port.
BUS BTX	Lit or blinking	Lights or blinks at ON-LINE : Input data (such as run status, I/O modules) are being sent to CP-213 line of BCN port. OFF at ON-LINE : Ineffective input data for run status are being sent to CP-213 line of BCN port.
BUS BRX	Lit	Signal is being received from CP-213 line of BCN port.
BUS ERR	OFF	Lights at CP-213 transmission error. Blinks at no response from a receiving station or a module fault.
I/O ERR	OFF	Lights at I/O operation (I/O bus) error, I/O module fault, or I/O allocation error.
ON-LINE	Lit	Lights when the correct initial data (I/O allocations) have been received.

Table 3.52 STATION ADDRESS rotary switch

Setting range		Station address range
×10	×1	
0	0	Cannot be set
	1 to 9	1 to 9
1 or 2	0 to 9	10 to 29
3	0 or 1	30 or 31
	2 to 9	Cannot be set
4 to 9	—	Cannot be set

Note: Station address range is set depending on the system.

Table 3.53 Switches (module front)

Switch		Function
RESET		Initialize this module by depressing this switch. Normally, do not depress this switch.
1SW	1	ON : Holds output data during CP-213 transmission error. OFF: Clears output data during CP-213 transmission error.
	2	ON : "BUS ERR" or "I/O ERR" LED is lit and held during CP-213 transmission error or I/O error. 〔If 1SW-3 is set ON (self-diagnosis mode), CP-213 becomes the master.〕 OFF: "BUS ERR" or "I/O ERR" LED is lit during CP-213 transmission error or I/O error, then the ERR LED goes OFF after clearing the error. 〔If 1SW-3 is set ON (self-diagnosis mode), CP-213 becomes the slave.〕
	3	ON : Self-diagnosis mode (Normally, do not set to this mode.) OFF: Operation mode
	4	ON : Momentary power loss mode ; When power is restored after momentary power loss occurs for 1 sec or less, the previous value is output according to the I/O allocation. 〔If 1SW-3 is set ON (self-diagnosis mode), transmission test is executed.〕 OFF: Normal mode 〔If 1SW-3 is set ON (self-diagnosis mode), no transmission test is executed.〕
	5	Not used. (Set to OFF.)
	6	ON : Duplex star mode OFF: Cascade mode

3.4.16 213 SIF module

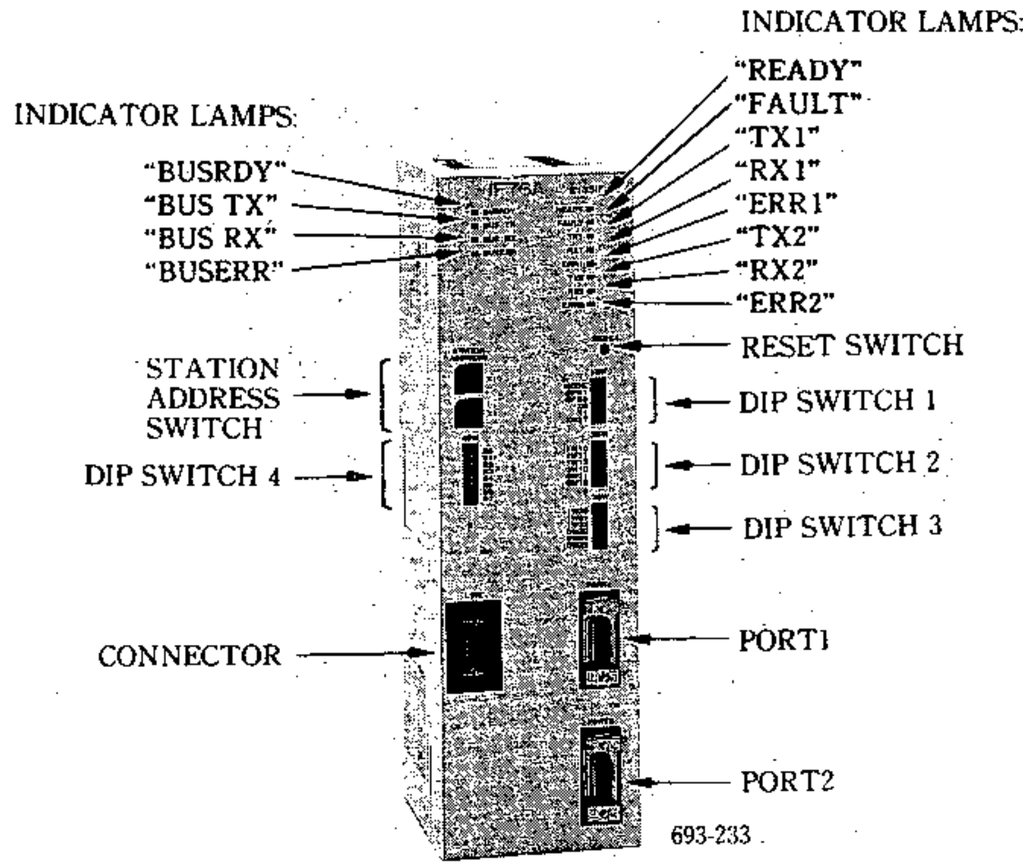


Fig. 3.23 213 SIF module

Table 3.54 213 SIF module specifications

Item		Specification	
Type		JAMSC-IF76V, JAMSC-IF76AV	
Function		This is a protocol converter to connect between CP-3300 and high-pressure GTO inverter VS-683GT3 or CP-3300 and intelligent remote operator GP-410-Y, via CP-213 line. One 213 SIF module (Type JAMSC-IF76V) can connect two remote operators GP-410-Y. One 213 SIF module (Type JAMSC-IF76AV) can connect one inverter VS-683GT3 or two remote operators GP-410-Y.	
Transmission Specification	LINE (CP-213)	Connector	MR-8MG, MR-8L (case)
		Transmission standard	CP-213 FA bus
		Transmission speed	1 Mbps
		Transmission protocol	HDLC (High-level Data Link Control)
		Insulation	Optical connection
		Transmission path	KPEV-S1.25 mm^2 (Complying with 213 IF modules)
		Transmission distance	300 m max.

Table 3.54 213 SIF module specifications (Cont'd)

Item		Specification	
Transmission Specification	PORT1 (Current loop) in VS mode	Connector	D-SUB (9 pins)
		Transmission specification	200 mA Current loop
		Transmission speed	2400/4800/9600 bps
		Transmission protocol	SAMI protocol
		I/O data amount	8 words
		Insulation	Optical connection (with built-in DC/DC converter)
		Cable length	100 m max. Standard type : IPEV-S (Cu) 2Px 1.25 made by Fujikura Ltd.
	PORT1, 2 (PS-232C) in GP mode	Connector	D-SUB (9 pins)
		Transmission specification	EIA RS-232C
		Transmission speed	19200/9600/4800/2400 bps
		Transmission protocol	MEMOBUS protocol
		I/O data amount	0/4/8/16/24/32/48/63 words
		Insulation	Optical connection (with built-in DC/DC converter)
		Cable length	15 m max. Standard type : JZMSZ-W1015-21 (2.5 m) JZMSZ-W1015-22 (15 m)
Current consumption		5 VDC, 1.4A	
Mounting base		MB60, MB22/22A, MB70, MB71S2 mounting base (2 slots)	
Overall size		75 mm (w) × 250 mm (h) × 94 mm (d)	
Approximate mass		0.7 kg	

: Valid only for module type JAMSC-IF76AV

Table 3.55 Indicator lamps

Name (Color)	Normal Status	Description
BUSRDY (green)	Lit	CP-213 FA bus is operating normally.
BUSTX (green)	Blinking	Data are being sent to FA bus.
BUSRY (green)	Blinking	Data are being received from FA bus.
BUSERR (red)	OFF	Lights when PC board in 213 IF module has a fault or a transmission error in FA bus.
READY (green)	Lit	Ongoing nomal operation
FAULT (red)	OFF	Lights when PC board in 213 IF module has a fault.
TX1 (green)	Blinking	Data are being sent to serial port 1. (OFF when no port is used.)
RX1 (green)	Blinking	Data are being received from serial port 1. (Lights when no port is used.)
ERR1 (red)	OFF	Lights when serial port 1 transmission has error
TX2 (green)	Blinking	Data are being sent to serial port 2. (OFF when no port is used.)
RX2 (green)	Blinking	Data are being received from serial port 2. (Lights when no port is used.)
ERR2 (red)	OFF	Lights when serial port 2 transmission has error.

Table 3.56 Switches (module front)

<table>
<tr><th>Switch</th><th>Name</th><th>Standard setting</th><th>Function</th></tr>
<tr><td>Push button</td><td>RESET</td><td>—</td><td>All setting in 213 SIF module are reset when this push button is depressed. Normally, do not depress this button.</td></tr>
<tr><td rowspan="4">1SW</td><td>MODE</td><td>*</td><td>OFF: Serial port is GP mode. ON : Serial port is VS mode.</td></tr>
<tr><td>BPS 1
BPS 2</td><td>*</td><td>Baud rate setting switch
<table>
<tr><td>Band rate</td><td>19200</td><td>9600</td><td>4800</td><td>2400</td></tr>
<tr><td>BPS 1</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>BPS 2</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
</table>
Note: Baud rate 19200 is not used in VS mode.</td></tr>
<tr><td>×0</td><td>*</td><td>OFF: No parity in GP mode
ON : Even parity in GP mode
OFF: 660V specification in VS mode
ON : 3300V specification in VS mode</td></tr>
<tr><td>×1</td><td>OFF</td><td>OFF: Operation mode
ON : Self-diagnosis mode (Normally, do not set this mode.)</td></tr>
<tr><td rowspan="2">2SW</td><td>ERST</td><td>OFF</td><td>OFF: Fault register remains while ERR LED is lit.
ON : Fault register is cleared after ERR LED goes OFF.</td></tr>
<tr><td>IW10
IW11
IW12</td><td>*</td><td>Set input data amount from host controller to port 1.
<table>
<tr><td>Words / Switch</td><td>0</td><td>4</td><td>8</td><td>16</td><td>24</td><td>32</td><td>48</td><td>63</td></tr>
<tr><td>IW10</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>IW11</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
<tr><td>IW12</td><td>OFF</td><td>OFF</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td><td>ON</td><td>ON</td></tr>
</table>
Note: word "0" means no port.</td></tr>
<tr><td>OW10
OW11
OW12</td><td>*</td><td>Set output data amount from port 1 to host controller.
<table>
<tr><td>Words / Switch</td><td>0</td><td>4</td><td>8</td><td>16</td><td>24</td><td>32</td><td>48</td><td>63</td></tr>
<tr><td>OW10</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>OW11</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
<tr><td>OW12</td><td>OFF</td><td>OFF</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td><td>ON</td><td>ON</td></tr>
</table>
Note: word "0" means no port.</td></tr>
</table>

*: Set depending on the system.
: Valid only for module type JAMSC-IF76AV

Table 3.56 Switches (module front) (Cont'd)

<table>
<tr><th>Switch</th><th>Name</th><th>Standard setting</th><th>Function</th></tr>
<tr><td rowspan="4">3SW</td><td>IW20
IW21
IW22</td><td>*</td><td>Set input data amount from host controller to port 2.
<table>
<tr><th>Words / Switch</th><th>0</th><th>4</th><th>8</th><th>16</th><th>24</th><th>32</th><th>48</th><th>63</th></tr>
<tr><td>IW20</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>IW21</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
<tr><td>IW22</td><td>OFF</td><td>OFF</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td><td>ON</td><td>ON</td></tr>
</table>
Note : "0" meand no port.</td></tr>
<tr><td>OW20
OW21
OW22</td><td>*</td><td>Set output data amount from port 2 to host controller.
<table>
<tr><th>Words / Switch</th><th>0</th><th>4</th><th>8</th><th>16</th><th>24</th><th>32</th><th>48</th><th>63</th></tr>
<tr><td>OW20</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>OW21</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
<tr><td>OW22</td><td>OFF</td><td>OFF</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td><td>ON</td><td>ON</td></tr>
</table>
Note: "0" means no port.</td></tr>
<tr><td>S0</td><td>OFF</td><td>OFF: Operation mode.
ON : Self-diagnosis mode. } This setting should be the same as 1SW-X1 setting.</td></tr>
<tr><td>S1</td><td>OFF</td><td>OFF: No CP-213 transmission port diagnosis is performed.
ON : CP-213 transmission port diagnosis is performed when selecting self-diagnosis mode.</td></tr>
<tr><td rowspan="2">4SW</td><td>S2</td><td>OFF</td><td>OFF: CP-213 slave station is set.
(Be sure to set this switch to OFF except when selecting self-diagnosis mode.)
ON : CP-213 master station is set.</td></tr>
<tr><td>S3 to S7</td><td>OFF</td><td>Not used. Be sure to set these switches to OFF.</td></tr>
<tr><td rowspan="2">STATION ADDRESS</td><td>×10</td><td rowspan="2">*</td><td>1 to 3</td><td rowspan="2">For station address, ×10 sets a figure in tens and ×1 sets a figure in hundreds.
(Setting range : 01 to 31)</td></tr>
<tr><td>×1</td><td>0 to 9</td></tr>
</table>

*: Set depending on the system.

Table 3.57 LINE connector

Terminal No.	Signal Name	Function
1	DATA +	Send and receive data lines
8	DATA -	
2	I/O +	Switches send and receive operations. Use for a three-wire system or when connecting an external repeater.
5	I/O -	
4	SCLK +	Clock signal for three-wire system in CP-213 system.
7	SCLK -	
3 and 6	—	Not used. (Do not connect anything.)

Table 3.58 PORT1 connector (in VS mode)

Terminal No.	Signal Name	Function
1	FG	Frame ground
7	TXD	Data send line
2	TXD-RETURN	
9	RXD	Data receive line
8	RXD-RETURN	
3	(LOW)	Not used. (Do not connect anything.)
4	(HIGH)	
5	(LOW)	
6	(NC)	

: Valid only for module type JAMSC-IF76AV

Table 3.59 PORT1, PORT2 connectors (in GP mode)

Terminal No.	Signal Name	Function
1	FG	Frame ground
2	TXD	Data send line
3	RXD	Data receive line
4	RTS (HIGH)	Control signal (Not used in 213 SIF module.)
5	CTS (LOW)	
6	(NC)	Not used. (Do not connect anything.)
7	SG	Signal ground
8	(NC)	Not used. (Do not connect anything.)
9	(HIGH)	

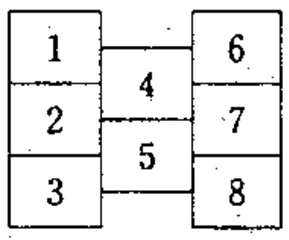


Fig. 3.24 LINE connector (front layout)

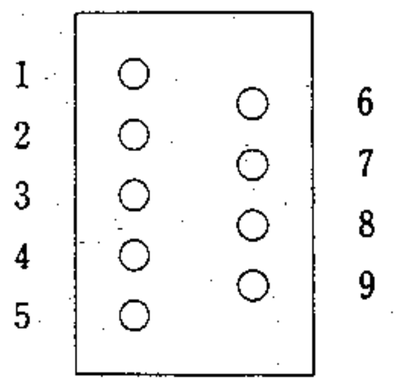


Fig. 3.25 PORT1, PORT2 connector (front layout)

3.4.17 CPL (optical star coupler) module

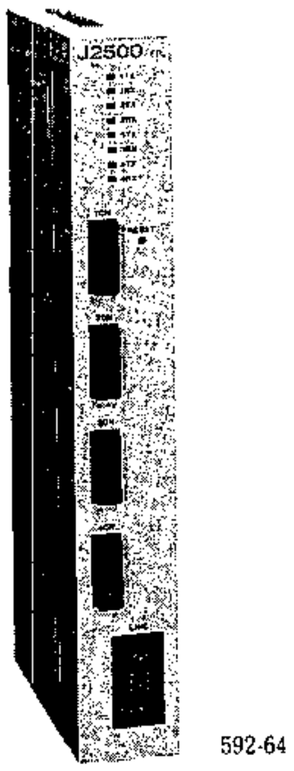


Fig. 3.26 CPL (optical star coupler) module

Table 3.60 CPL module specifications

Item		Specification
Type		DISCT-J2500
Function		Has four optical connectors (1CN to 4CN) and divides the send/receive data among connectors. Closes the connect when the communication line is normal.
Indicating lamps	1TX	Lights at bit stream of data sent from 1CN port.
	1RX	Lights at bit stream of data received in 1CN port.
	2TX	Lights at bit stream of data sent from 2CN port.
	2RX	Lights at bit stream of data received in 2CN port.
	3TX	Lights at bit stream of data sent from 3CN port.
	3RX	Lights at bit stream of data received in 3CN port.
	4TX	Lights at bit stream of data sent from 4CN port.
	4RX	Lights at bit stream of data received in 4CN port.
Switch	RESET	when the contact output is opened once, depress this switch to close.
Contact output (for only DC load)		Contact switching capacity : 48 VDC, 200 mA max. (min. load : 1VDC, 1 mA) When power supply turns ON, the contact is "closed" status. The contact is opened when 1CN to 4CN ports have not received data for one to three seconds after receiving data normally once. For unused ports or ports disconnected from sending station, the contacts are not opened because they cannot detect the receiving data.
Current consumption		5 VDC, 1.5 A
Mounting base		MB60, MB70, MB22/22A mounting base
Overall size		37.5 mm (w) × 250 mm (h) × 94 mm (d)
Approximate mass		0.6 kg

3.5 I/O MODULES

2000 series I/O modules are used. Table 3.61 lists their specifications. Refer to the manuals of 2000 series I/O modules and of the individual module for details.

In addition, the CP-3300 system I/O devices include CP-313M series, CP-813 series, and CP-815. Refer also to the respective manuals for these devices.

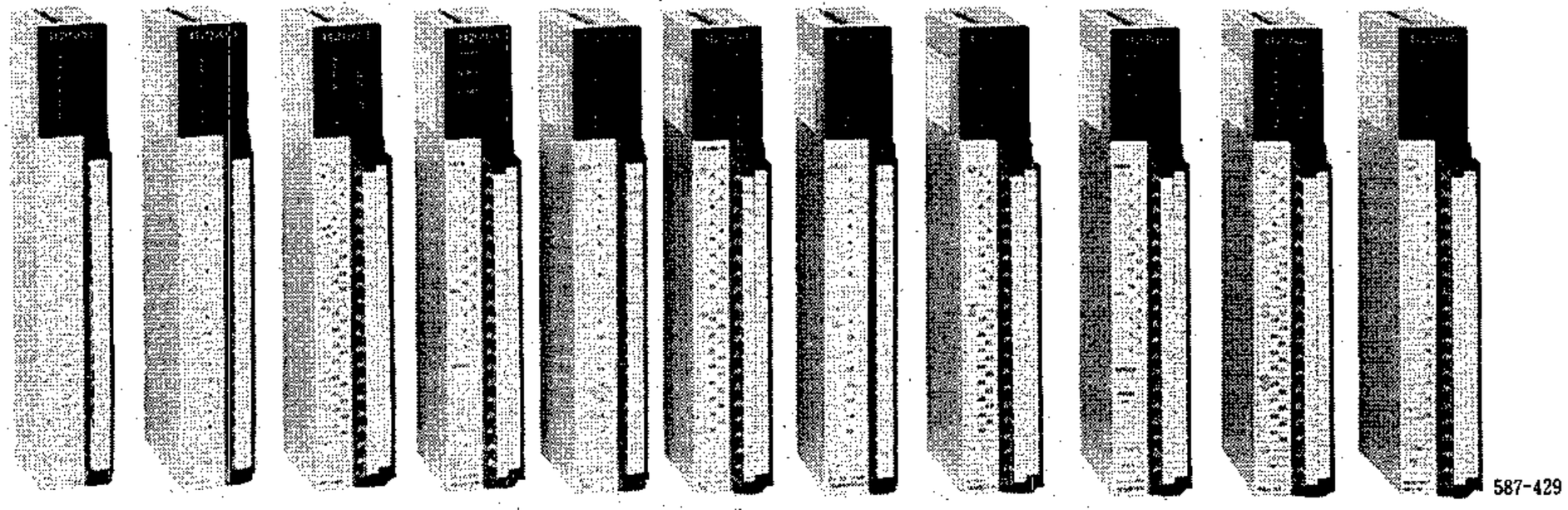


Fig. 3.27 2000 series I/O modules

Table 3.61 2000 series I/O module specifications

Type		Item: Type JAMSC-	Voltage	Current	Input impedance	Note	Response	Number of circuits contained
Input	AC	100V	B2501V	100VAC	10mA	About 10kΩ (at 50Hz)	—	OFF → ON 15ms or less ON → OFF 25ms or less	16
		200V	B2503V	200VAC	10mA	About 20kΩ			16
		100V	B2505V	100VAC	10mA	About 10kΩ			32
		200V	B2507V	200VAC	10mA	About 20kΩ			32
	DC	12/24V	B2601V	12/24VDC	5mA/10mA	About 2.4kΩ		OFF → ON 5ms or less	16
			B2603V	12/24VDC	5mA/10mA	About 2.4kΩ	—	OFF → ON 7ms or less ON → OFF 10ms or less	32
			B2605V	12/24VDC	2.5mA/5mA	About 4.7kΩ		OFF → ON 5ms or less ON → OFF 10ms or less	64
			B2615V						64
		5/12V	B2607V	5/12VDC	4mA/11mA	About 1.2kΩ	—	OFF → ON 0.5ms or less ON → OFF 0.5ms or less	32
		48V	B2611V	48VDC	9.4mA	About 5kΩ		OFF → ON 5ms or less	16
	Register		B2701V	12/24VDC	8mA/24VDC	About 2.4kΩ	—	—	8
			B2711V						8
	Analog		B2703V	0 to 10V	—	—	See manual No. SIE-C815-13.9.	—	8
			B2733V	-10 to +10V	—	—		—	8
			B2743V	1 to 5V	4 to 20mA	—		—	8
	Instrumentation		B2705V	—	—	—	Contact your Yaskawa representative.	—	4

Table 3.61 2000 series I/O module specifications (Cont'd)

Type		Item	Type JAMSC-	Voltage	Current	Input impedance	Note	Response	Number of circuits contained
Output	AC	100/200V	B2500V	100/200VAC	1A/circuit 3A/8 circuits	–	With CR and varistor. Fuse : 7.5A/8 circuits	OFF → ON 10ms or less ON → OFF 15ms or less	16
			B2504V	100/200VAC	0.3A/circuit 1.2A/8 circuits		With CR and fuse		32
	DC	12/24v	B2600V	12/24VDC	2A/circuit 5A/8 circuits	–	–	1ms or less	16
			B2602V	12/24VDC	0.3A/circuit 0.6A/4 circuits		–		32
			B2602AV				With fuse		32
			B2604V	12/24VDC	0.1A/circuit 0.4A/8 circuits		–		64
		5/12V	B2606V	5/12VDC	20mA/circuit 640mA/82 circuit		–		32
	Relay contact		B2902V	24VDC 100VAC 200VAC	24VDC 1A/circuit 220VAC 1A/circuit		Relay coil voltage: 24 VDC ± 10% Min. Voltage and current requirement: 5V, 10mA	OFF → ON 10ms or less ON → OFF 15ms or less	32
			B2912V						32
			B2904V	110VDC 200VAC	110VDC 0.3A/inductive load 220VAC 0.5A/inductive load	–	Relay coil voltage: 24 VDC ± 5% Min. Voltage and current requirement 5V, 1mA	5ms or less	16
			B2914V				Relay coil voltage: 24 VDC ± 5% Min. voltage and current requirement: 24V, 10mA		16
	Register		B2700V	12/24VDC	100mA/circuit	–	–	–	8
			B2710V						8
	Analog		B2702V	0 to 10V	–	–	See manual No. SIE-C815-13.9	–	2
			B2712V	0 to 5V					2
			B2722V	-5 to +5V					2
			B2732V	-10 to +10V					2
			B2742V	4 to 20mA					2
Intelligent	Reversible counter		B2801V	–	–	–	See manual No. SIE-C815-13.11	–	1
	Preset counter		B2802V				SIE-C815-13.12		1
	Positioning		B2803V				SIE-C815-13.13		1
			B2813V				SIE-C815-13.14		
			B2823V				SIE-C815-13.16		
	Interface between modules		B2806V				SIE-C815-14.18		1
	PID		B2800V				SIE-C815-14.24		1
	NFT counter		B2807V				–		1
	R/D converter		B2809V						1
	S/R converter		DISCT-J2009V						1

3.6 MOUNTING BASES

3.6.1 MB60 mounting base

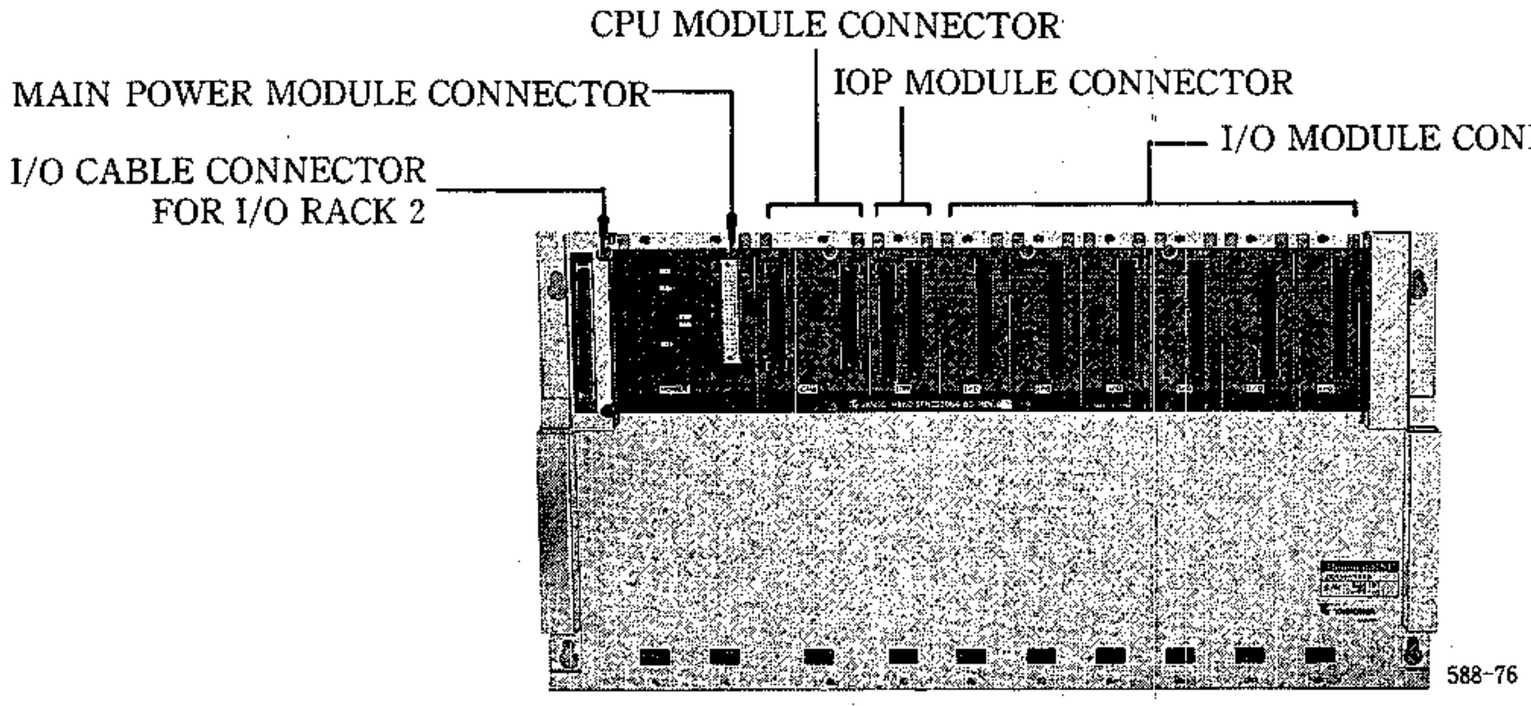


Fig. 3.28 MB60 mounting base

Table 3.62 MB60 mounting base specifications

Item	Specification
Type	JRMSI-MB60
Use	Can mount a main power module, CPU module, communication modules (IOP, COMM, RIOD, 213IF), or up to 6 I/O modules.
Overall size	480 mm (w) ×250 mm (h) ×21 mm (d)
Approximate mass	1.4 kg

3.6.2 MB22/22A mounting base

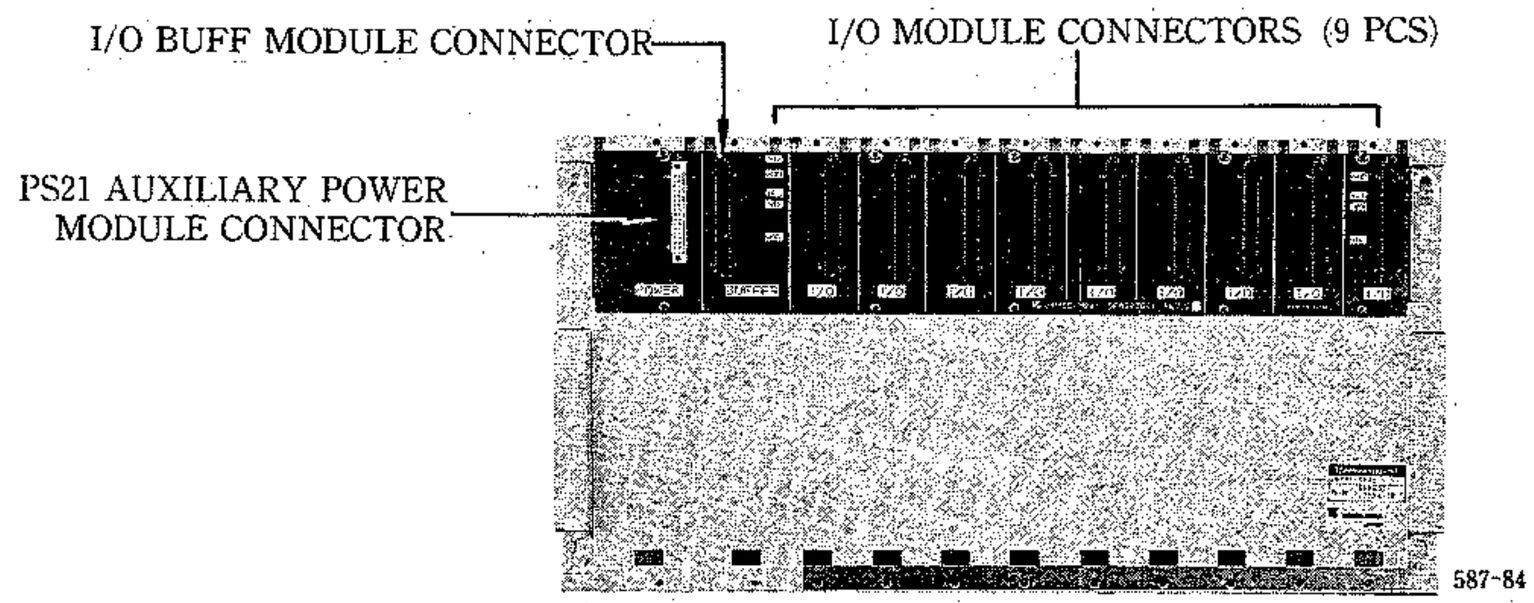


Fig. 3.29 MB22/22A mounting base

Table 3.63 MB22/22A mounting base specifications

Item	Specification
Type	JRMSI-MB22, JRMSI-MB22A
Use	For additional I/O. Can mount an auxiliary power module, I/O BUFF module, and up to 9 I/O modules.
Overall size	480 mm (w) ×250 mm (h) ×21 mm (d)
Approximate mass	1.3 kg

3.6.3 MB70 mounting base

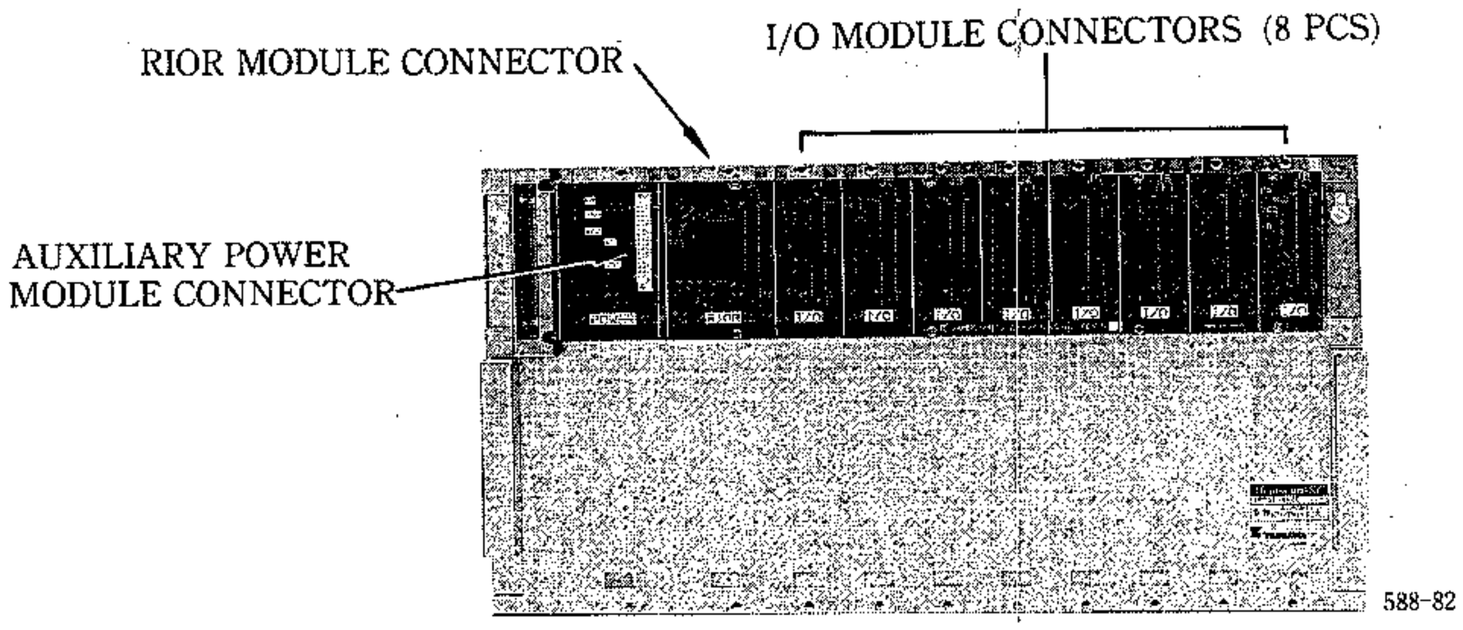


Fig. 3.30 MB70 mounting base

Table 3.64 MB70 mounting base specifications

Item	Specification
Type	JRMS1-MB70
Use	Can mount an auxiliary power module, RIOR module, and up to 8 I/O modules.
Overall size	480 mm (w) ×250 mm (h) ×21 mm (d)
Approximate mass	1.3 kg

3.6.4 MB61 mounting base

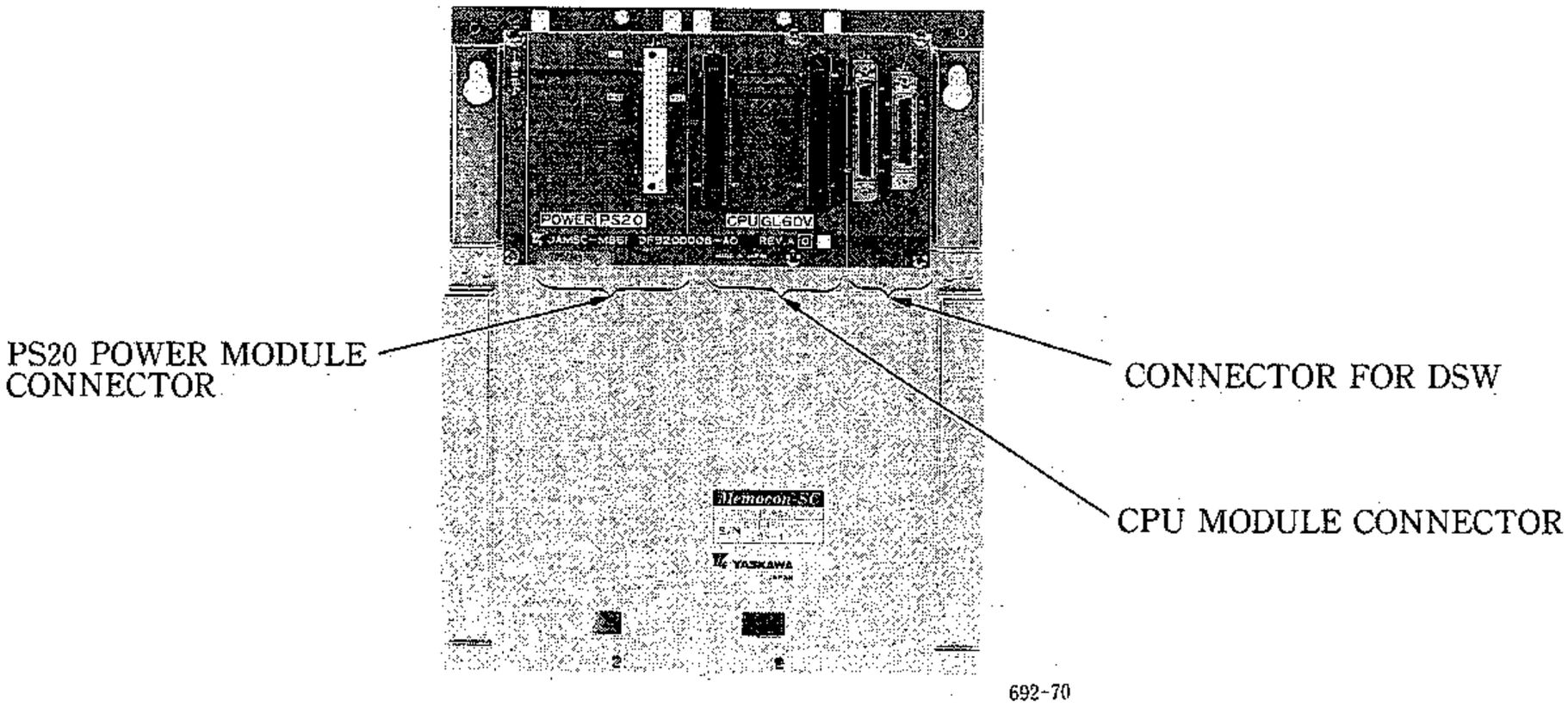


Fig. 3.31 MB61 mounting base

Table 3.65 MB61 mounting base specifications

Item	Specification
Type	JRMSI-MB61
Use	For mounting CPU in a dual CPU system. Can be connected to DSW module by 2 cables.
Overall size	200 mm (w) × 250mm (h) × 21mm (d)
Approximate mass	0.6 kg

3.7 INTER-RACK CONNECTING CABLES

3.7.1 CPU connecting cables

Table 3.66 CPU connecting cable specifications

Type	Length	Connector	Use
JZMSZ-3300-2	90 cm	36	For connecting CPU to DSW module in a dual CPU system.
JZMSZ-3301-2	90 cm	50	

3.7.2 I/O cables

Table 3.67 I/O cable specifications

Item	Specification	
Type	JZMSZ-W20-1	JZMSZ-W20-2
Length	0.5 m	1.5 m
Use	For connecting extended I/O rack.	

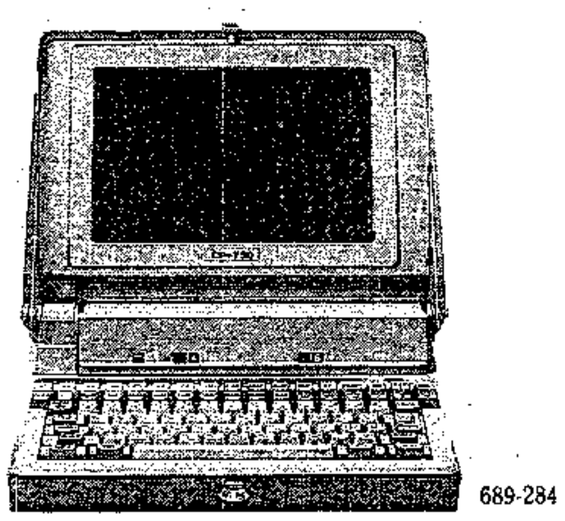
3.8 FAN UNIT

Table 3.68 Fan unit specifications

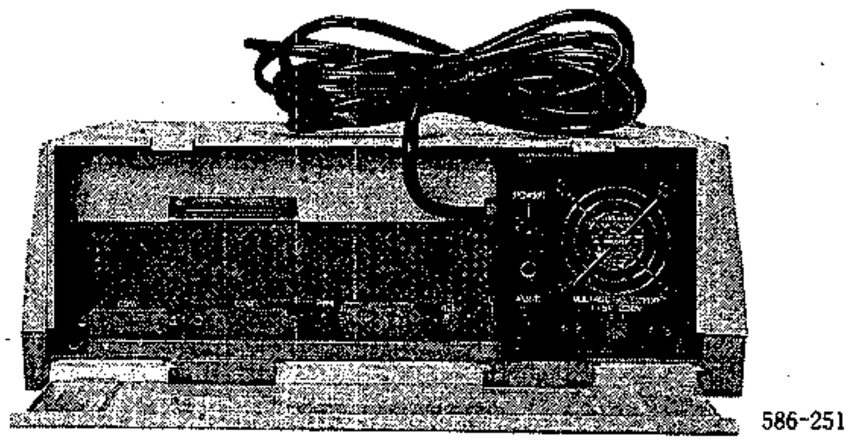
Item	Specification	
Code number	EUX003460	EUX003480
Use	For single CPU system	For dual CPU system
Power supply	100 / 110 VAC; 50 / 60 Hz.	
Others	With alarm contact output.	

3.9 PROGRAMMING PANEL

Refer to the Control Pack CP-3300 Programming Panel Operator's Manual for details of operation.



(A) FRONT VIEW



(B) REAR VIEW

Fig. 3.32 Programming panel

Table 3.69 Programming panel specifications

Item		Specification	Remarks
Code No.		87750-05100-S11XX*	English Version
Function		• Creating, modifying, and editing programs • Storing programs in the operation section, reading and displaying programs • Loading, dumping, and comparing programs • I/O allocation • Monitoring, file management	
Attached devices	Graphics display	• Plasma display 640 dots × 400 dots	
	Keyboard	• Function keys, numeric keys, ASCII keys • CRT label key	
	Floppy disk drive	• 3.5 inch microfloppy disk drive : 2 units	
	Video output	• Monitor type: monochro raster scan CRT	NEC product PC-8841 can be used
	Communication port	• Parallel (Centronics specification standard) port : 1 unit • EIA RS-232C port : 2 units, 9600 bps	Can print out a ladder circuit by connecting a printer.

* XX : 00 to 99

Table 3.69 Programming panel specifications (Cont'd)

Item		Specification	Remarks
CPU module port 1 connection		Use cable W1015-T1 (5 m long) or W1015-T2 (15 m long).	
General specification	Power supply	Single-phase, 85 to 132/195 to 265 VAC (switched)	100/110/120 VAC, 50/60Hz used.
	Power loss	120 VA	
	Operating ambient temperature	+5 to +40 ℃	
	Storage temperature	−20 to +60 ℃	
	Humidity	20 to 80% RH (non-condensing)	
	Environment	Non-explosive or non-flammable, absence of corrosive gases, no excessive dust.	
	Grounding	The chassis grounding line is connected via a cable connecting to the CPU module.	
	Approximate mass	9 kg	

4 DRAWING AND PROGRAM HIERARCHICAL STRUCTURE

This section describes the drawings as basic units of a program and their hierarchical structure.

There are five types of drawings each with a hierarchical structure.
Functions defined independent of drawings can be referred freely by any drawing.

CONTENTS PAGE

4.1 BASIC DRAWING TYPES AND PRIORITY

The CP-3300 user programs are managed in drawings (DWG) identified by drawing numbers (DWG. No.). The drawings are hierarchically arranged with basic, detailed, and expanded drawings.

The basic drawings are classified by the first character (A, I, H, L, B) of the drawing number to match the processing purpose. The maximum number of detailed drawings plus expanded drawings, priority, and execution conditions are defined in Table 4.1.

Table 4.1 Basic drawing types and priority

Drawing	Drawing role	Number of drawings	Priority	Execution conditions	Remarks
DWG.A	Start processing	16	1	Power on	Executed once at power-on
DWG.I	Interrupt processing	16	2	Interrupt generation	Executed once at interrupt generation
DWG.H	High-speed scanning	64	3	Constant period startup	Executed for every high-speed scan time
DWG.L	Low-speed scanning	256	4	Constant period startup	Executed for every low-speed scan time
DWG.B	Batch job processing	16	5	START instruction	Executed by a START instruction in the low-speed scan processing drawing

Notes:

1. A maximum of 300 drawings can be created. Note that this is less than the total number of drawings in Table 4.1.
2. A batch job processing drawing has no basic drawing, but only detailed and expanded drawings. It has no operation error handling drawing (DWG.B00) either.

4.2 DRAWING HIERARCHICAL STRUCTURE

In the CP-3300, user programs are managed in drawings distinguished by drawing numbers (DWG. No.). These drawings form the basis of user programs.

Drawings are classified by the first character (A, I, H, L, B) to match the processing pbjective and made up of basic, detailed, and expanded drawings.

(1) Hierarchical structure of drawings

The user will arrange each processing program hierarchically into basic, detailed, and expanded drawings as shown in Fig. 4.1.
Detailed and expanded drawings are executed by DWG reference (SEE) instructions in the basic and detailed drawings. All drawings can refer functions.

If an operation error occurs, the operation error handling drawing (DWG. X00) is started corresponding to each drawing.

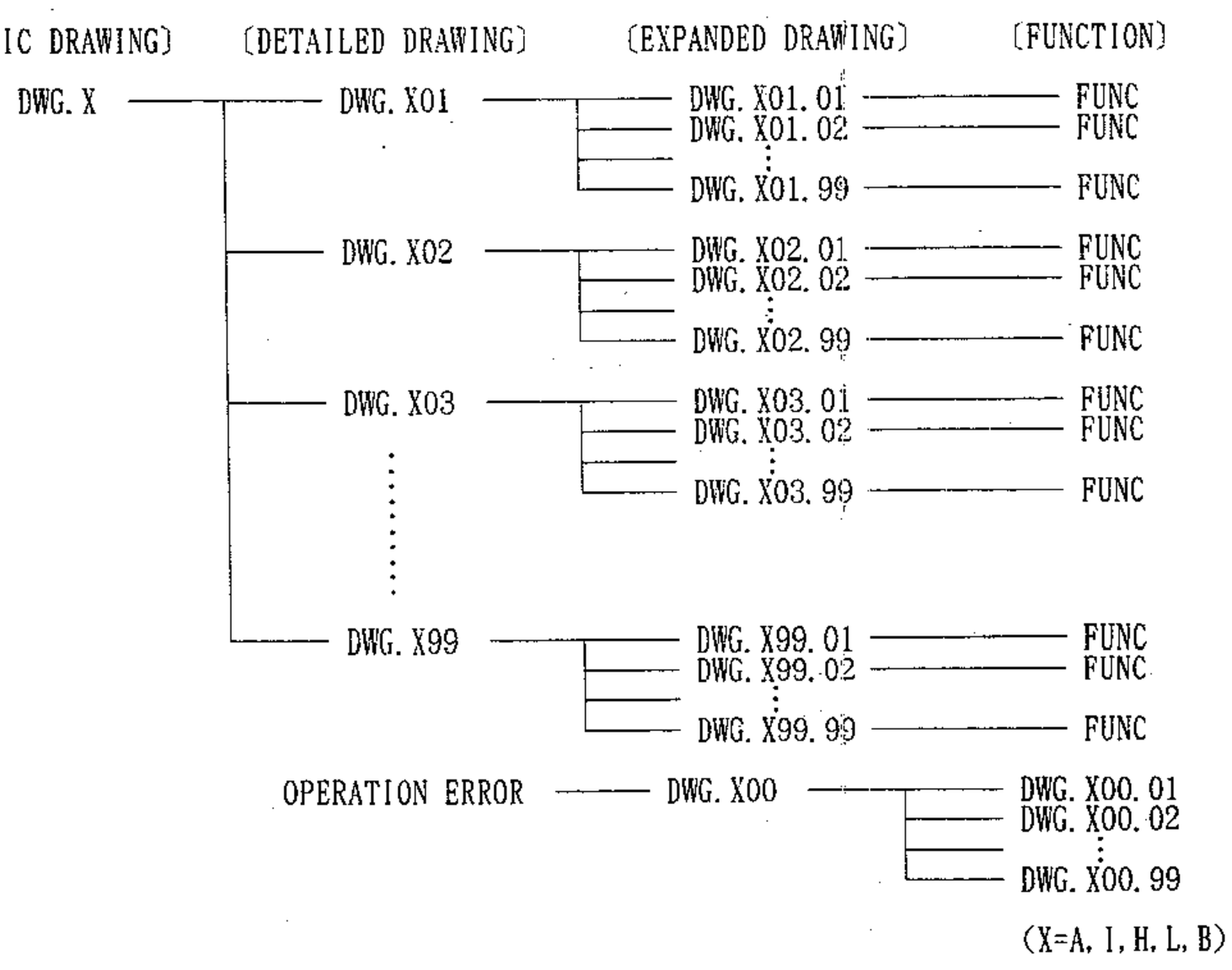


Fig. 4.1 DWG hierarchical structure

(2) Drawing processing method

Hierarchical drawings is processed by letting the upper drawing refer the lower drawing as shown in Fig. 4.2.

DWG XYY. ZZ

X : = drawing type; YY : = detailed drawing number; ZZ : = expanded drawing number

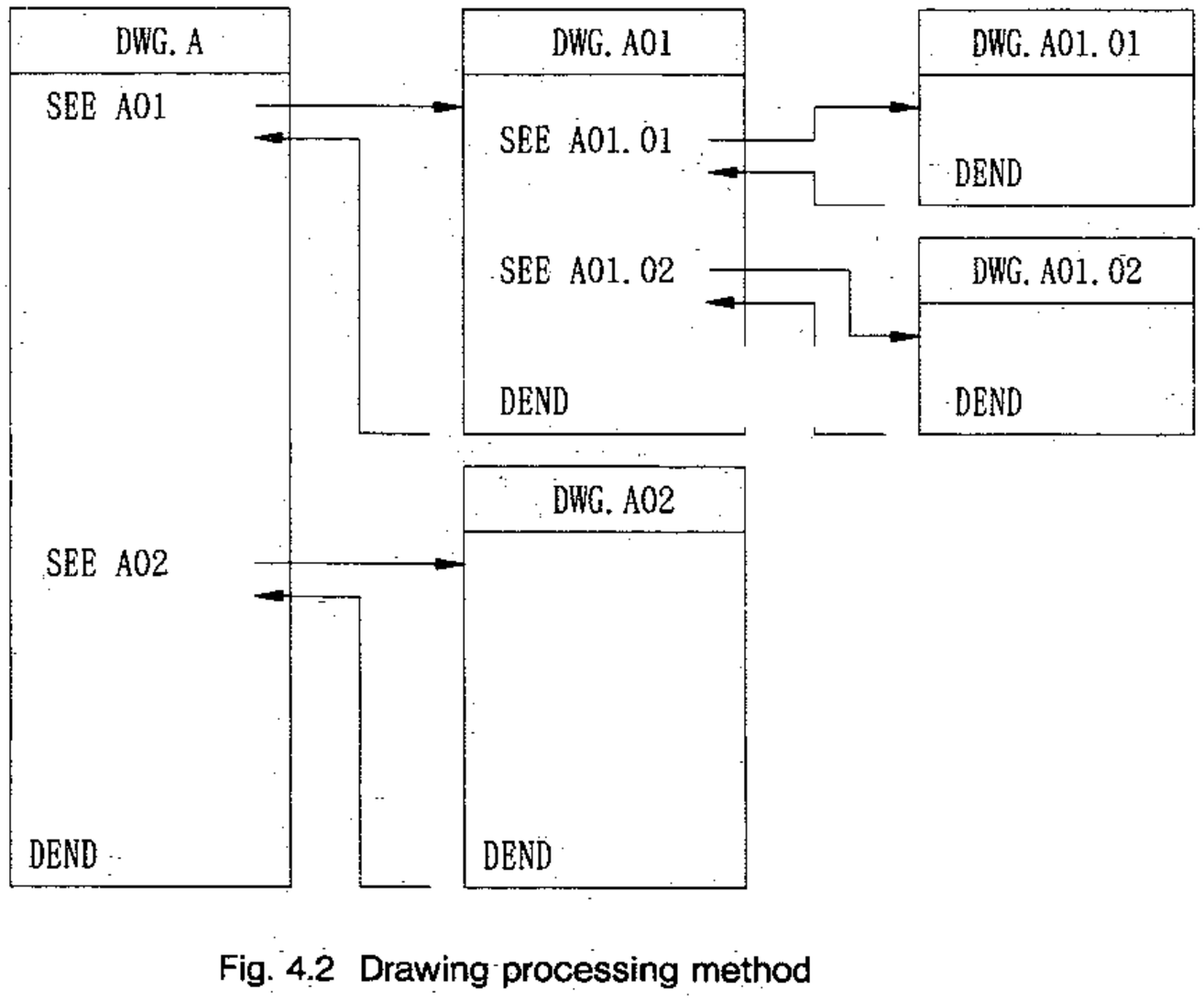


Fig. 4.2 Drawing processing method

4.3 FUNCTIONS

Functions are a new concept replacing the conventional subroutines. Functions are defined quite independently of drawings and can be referenced by any drawing. An existing function can be referenced from another function.

The function are grouped into system standard functions and user defined functions. They are treated as equivalent from the programming point of view.

Using these functions facilitates program modularization and program creation and maintenance.

(1) Function graphic notation

To reference a function from a drawing or a function, a function graphic notation shown in Fig. 4.3 is used. When creating a function, the user will define a graphic notation according to the function I/O condition.

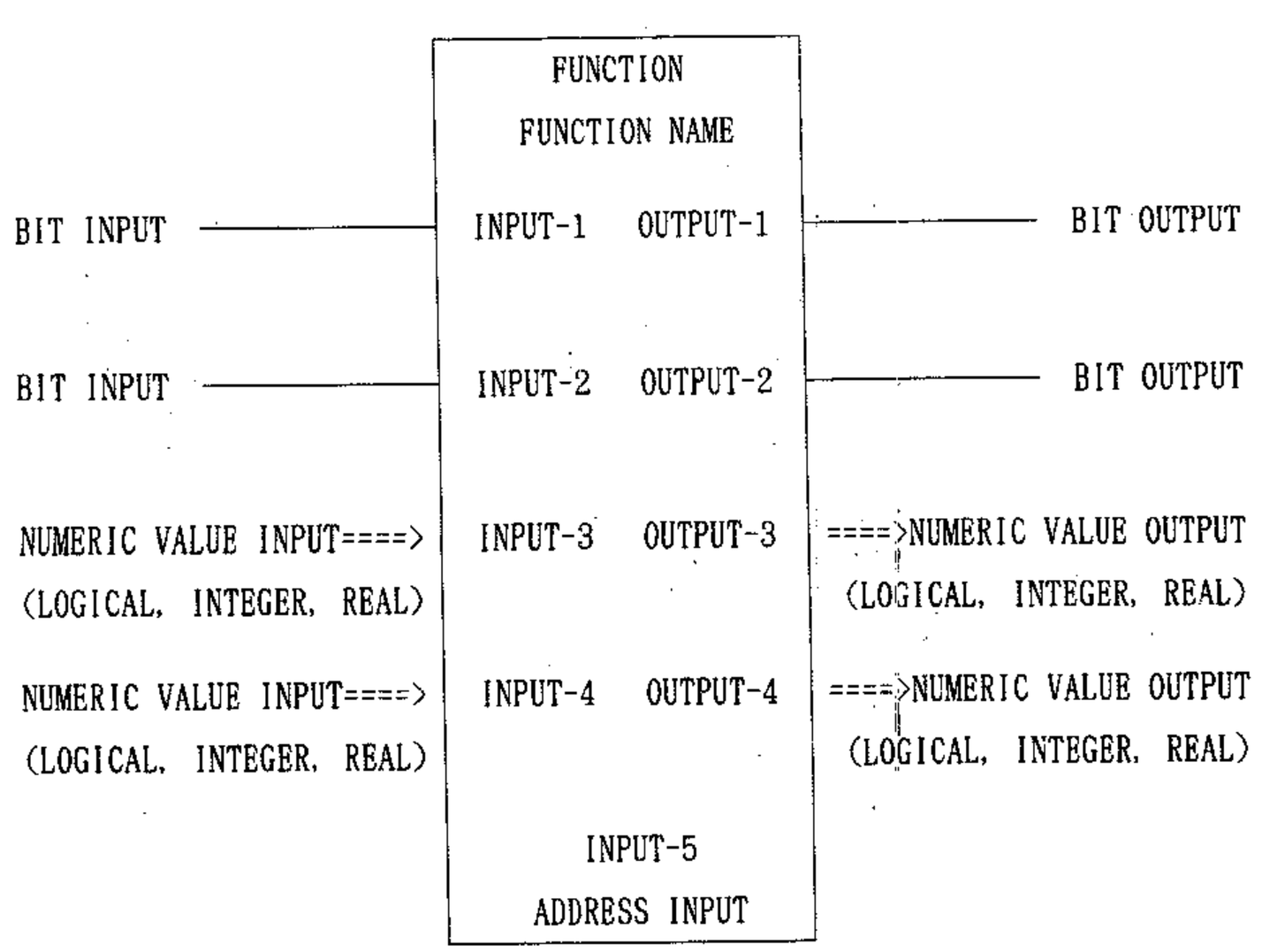


Fig. 4.3 Function graphic notation

(2) Function processing method

All variables (input, output, internal variables) used in a function are processed on the stack. In other words, each time a function is referenced from a drawing, a stack area is first allocated as shown in Fig. 4.4 and the input is copied in the stack area. Then the function processing program is executed. At the end of processing, the output is passed to the drawing and the stack area is finally freed. This method of function processing ensures that the function can reenter and eliminates allocation of a data area for function processing.

For structure type data such as an array, the top address is passed to a function which handles the data directly.

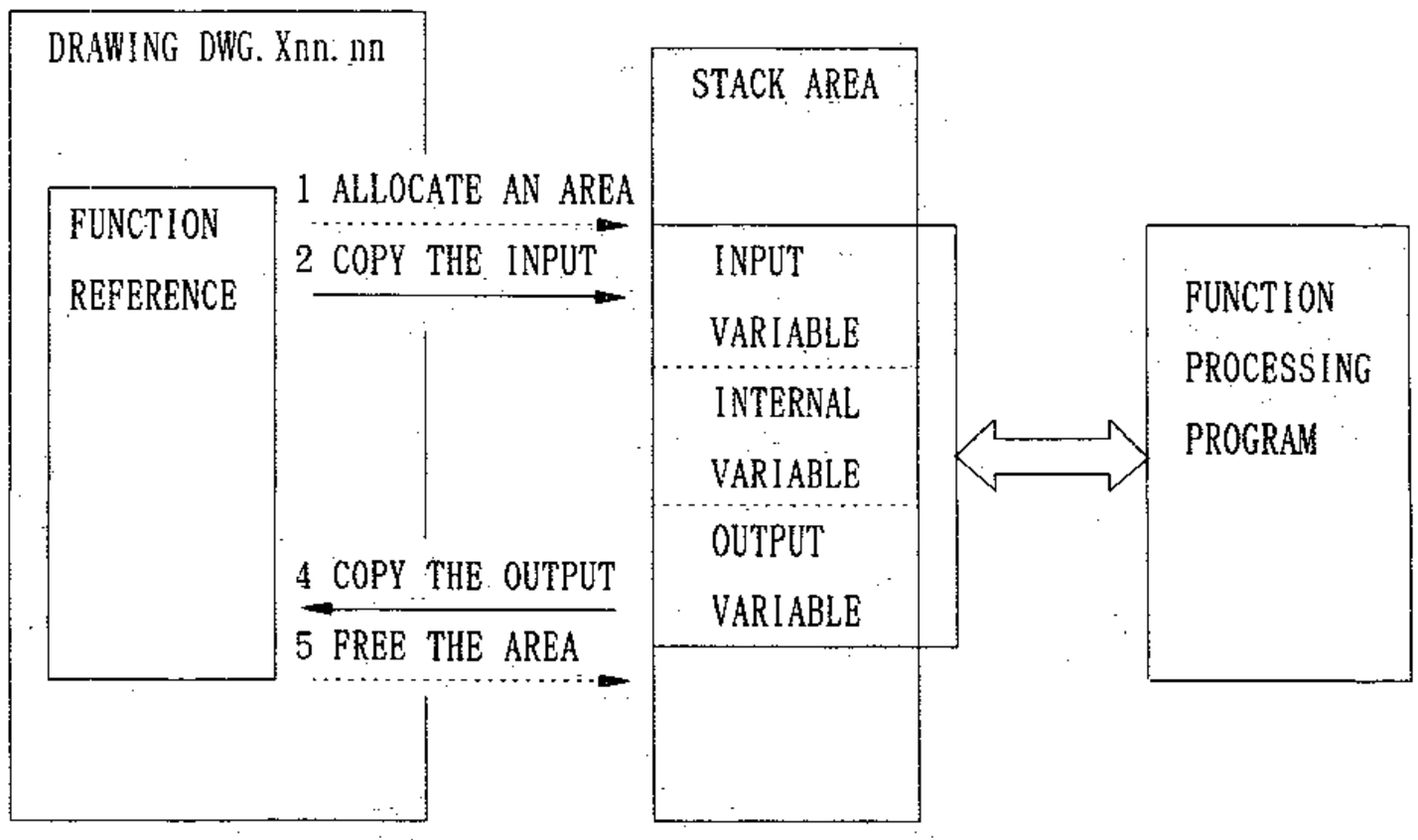


Fig. 4.4 Function processing method

5 VARIABLE MANAGEMENT

This section describes the management of CP-3300 variables.

The variables are classified by usage into groups each with a different attribute. A variable can be specified either directly by its address or through a symbol. The symbol management is also explained.

CONTENTS PAGE

5.1 VARIABLE SPECIFYING METHOD

The CP-3300 variable specifying method has two types as shown in Table 5.1 ; data address direct specifying and symbol specifying, and can mix in user program. When specifying a symbol, the symbols are linked to the data address in a symbol table as described in Par. 5.3.

Table 5.1 Variable specifying method

<table>
<tr><th>Variable
Specifying method</th><th>Description</th><th>Remarks</th></tr>
<tr><td>Data address
direct specifying</td><td>Set a data address according to Fig. 5.1.
<table><tr><th>Data Type</th><th>Format(Example)</th></tr><tr><td>Bit
Logical
Integer
Real
Address</td><td>MB00100Ax
MW00100x
MW00100x
MF00100x
MA00100x</td></tr></table></td><td>See par. 5.2.

x=I or J
(Index specifying)</td></tr>
<tr><td>Symbol specifying</td><td>Set a symbol by alphanumeric string to 8 characters.
<table><tr><th>Data Type</th><th>Format(Example)</th></tr><tr><td>Bit
Logical
Integer
Real
Address</td><td>RESET 1-A.x
MASKDATA.x
STIME-H.x
PCONST.x
PID-DATA.x</td></tr></table></td><td>See par. 5.3.

x=I or J
(Index specifying)</td></tr>
</table>

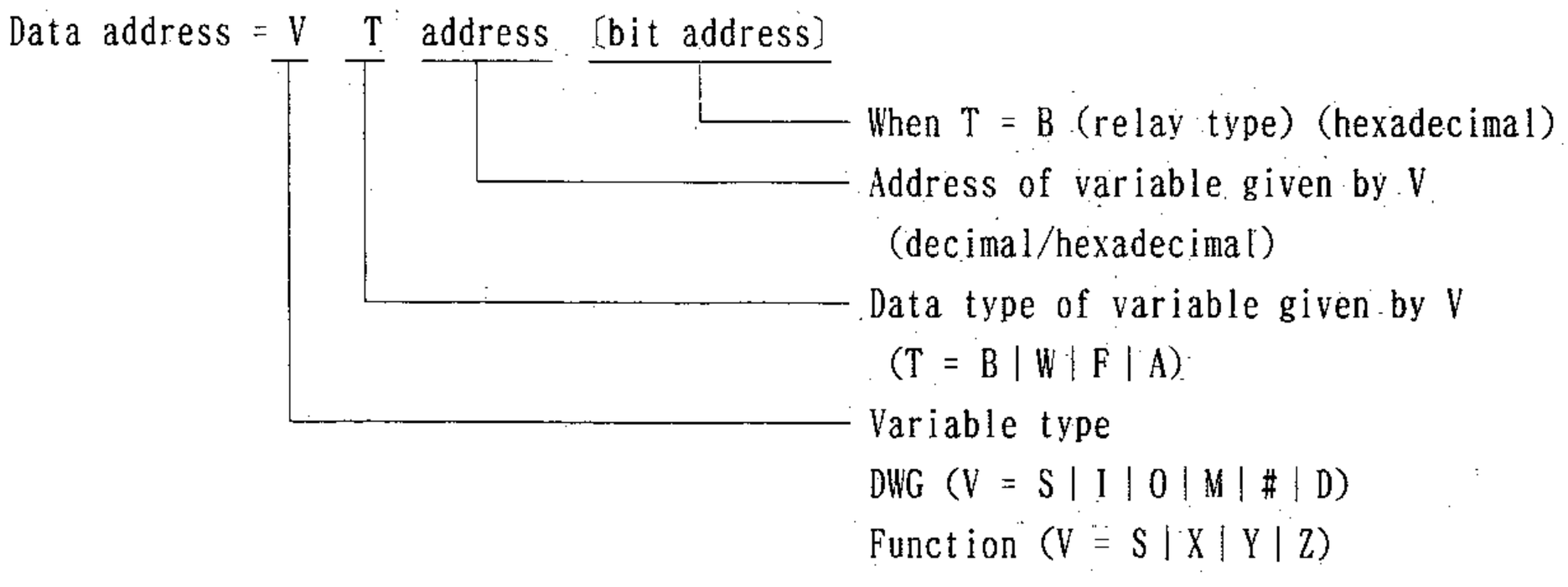


Fig. 5.1 Format of data address direct specifying

5.2 VARIABLE TYPES

The CP-3300 classifies the variables by usage and assigns each class an attribute.

Some of these variables can be used only in DWG, others only in functions, and still others in both DWG and functions.

(1) DWG variables
Six types of variables shown in Table 5.2 are available to each DWG :

Table 5.2 DWG variables

No.	Variable type	Variable specification method	Range	Content	Characteristic
1	System variable	SB, SW, SFnnnnn (SAnnnnn)	SW00000 to SW00767	System-provided variables. Calendar, scan time, etc.	System common
2	Input variable (process)	IB, IW, IFnnnn (IAnnnn)	IW0000 to IW0FFF	Process input (reference only) Local input : IW0000 to IW01FF Remote input : IW0200 to IW03FF CP-213 input : IW0400 to IW0BFF (512 wd/line) Link input : IW0C00 to IW0FFF	
3	Output variable (process)	OB, OW, OFnnnn (OAnnnn)	OW0000 to OW0FFF	Process output Local output : OW0000 to OW01FF Remote output : OW0200 to OW03FF CP-213 input : OW0400 to OW0BFF (512 wd/line) Link input : OW0C00 to OW0FFF	
4	DWG common variable	MB, MW, MFnnnnn (MAnnnnn)	MW00000 to MW16383	Variable shared by DWG. Used for I/F of DWG.	
5	Constant data	#B, #W, #Fnnnnn (#Annnnn)	#W00000 to #W00511	Variable that can only be referenced from a program. Can be referenced only by a relevant DWG. The actual use range is specified by the user in PP.	DWG dependent
6	DWG internal variable	DB, DW, DFnnnnn (DAnnnnn)	DW00000 to DW08191	Internal variable unique to each DWG. Can be used only by a relevant DWG. The actual use range is specified by the user in PP.	

(2) Function variables

Eight types of variables shown in Table 5.3 are available to each function :

Table 5.3 Function variables

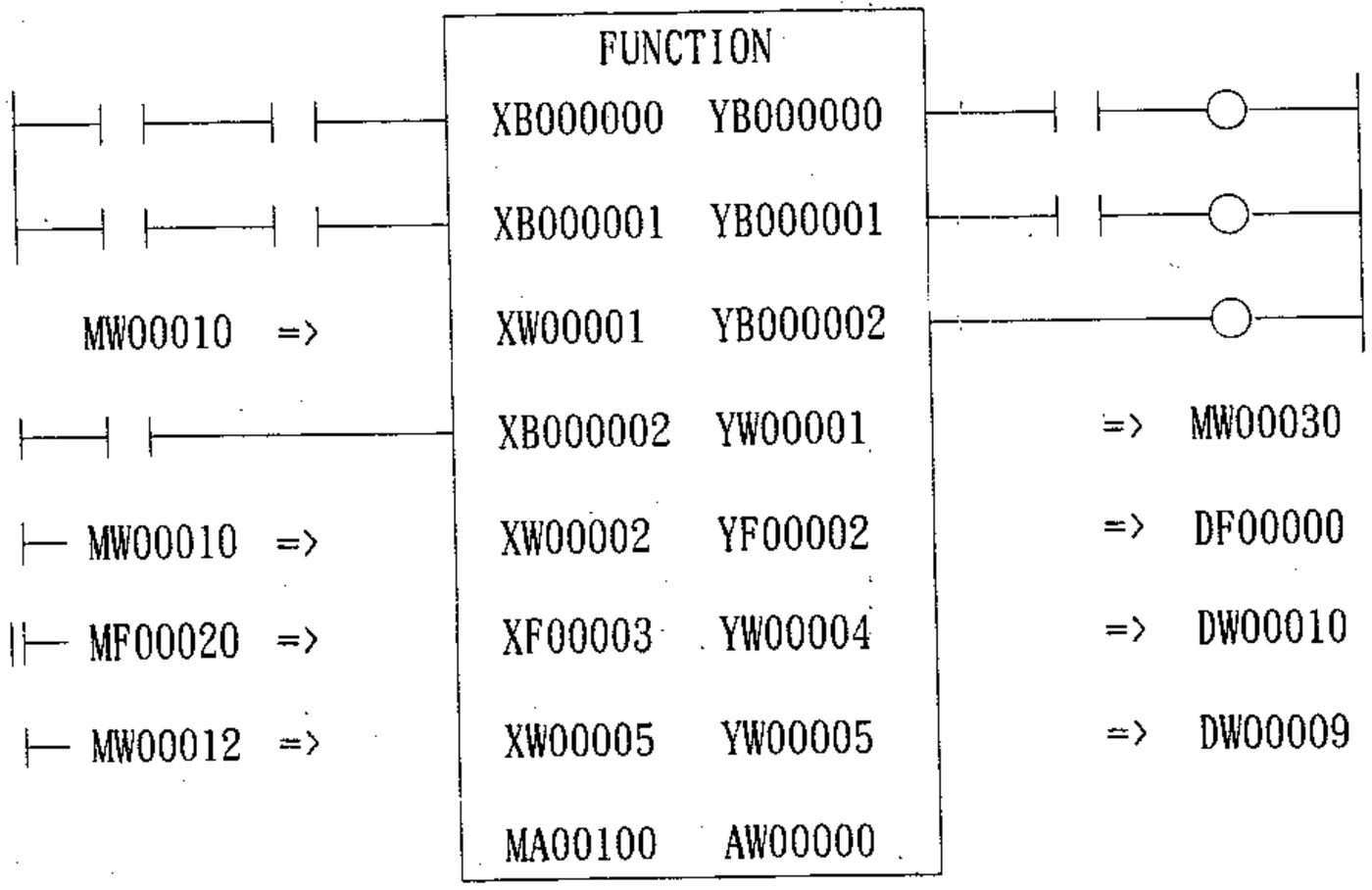
No.	Variable type	Variable specification method	Range	Content	Characteristic
1	Function input variable	XB, XW, XFnnnnn	XW00000 to XW00016	Input to a function. Bit input : XB000000 to XB00000F Word input : XW00001 to XW00016 Real number input : XF00001 to XF00015	Function dependent
2	Function output variable	YB, YW, YFnnnnn	YW00000 to YW00016	Output from a function. Bit output : YB000000 to YB00000F Word output : YW00001 to YW00016 Real number output : YF00001 to YF00015	
3	Function internal variable	ZB, ZW, ZFnnnnn	ZW00000 to ZW00063	Internal variable unique to each function. Function internal processing work	
4	Function external variable	AB, AW, AFnnnnn	AW00000 to AW08191	External variable with address input value as a base address. Can connect to {S, I, O, M, #, DAnnnnn}	
5	System variable	SB, SW, SFnnnnn		Same as DWG variables	System common
6	Input variable	IB, IW, IFnnnn			
7	Output variable	OB, OW, OFnnnn			
8	DWG common variable	MB, MW, MFnnnnn			

(3) Function inputs/outputs and function variables

The inputs/outputs in function reference are associated with the function variables shown in Table 5.4 :

Table 5.4 Function inputs/outputs and function variables

No.	Function I/O	Function variable
1	Bit input	The bit numbers increase continuously from XB000000 in sequence of bit input. (XB000000, XB000001, XB000002, ·····, XB00000F)
2	Word, real number input	The register numbers increase continuously from XW, XF00001 in sequence of word, real number input. (XW00001, XW00002, XW00003, ·····, XW00016)
3	Address input	The address input value corresponds to the zero address of external variable. (Input value = MA00100 : MA00100 = AW00000, MA00101 = AW00001, ···)
4	Bit output	The bit numbers increase continuously from YB000000 in sequence of bit output. (YB000000, YB000001, YB000002, ·····, YB00000F)
5	Word, real number output	The register numbers increase continuously from YW, YF00001 in sequence of word, real number output. (YW00001, YW00002, YW00003, ·····, YW00016)



(4) Program and variable reference range

Fig. 5.2 illustrates the above description.

Fig. 5.1 shows that DWG can handle the following variables

- Constant data
- DWG dependent variables
- System variables
- I/O variables (input variables and output variables)
- DWG common variables

The constant data and DWG dependent variables must belong to a particular DWG in question.

On the other hand, the figure shows that functions can handle the following variables :

- Function input variables
- Function output variables
- Function internal variables
- System variables
- I/O variables (input variables and output variables)
- DWG common variables

The figure also shows that by using function external variables, the following can be handled :

- Constant data
- DWG internal variables

These data must belong to DWG where the function is described.

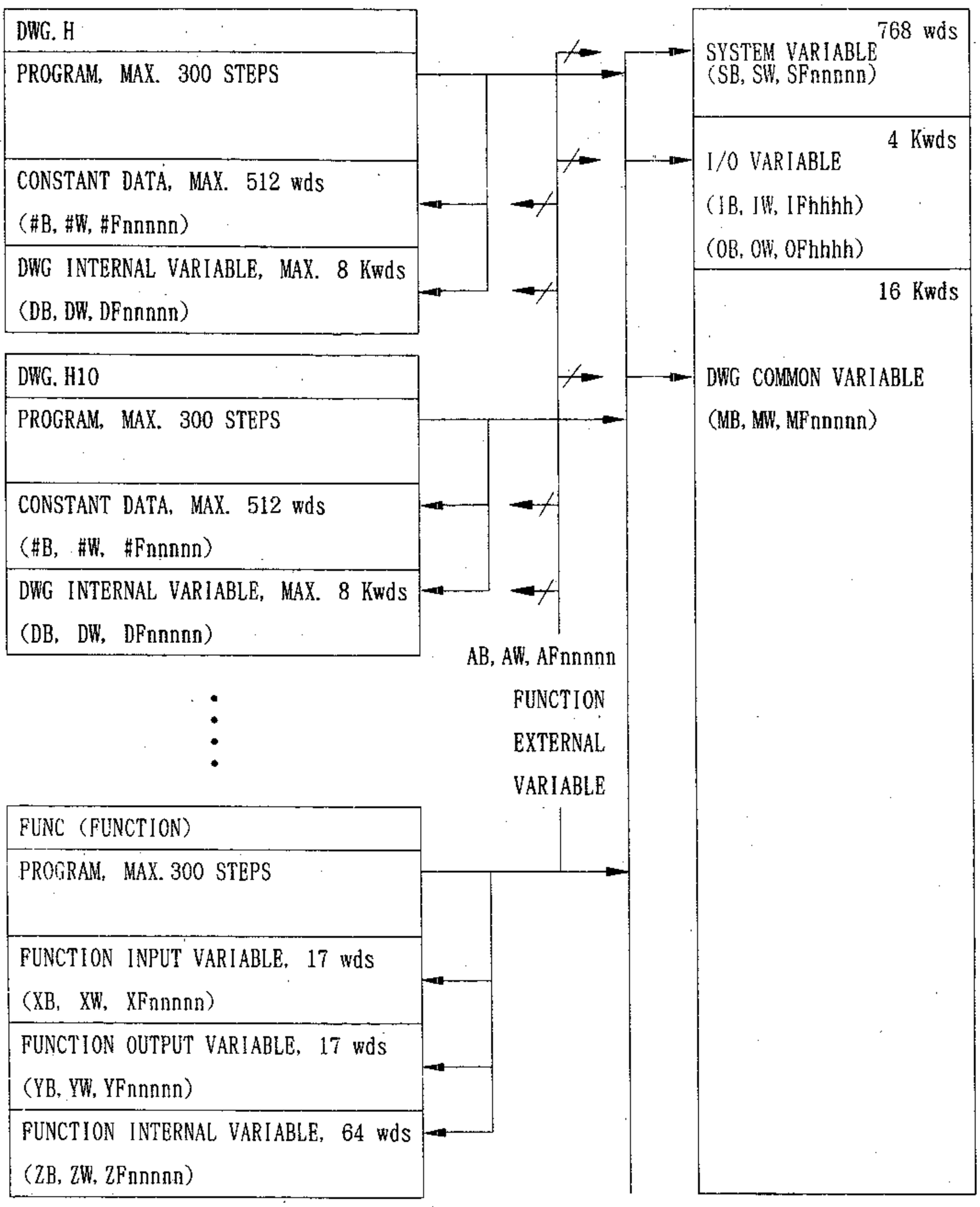


Fig. 5.2 Program and variable reference range

Notes:

1. The marks ⟋► and ◄⟋ show that it can reference as function external variable.
2. Variable number "nnnnn" is in decimal notation.
   Variable number "hhhh" is in hexadecimal notaion.

5.3 SYMBOL MANAGEMENT

(1) Symbol management in DWG
All Symbols used in DWG are managed by the DWG variable table shown in Table 5.5.

The user will specify a symbol registration and data address specification using the DWG variable table. The symbol registration, deletion, and change and data address specification and change can be done at any time during program creation. If an unregistered symbol is used in program creation, only the symbol is entered automatically into the DWG variable table. If write is performed with an unregistered symbol, a variable area inside the DWG is automatically allocated to the unregistered symbol.

To create a program with a data structure (array, index processing data, etc.), the data structure must be defined in the DWG variable table.

Up to 200 symbols can be registered for each drawing.

Table 5.5 DWG variable table

No.	Data Address	Symbol	Size	Remarks
0	SB00000A	PULSE1	1	
1	SB00000B	PULSE2	1	
2	SW00001	SCANTIME	1	
3	IB00AFC	STARTPBL	1	Data address is in hexadecimal notation.
4	0B0170A	STARTCOM	1	Data address is in hexadecimal notation.
5	MW00000	SPDMAS	1	
6	DB000000	WORK-DB	16	
7	DW00010	PIDDATA	10	
8	DW00020	LAUIN	1	
9	DW00021	LAUOUT	1	
• • •				
N				

(2) Symbol management in a function

All symbols used in a function are managed by the function variable table shown in Table 5.6.

The registration, deletion, and change of a symbol and the specification and change of a parameter number are performed the same as in DWG.

Table 5.6 Function variable table

No.	PARAM. No.	Symbol	Size	Remarks
0	XB000000	EXECOM	1	
1	XW00001	INPUT	1	
2	AW00000	PIDDATA	20	Specify the number of words of A variable for the size.
3	AW00001	P-GAIN	1	
4	AB00000F	ERROR	1	
5	YB000000	PIDEXE	1	
6	YW00001	PIDOUT	1	
7	ZB000000	WORKCOIL	1	
8	ZW00001	WORK1	1	
9	ZW00002	WORK2	1	
•				
•				
•				
N				

5.4 VARIABLE TABLE LINKING AND AUTOMATIC NUMBERING

(1) Linking variable tables

It is possible to define automatically undefined data addresses for system, input, output, DWG common variables by linking the variable tables created for both DWG and functions to the upper level variable table. Table 5.7 shows the linkable variable tables :

Table 5.7 Linkable variable tables

No.	Variable table	Upper level variable table		
		System	Basic drawing	Detailed drawing
1	Basic drawing variable table	○	×	×
2	Detailed drawing variable table	○	○	×
3	Expanded drawing variable table	○	○	○
4	Function variable table	○	×	×

(2) Assigning automatically data addresses

In each variable table, for all variables with undefined data addresses, data addresses can be assigned automatically for each variable (a sequence of data addresses is allocated automatically) by specifying the top data address.

Table 5.8 shows the variables that permit automatic numbering for the DWG and function variable tables :

Table 5.8 Automatic numbering of data addresses

No.	DWG variable table		Automatic numbering	No.	Function variable table		Automatic numbering
1	System variable	S	×	1	System variable	S	×
2	Input variable	I	×	2	Function input variable	X	×
3	Output variable	O	×	3	Function output variable	Y	×
4	DWG common variable	M	○	4	Function internal variable	Z	○
5	Constant data	#	○	5	Function external variable	A	×
6	DWG internal variable	D	○				

Note: The marks × and ○ in the automatic numbering column indicate the automatic numbering is not possible and possible, respectively.

6 DATA HANDLING

This section describes data handling in the CP-3300.

There are five data types, including bit, integer, and real.
They are selected to match the purpose.

This section also explains the relationship between the CPU internal registers and the operation.

CONTENTS PAGE

6.1 BASIC DATA TYPES

The CP-3300 can use five kinds of data types listed in Tabel 6.1.

Each data type is specified by the second character in the data address direct specification method. A user program uses, in various operations, the first four kinds of data types in Table 6.1.

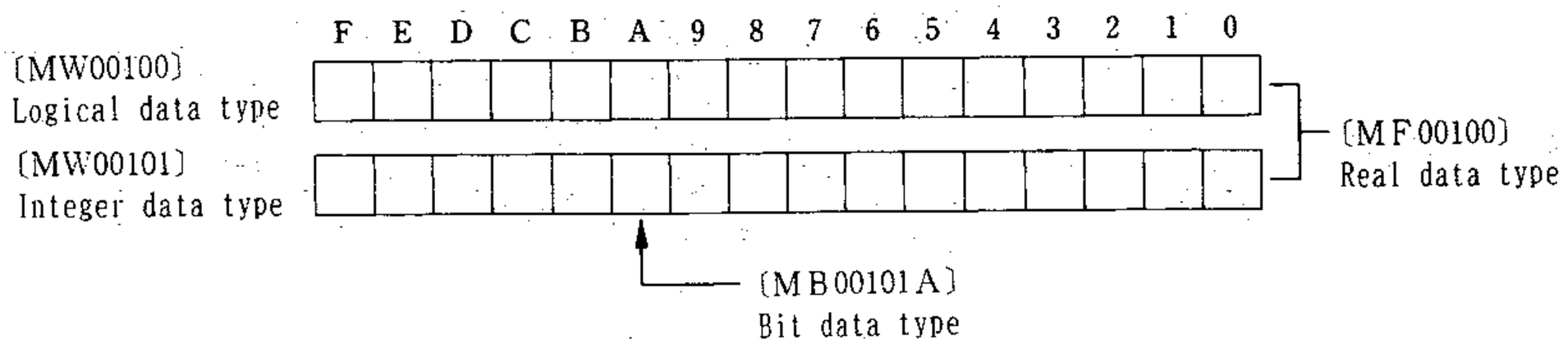
The address type data are used solely to specify a variable address to be passed to a function.

Table 6.1 Basic data types

No.	Data type	Type specification	Value range	Remarks
1	Bit	B	ON, OFF	Used in a relay circuit.
2	Logical	W	0000H to FFFFH	Used in a logical operation.
3	Integer	W	-32,768 to +32,767	Used in a numerical operation.
4	Real	F	±(1.17E-38 to 3.40E+38)	Used in a numerical operation. (Floating-point operation)
5	Address	A	0 to 16,383	Used only for entering a variable address to a function.

【Variable area and data type 】

The relationship between the variable area and data types is shown below with a DWG common variable as an example.



6.2 CPU INTERNAL REGISTERS

The CP-3300 provides registers shown in Table 6.2 inside the CPU and processes user programs using these registers.

Table 6.2 CPU internal registers

No.	Register	Bit width	Use
1	A register	16	Used for integer and logical operation.
2	D register	16	Used as an extended register for integer multiplication and division. Not availabel to the user at this time.
3	F register	80	Used for real number operation (floating-point operation).
4	B register	1	Used in a relay circuit operation.
5	Branch point register	15	Used in a relay circuit operation. Not available to the user at this time.
6	Join point register	16	Used in a relay circuit operation. Not available to the user at this time.
7	I register	16	Used as an index register (I).
8	J register	16	Used as an index register (J).

6.3 SWITCHING THE NUMERICAL OPERATION REGISTERS

In a user program, A register and F register are used in a numerical operation as shown in Table 6.2. A register is used for integer and logical operation where only the operation with logical and integer type variables is permitted. F Register is used for real type numeric operation and the operation with integer and real variables is permitted. (The integer type variable is internally converted to real type before operation.)

Switching the numerical operation registers is performed with two types of load instructions as shown in Table 6.3.

Load instructions
├─: Load A register and start A register operation (integer and logical operation)
||─: Load F register and start F register operation (real number operation)

Table 6.3 Switching the numerical operation registers by load instructions

```
├─ MW00010  + MW00020  −  MW00021  => MW00018 ─┐ Integer operation by A register.
           × MW00022  ÷  MW00023  => MW00025   │ Only integer and logical type
           + MW00026  +  MW00027  => MW00028   │ variables can be used.
⋮                                              ┘


||─ MW00030  + MF00050  −  MF00052  => MF00058 ─┐ Real number operation by F register.
           × MW00054  ÷  MF00056  => MW00026   │ Both integer and real type
           − MF00060  +  MF00062  => MF00064   │ variables can be used. The integer
⋮                                              ┘ type variable is converted to real
                                                 type for operation.

├─ MW00010  + MW00011  −  MW00012  => MW00013 ─┐ Integer operation by A register.
           × MW00014  −  MW00015  => MW00016   │
⋮                                              ┘
⋮
```

7 INSTRUCTIONS

This section lists all instructions available in the CP-3300.

Each instruction is accompanied by a brief explanation.

For details, see Section 8 and Appendix 3.

CONTENTS PAGE

7.1 INSTRUCTION SET

The instruction set available in the CP-3300 are grouped into the instruction set types shown in Table 7.1.

Table 7.1 Instruction set

No.	Instruction set type	Instructions
1	Program control instruction	SEE START FOR WHILE ON OFF IFON IFOFF ELSE END
2	Direct I/O instruction	IN OUT
3	Sequence circuit instruction	⊣⊢ ⊣/⊢ —[↓]— —[↑]— —[T]— —[T]— —○—
4	Logical operation instruction	∧ ∨ ⊕
5	Numerical operation instruction	⊢ ⊢| ⇒ + − ++ -- × ÷ INC DEC MOD REM
6	Numerical conversion instruction	INV COM ABS BIN BCD PARITY
7	Numerical comparison instruction	< ≦ = ≠ ≧ >
8	Data transfer instruction	ROTL ROTR MOVB MOVW XCHG
9	Basic function instruction	SQRT SIN COS TAN ASIN ACOS ATAN EXP LN LOG
10	DDC instruction	DZA DZB LIM PI PD PID LAG LLAG FGN IFGN LAU SLAU
11	SFC instruction	SFC ⊥ ⊻ + ABOX AEND SBOX
12	System standard function	ISET-213 TRACE FINFOUT COUNTER COMM READ WRIT LINK-MST LINK-SLV LINK-SND LINK-RCV NET-MBUS NET-PEER NET-BROD NET-DIAG NET-SND NET-RCV IO-TRACE NINP NOUT

7.2 INSTRUCTIONS

Table 7.2 Instructions

Type	Name	Instruction word	Data type B	Data type W	Data type F	Instruction with [ ]	Description
Program control instruction (par. 8. 1)	Detailed drawing reference Expanded drawing reference	SEE				○	SEE H01. 01 (detailed. expanded drawing number)
	Batch drawing startup	START				○	START B01 (batch detailed drawing number)
	FOR syntax	FOR FEND					Repeat execution syntax -1 For (V=a to b by c) V: any integer variable. I. J allowed. a, b, c: any integer value allowed.
	WHILE syntax	WHILE ON/OFF WEND					Repeat execution syntax -2 WEND: End of WHILE-ON/OFF instruction
	IF syntax	IFON/ IFOFF ELSE IEND					Conditional execution syntax A syntax without ELSE is allowed. IEND: End of IFON/IFOFF instruction
	END	FEND WEND IEND DEND					A specialized END instruction is used for each of the above syntax. Use a DEND instruction for a drawing END.
Direct I/O instruction (par. 8. 2)	Input instruction	IN					IN IO_ADDR —○— =>MW00001 B Register is ON with an input error. Indexing allowed.
	Output instruction	OUT					├—MW00100 OUT IO_ADDR —○— Register B is ON with an output error. Indexing allowed.

Note: In the data type column, B denotes a bit type, W an integer type, and F a real type.
The mark ○ indicates the data type in the operand of the instruction.

The ○ in the "instruction with [ ] " column indicates that the instruction permits conditional execution by the value of the preceding B register.

Table 7.2 Instructions (Cont'd)

Type	Name	Instruction word	Data type B	Data type W	Data type F	Instruction with { }	Description
Sequence circuit instruction (par. 8.3)	Normally open contact	⊣ ⊢	○				No restrictions on the series connection. All variables of bit type can be specified as relay numbers.
	Normally closed contact	⊣/⊢	○				
	Rising pulse	⊣↑⊢	○				
	Falling pulse	⊣↓⊢	○				
	ON delay timer	⊣[T]⊢	○				Preset value Count register ⊣[T]⊢ Preset value: M, D, # variable, constant Count register: M, D variable A count register can be omitted during program entry. (Automatically assigned to D variable.)
	OFF delay timer	⊣[T]⊢	○				
	Coil	⊣○⊢	○				⊢DW00200 = 00001 ⊣○⊢ IFON Holds the content of B register which is the result of comparison.
	Branch, join						Branch, join instructions can be connected to all the above relay type instructions.
Logical operation instruction (par. 8.5)	Logical AND	∧		○		○	All variables and constants of W type can be specified. No compound instruction with NOT (NAND, for example).
	Logical OR	∨		○		○	All variables and constants of W type can be specified.
	Exclusive logical OR	⊕		○		○	All variables and constants of W type can be specified.

Table 7.2 Instructions (Cont'd)

Type	Name	Instruction word	Data type B	Data type W	Data type F	Instruction with [ ]	Description
Numerical operation instruction (par. 8.4)	Integer type load	⊢		○		○	Start an integer type operation. (Only allowed for integer operation.) A real type variable cannot be used. ⊢ MW00280+00100 => DW00220
	Real type load	| ⊢		○	○	○	Start a real type operation. (Real and integer can be mixed.) |⊢ MW00208 (Execute conversion to real) + MF00200 × DW00102 (Execute conversion to real)
	Store	=>		○	○	○	Store an operation result to a specified variable. Type conversion is executed when started with |⊢ and ending with => (integer variable). (I, J not allowed.)
	Add	+		○	○	○	Normal numerical addition (Operation error possible.) ⊢MW01000 + DW00200 => MW00202 All variables and constants can be specified.
	Subtract	−		○	○	○	Normal numerical subtraction (Operation error possible.) ⊢ DW00300− 00200 => MW00202 All variables and constants can be specified.
	Extended add	++		○		○	Closed numerical addition (No operation error) 32767 + 1=−32768 0 →32767 →−32768 → 0
	Extended subtract	--		○		○	Closed numerical subtraction (No operation error) −32768 −1 =32767 0 →−32768 →32767 → 0
	Multiply	×		○	○	○	Can be used alone with real type data. With an integer type, × and÷must be paired.
	Divide	÷		○	○	○	Can be used alone with real type data. With an integer type, × and ÷ must be paired.
	Increment	INC		○		○	Add 1 to a specified variable.
	Decrement	DEC		○		○	Subtract 1 from a specified variable.
	Integer type remainder	MOD		○			⊢ MW00100×1000÷121 MOD => MW00101 The quotient disappears in this example.
	Real type remainder	REM			○		|⊢ MF00200 REM MF00202

Table 7.2 Instructions (Cont'd)

Type	Name	Instruction word	Data type B	Data type W	Data type F	Instruction with [ ]	Description
Numerical conversion instruction (par. 8.7)	Sign reverses	INV		○	○	○	├— MW00100 INV
	Complement of 1	COM		○		○	├— MW00100 COM MW00100 =FFFFh Operation result = 0000h
	Absolute value conversion	ABS		○	○	○	├— MW00100 ABS
	Binary conversion	BIN		○		○	├— MW00100 BIN
	BCD conversion	BCD		○		○	├— MW00100 BCD
	Parity conversion	PARITY		○		○	├— MW00100 PARITY
Numerical comparison instruction (par. 8.6)	<	<		○	○		As a result of comparison operation, the B register is set (ON) or reset (OFF).
	≦	≦		○	○		
	=	=		○	○		
	≠	≠		○	○		
	≧	≧		○	○		├— MW00000<10000 —○—┤
	>	>		○	○		
Data transfer instruction (par. 8.8)	Bit rotate left	ROTL	○			○	ROTL MB00100A Address N=1 Count W=20 Data width
	Bit rotate right	ROTR	○			○	ROTR MB00100A N=1 W=20
	Bit transfer	MOVB	○			○	MOVB MB00100A Transfer source → MB00200A Transfer destination W=20 Data width
	Word transfer	MOVW		○		○	MOVW MW00100 → MW00200 W=20
	Exchange transfer	XCHG		○		○	XCHG MW00100 → MW00200 W=20

Table 7.2 Instructions (Cont'd)

Type	Name	Instruction word	Data type B	Data type W	Data type F	Instruction with [ ]	Description
Basic function instruction (par. 8.9)	Square root	SQRT			○	○	Square root of negative is the value of the square root of the absolute value times -1. ||—MF00100 SQRT
	Sine	SIN			○	○	Input = degrees ||—MF00100 SIN
	Cosine	COS			○	○	Input = degrees ||—MF00100 COS
	Tangent	TAN			○	○	Input = degrees ||— MF00100 TAN
	Arcsine	ASIN			○	○	||— MF00100 ASIN MF00102 ASIN (MF00100/MF00102)
	Arccosine	ACOS			○	○	||— MF00100 ACOS MF00102 ACOS (MF00100/MF00102)
	Arctangent	ATAN			○	○	||— MF00100 ATAN MF00102 ATAN (MF00100/MF00102)
	Exponential	EXP			○	○	||— MF00100 EXP
	Natural logarithm	LN			○	○	||— MF00100 LN
	Common logarithm	LOG			○	○	||— MF00100 LOG
DDC instruction (par. 8.10)	Dead band A	DZA		○	○	○	├— MW00100 DZA 100
	Dead band B	DZB		○	○	○	├— MW00100 DZB 100
	Upper/lower limit	LIM		○	○	○	Lower limit Upper limit ├— MW00100 LIMIT −1000 1000
	PI control	PI		○	○	○	├— MW00100 PI MA00200
	PD control	PD		○	○	○	├— MW00100 PD MA00200
	PID control	PID		○	○	○	├— MW00100 PID MA00200
	First order lag	LAG		○	○	○	├— MW00100 LAG MA00200
	Lead lag	LLAG		○	○	○	├— MW00100 LLAG MA00200
	Function generator	FGN		○	○	○	├— MW00100 FGN #A00200
	Inverse function generator	IFGN		○	○	○	├— MW00100 IFGN #A00200
	Linear accelerator 1	LAU		○	○	○	├— MW00100 LAU MA00200
	Linear accelerator 2	SLAU		○	○	○	├— MW00100 SLAU MA00200

Table 7.2 Instructions (Cont'd)

Type	Name	Instruction word	Data type B	Data type W	Data type F	Instruction with [ ]	Description
SFC instruction (par. 8.11)	SFC execution	SFC					SFC MASTERSQ
	A contact transition decision	⊥ ⊤					Transition condition specification ⊥ ⊤ IB00010A (Indexing not allowed)
	B contact transition decision	⊥ ⊤					⊥ ⊤ MB00012B (Indexing not allowed)
	Timer transition decision	+					Transition Change timer preset value + 10.000 (Indexing not allowed)
	Action box	ABOX					ABOX-010: Start step-10 action box.
	Action box	SBOX					Action box that executes only one scan when the execution conditions are satisfied.
	Action box end	AEND					End of SFC action box.

Table 7.2 Instructions (Cont'd)

Type	Name	Instruction word	Data type B	Data type W	Data type F	Instruction with ( )	Description
System standard function (Appendix 3)	Trace function	TRACE					Data trace execution control
	Counter	COUNTER					Up/down counter
	FINFOUT	FINFOUT					First-in first-out function
	213IF initialization	ISET-213					213IF initial data setting
	COMM communication function	COMM					COMM memobus communication master function
	FABUS-II memobus communication function	LINK-MST					FABUS-II memobus communication master function
		LINK-SLV					FABUS-II memobus communication slave function
		LINK-SND					FABUS-II general message transmission function
		LINK-RCV					FABUS-II general message receive function
	ASCII communication function	READ					ASCII input function
		WRIT					ASCII output function
	YENET Communication function	NET-MBUS					YENET memobus message transmission master function
		NET-PEER					YENET selective command transmission function
		NET-BROD					YENET complete station broadcasting function
		NET-DIAG					YENET node diagnosis function
		NET-SND					YENET data send function
		NET-RCV					YENET data receive function
	I/O trace	I/O TRACE					FDS device data trace via CP-213 transmission
	I/O function	NINP					Local I/O direct input function
		NOUTP					Local I/O direct output function

8 PROGRAMMING

This section describes the specific method of using CP-3300 instructions.

See section 7 for the list of all instructions available in the CP-3300.

CONTENTS PAGE

This section explains the specific method of using each of the CP-3300 instructions.

For an instruction with an operand, specify the operand with a register number, a relay number, or a symbol. A symbol begins with a letter (including $) and has up to eight characters.

The following abbreviations are used in this section:

(1) Selection

〔Format〕

```
┌                  ┐
|  ..............  |
|  ..............  |
|  ..............  |
└                  ┘
```

Select one of the operands.

〔Example〕

```
IN ┌                                  ┐
   |  integer type variable           |
   |  integer type variable with index|
   |  integer type constant           |
   |  integer type constant with index|
   └                                  ┘
```

The above notation indicates all of the following formats:

IN 〈integer type variable〉
IN 〈integer type variable with index〉
IN 〈integer constant〉
IN 〈integer constant with index〉

(2) Assignment

〔Format〕 Y : = X

Replace the value of Y with that of X.

8.1 PROGRAM CONTROL INSTRUCTIONS

8.1.1 SEE instruction

〔Format〕 SEE <drawing number>

〔Explanation〕 A SEE instruction is used to reference a detailed drawing from a basic drawing or an expanded drawing from a detailed drawing. Reference between drawings of different kinds is not allowed. For example, SEE H20.01 cannot be described in DWG L20.

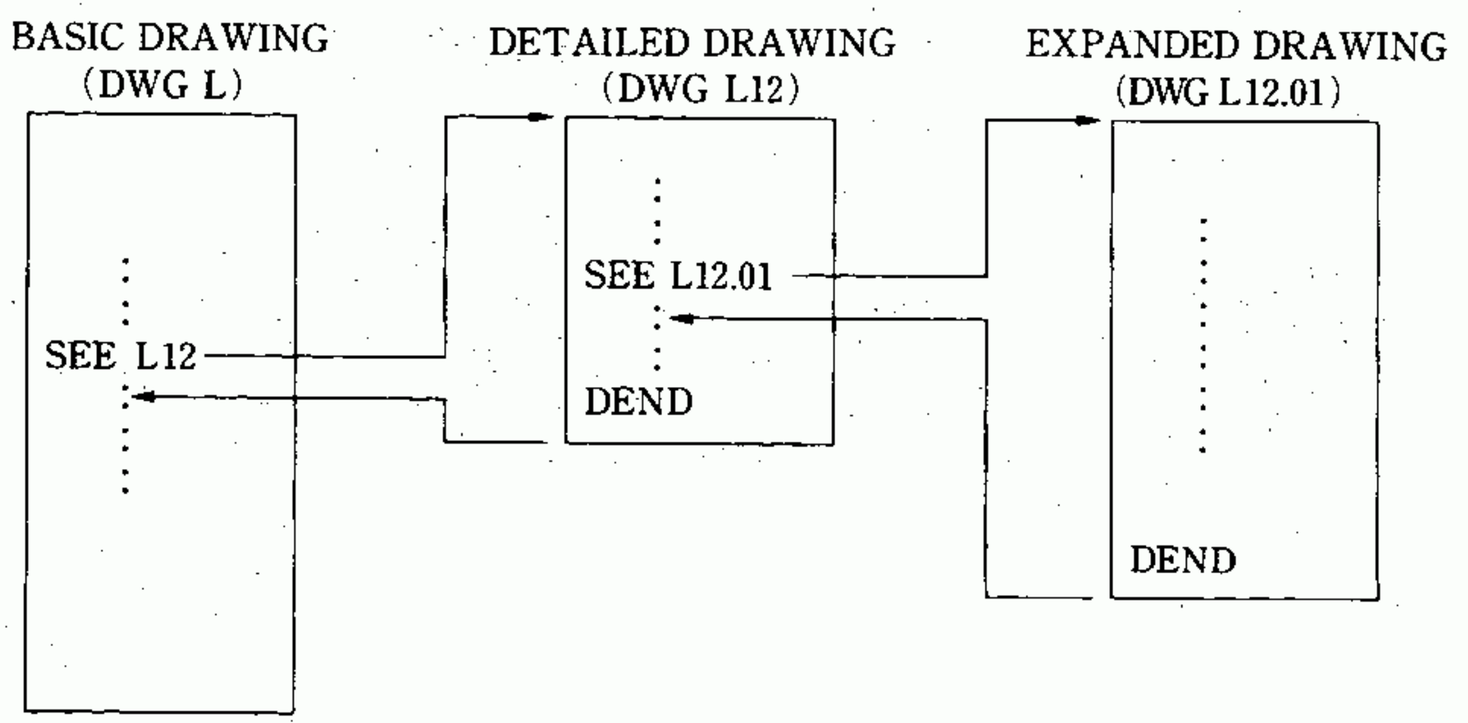


Fig. 8.1 Referencing detailed and expanded drawings with a SEE instruction

〔Example〕 SEE L12

8.1.2 START instruction

〔Format〕 START <detailed drawing number>

〔Explanation〕 A START instruction is used to start up batch detailed drawings. This instruction is allowed only in a low-speed scan basic drawing (DWG L). The started batch detailed drawings are executed one drawing at a time for each scan after the low-speed scan.

Note: Batch processing drawings have no basic drawing.
To reference a batch expanded drawing from a batch detailed drawing, use SEE instruction.

〔Example〕 START B01

8.1.3 FOR syntax

〔Format〕

```
FOR V = A to B by C
    instruction sequence (processing program)
FEND
```

〔Explanation〕 An instruction sequence enclosed with a FOR instruction and a corresponding FEND instruction is executed a specified number of times 〔N = (B-A)/C 〕. The variable V starts with an initial value A and incremented by C for each repeat execution. The following variables may be used for V, A, B, and C:

V : integer type variable, integer type variable with index, index register (I, J)

A, B, C : integer type variable, integer type variable with index, constants, index register (I, J)

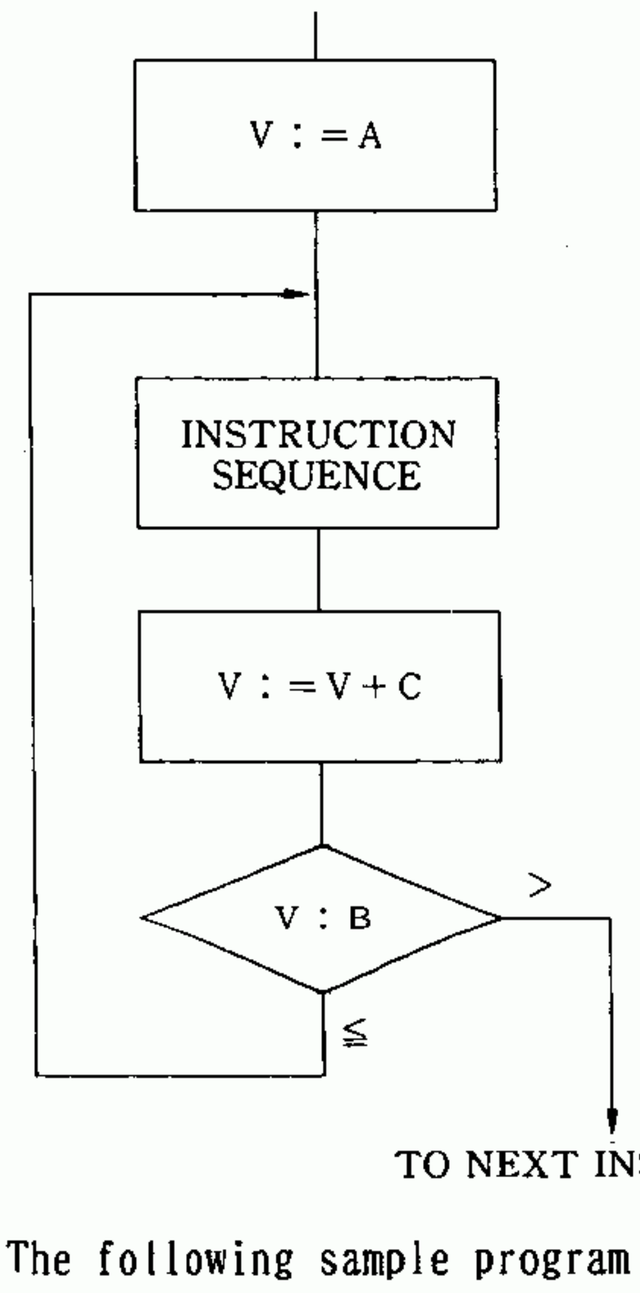


Fig. 8.2 FOR syntax execution

〔Example〕 The following sample program obtains a total of 100 registers from MW00100 to MW00199 in MW00200.

```
    ├─ 00000                                  => MW00200
FOR J = 00000 to 00099 by 00001
    ├─ MW00200 + MW00100j                     => MW00200
FEND
```

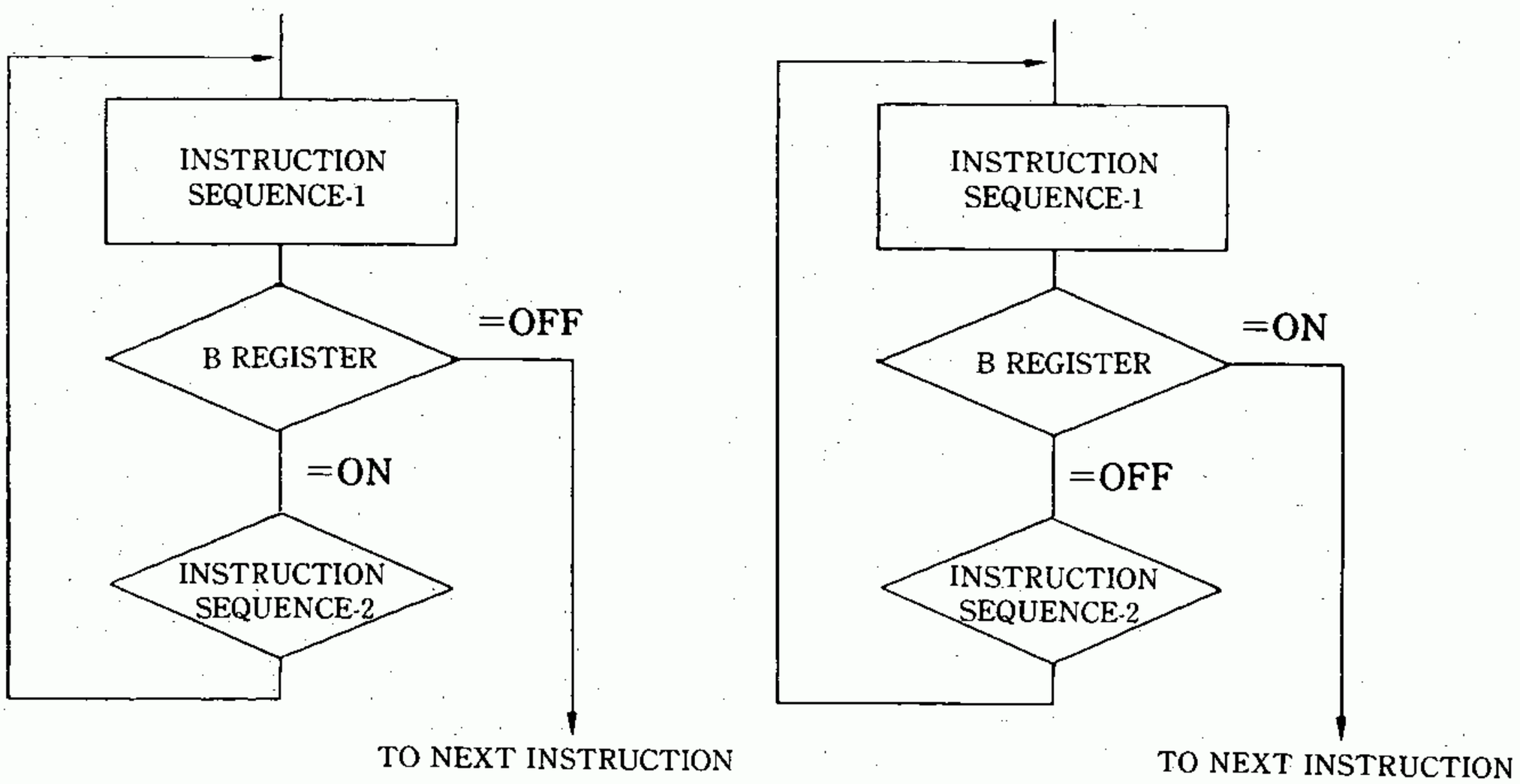
8.1.4 WHILE syntax

〔Format〕

```
WHILE
    instrucation sequence-1 (repeat condition decision)
ON/OFF
    instrucation sequence-2 (processing program)
WEND
```

〔Explanation〕 When the conditions of instruction sequence-1 and ON (or OFF) instruction are satisfied, a block between WHILE and WEND is repeated.

The condition of executing instruction sequence-2 is determined by the status of B register immediately preceding the ON (or OFF) instruction, as shown in Fig. 8.3. If the condition fails to be established by the first execution of instruction sequence-1, instruction sequence-2 will not be executed at all and the program will move to the instruction next to the WEND instruction.



(1) WHILE-ON-WEND SYNTAX (2) WHILE-OFF-WEND SYNTAX

Note: The program should be written so that the condition of a WHILE syntax (instruction sequence-1) may not ultimately be satisfied. If the system cannot get out of the loop, the watchdog timer will act to cause the system to go down.

Fig. 8.3 WHILE syntax execution

〔Example〕 The following sample program obtains a total of 100 registers from MW00100 to MW00199 in MW00200:

```
├─ 00000                              => I
                                       => MW00200
┌─ WHILE
│  ├─ I          < 00100
├─ ON
│  ├─ MW00200 + MW00100i              => MW00200
│  ├─ I        + 00001                => I
└─ WEND
```

8.1.5 IF syntax

Two formats of IF syntax are considered depending on presence or absence of an exclusive condition. They are described separately below, but there is no essential difference between them.

(1) IF syntax-1

〔Format〕

```
IFON/IFOFF
  instruction sequence (processing program)
IEND
```

〔Explanation〕 When the system encounters an IFON instruction, the instruction sequence between IFON and IEND is executed if the B register value is ON and not executed if OFF.

On the other hand, for an IFOFF instruction, the instruction sequence between IFOFF and IEND is executed if the B register value is OFF and not executed if ON.

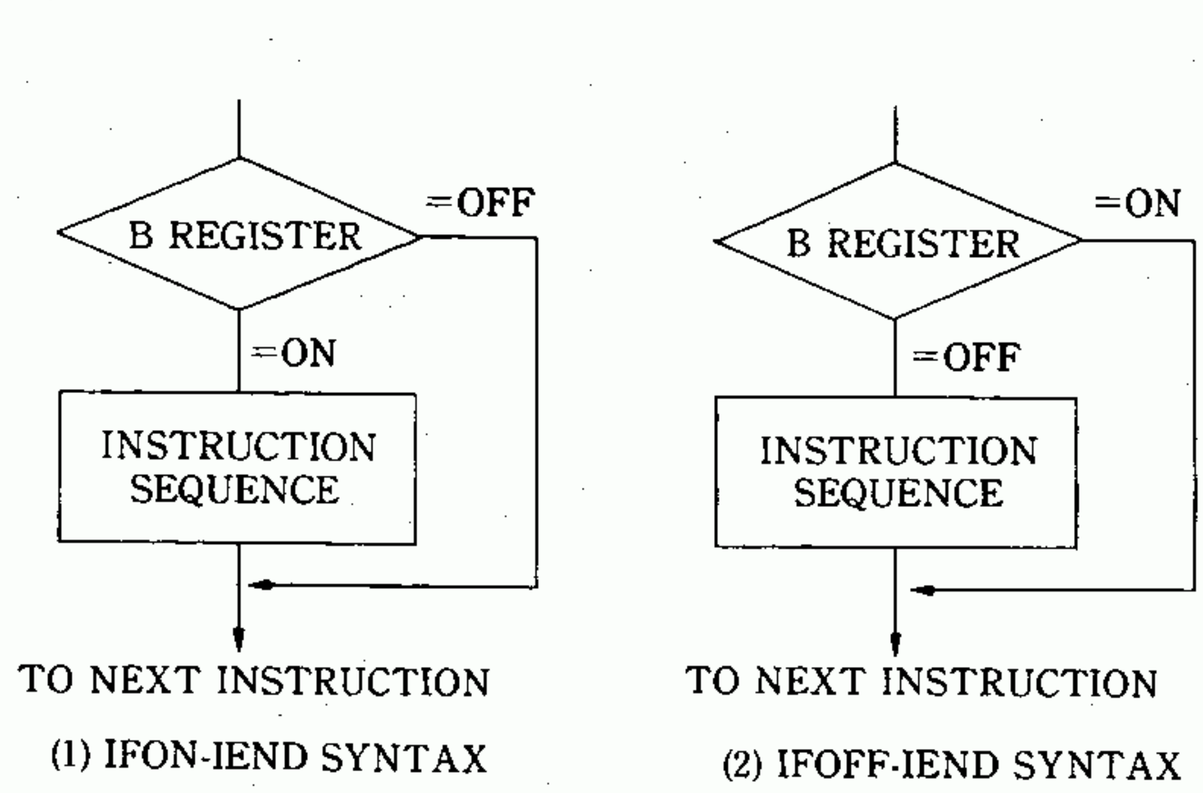


Fig. 8.4 IF syntax execution (1)

〔Example〕 If relay MB000108 is ON, set the content of DW00021 to 0.

```
  MB000108
 ├──┤ ├──
IFON
 ├─  00000          => DW00021
IEND
```

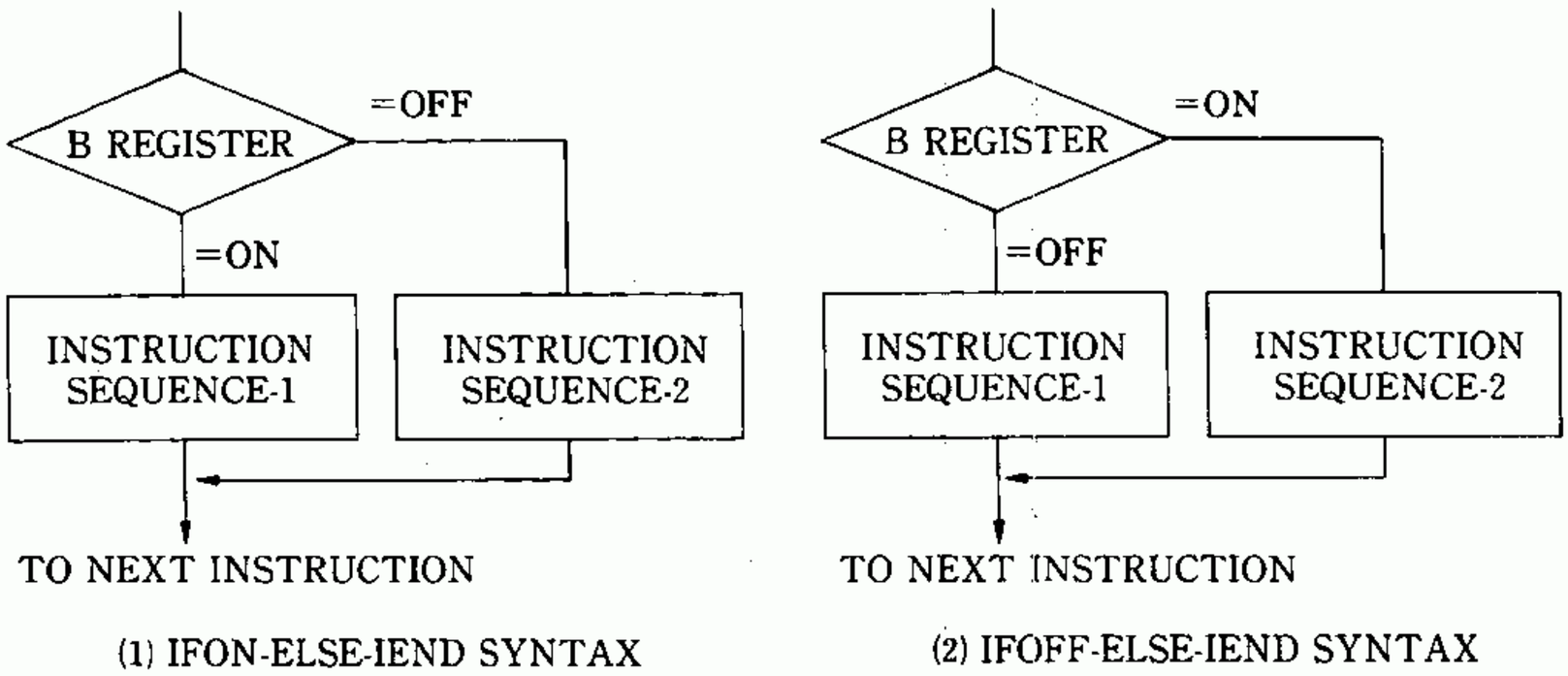
(2) IF syntax-2

〔Format〕

```
┌─ IFON/IFOFF
│    instruction sequence-1
├─ ELSE
│    instruction sequence-2
└─ IEND
```

〔Explanation〕 When the system encounters an IFON instruction, only instruction sequence-1 is executed if the B register value is ON. Instruction sequence-2 is not executed. Conversely, only instruction sequence-2 is executed and not instruction sequence-1 if OFF.

On the other hand, for an IFOFF instruction, instruction sequence-1 is executed and not instruction sequence-2 if the value is OFF. If ON, instruction sequence-2 is executed and not instruction sequence-1.



(1) IFON-ELSE-IEND SYNTAX (2) IFOFF-ELSE-IEND SYNTAX

Fig. 8.5 IF syntax execution (2)

〔Example〕 Set the content of DW00011 to 0 if the content of DW00010 is a positive number and to 1 if a negative number.

```
├─ DW00010   ≧   00000
IFON
├─ 00000              =>  DW00011
ELSE
├─ 00001              =>  DW00011
IEND
```

8.2 DIRECT I/O INSTRUCTIONS

Direct I/O instructions are used to perform direct I/O in the user program independenty from system I/O. Direct I/O instructions are allowed only in the local I/O mode.

Note that the next instruction will not be executed until the I/O operation is completed.

8.2.1 IN instruction

〔Format〕

IN [integer type variable
integer type variable with index
constant]

The effective parameters are 〈module type〉, 〈rack number〉, 〈slot number〉, and 〈offset〉, corresponding to each digit of a four-digit hexadecimal.

Each digit is subject to the following restrictions:

〈module type〉 = 0 for a discrete input module
= 1 for a register input module
1 ≦ 〈rack number〉 ≦ 5
1 ≦ 〈slot number〉 ≦ 6 where 〈rack number〉 = 1
≦ 9 where 2 ≦ 〈rack number〉 ≦ 5
0 ≦ 〈offset〉 ≦ {(total number of input words of input module considered)−1} ×2

〔Explanation〕 One word of input data is entered from an input module specified by the rack number, slot number, and offset. (The input data are held in the A register.) If the input operation is correct, the B register is set to OFF. If incorrect, the B register is set to OFF. If incorrect, the B register is set to ON.

〔Example〕 The following sample program enters the upper 16 bits of a 32-bit discrete input module mounted in slot 6 of rack number 3. The input data are stored in MW00100 only when they are input normally.

```
IN H0362
IFOFF
                    => MW00100
IEND
```

8.2.2 OUT instruction

〔Format〕

OUT [integer type variable
integer type variable with index
constant]

The effective parameters are 〈module type〉, 〈rack number〉, 〈slot number〉, and 〈offset〉, corresponding to each digit of at four-digit hexadecimal.

Each digit is subject to the following restrictions:

〈module type〉 = 0 for a discrete output module
= 1 for a register output module

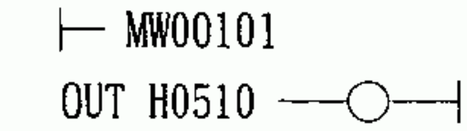
1 ≦ 〈rack number〉 ≦ 5

1 ≦ 〈slot number〉 ≦ 6 where 〈rack number〉 = 1
≦ 9 where 2 ≦ 〈rack number〉 ≦ 5

0 ≦ 〈offset〉 ≦ {(total number of output words of output module considered) − 1} ×2

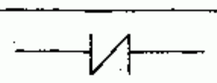
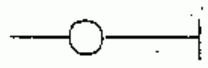
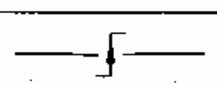
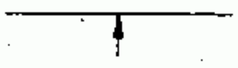
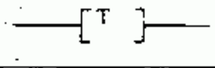
〔Explanation〕 Data held in the A register are output to an offset-specified word of an output module specified by the rack number and slot number. If the output operation is correct, the B register is set to OFF. If incorrect, the B register is set to ON.

〔Example〕 The following sample program outputs data stored in MW00101 to a 16-bit discrete output module in slot number 1 of rack number 5. Whether or not the data have been output normally is set in the coil.

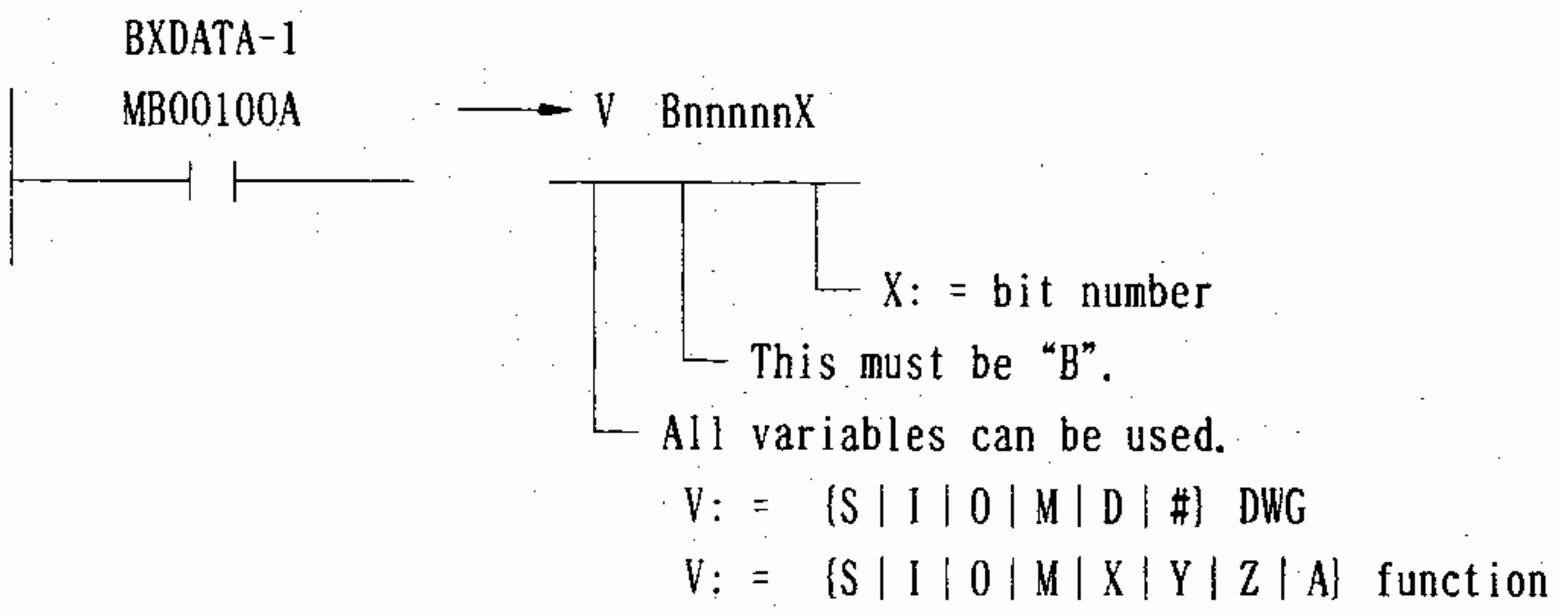


8.3 SEQUENCE CIRCUIT INSTRUCTIONS

Combine the following elements to create a sequence circuit:

No.	Sequence circuit element	Symbol	Remarks
1	Normally open contact (contact A)		Connection indicating elements
2	Normally closed contact (contact B)		(1) Branch 
3	Coil		
4	Rising pulse		(2) Parallel connecting point 
5	Falling pulse		
6	ON delay timer		(3) Parallel connection 
7	OFF delay timer		

Specify a relay number as follows:



8.3.1 Normally open contact instruction (contact A)

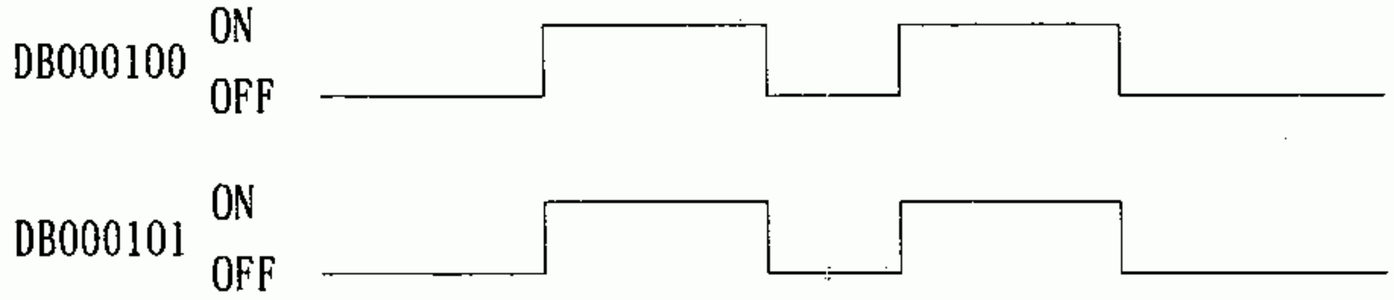
〔Format〕

bit type variable
bit type variable with index

〔Explanation〕 The normally open contact instruction sets the B register status to ON when the reference variabale value is 1 (ON).

Conversely, if the reference variable value is 0 (OFF), the instruction sets the B register status to OFF.

〔Example〕 In the following sample program, when DB000100 is ON, DB000101 goes ON.



8.3.2 Normally closed contact instruction (contact B)

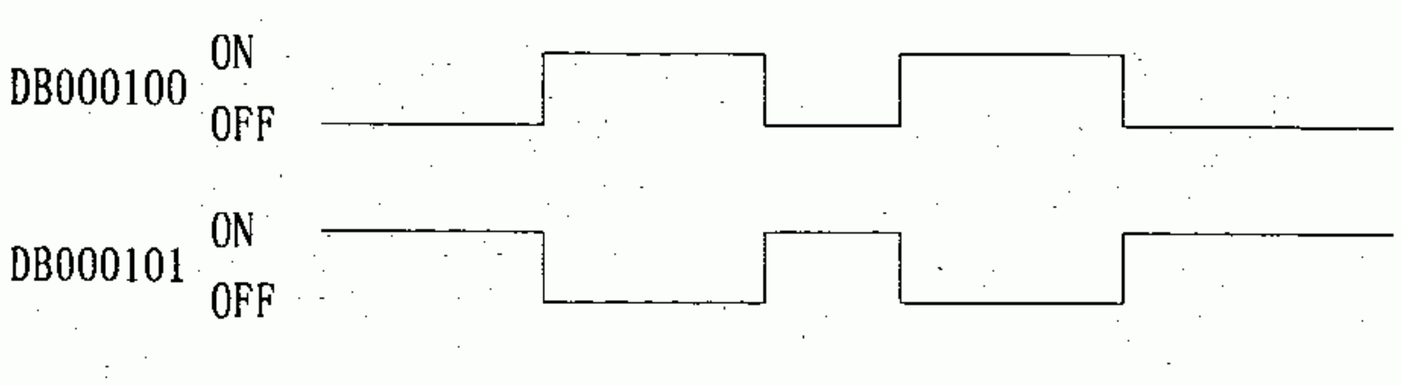
〔Format〕

bit type variable
bit type variable with index

〔Explanation〕 The normally closed contact instruction sets the B register status to OFF when the reference variable value is 1 (ON).

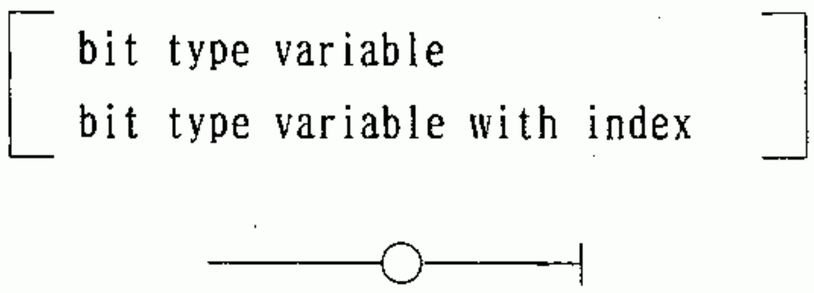
Conversely, if the reference variable value is 0 (OFF), the instruction sets the B register status to ON.

〔Example〕 In the following sample program, when DB000100 is ON, DB000101 goes OFF.



8.3.3 Coil instruction

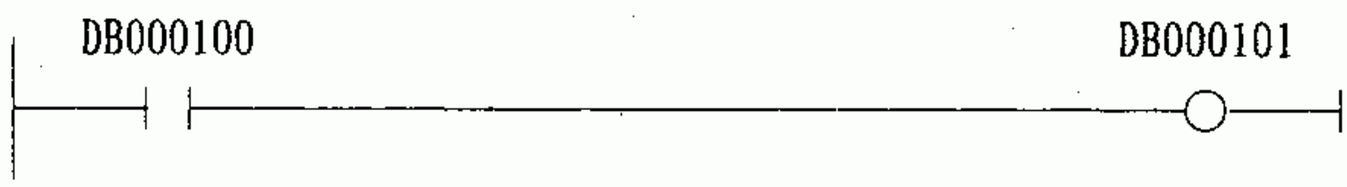
〔Format〕

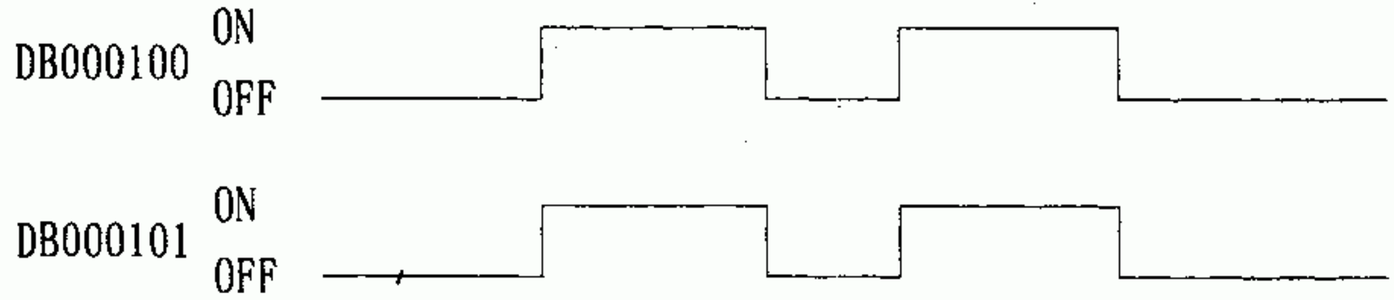


〔Explanation〕 The coil instruction sets the reference variable value to 1 (ON) when the preceding B register status is ON.

Conversely, when the preceding B register status is OFF, the instruction sets the reference variable value to 0 (OFF).

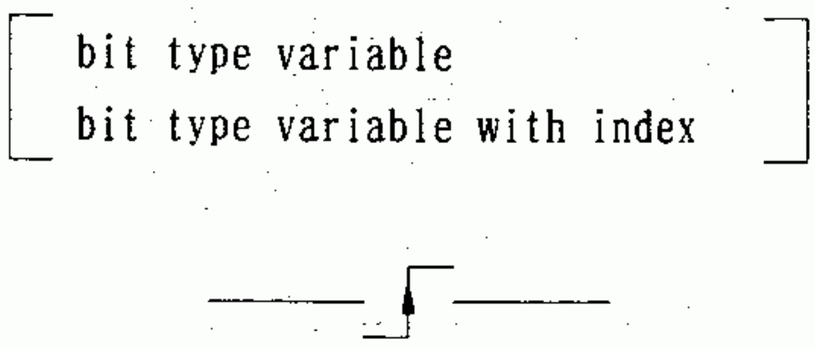
〔Example〕 In the following sample program, when DB000100 is ON, DB000101 goes ON.





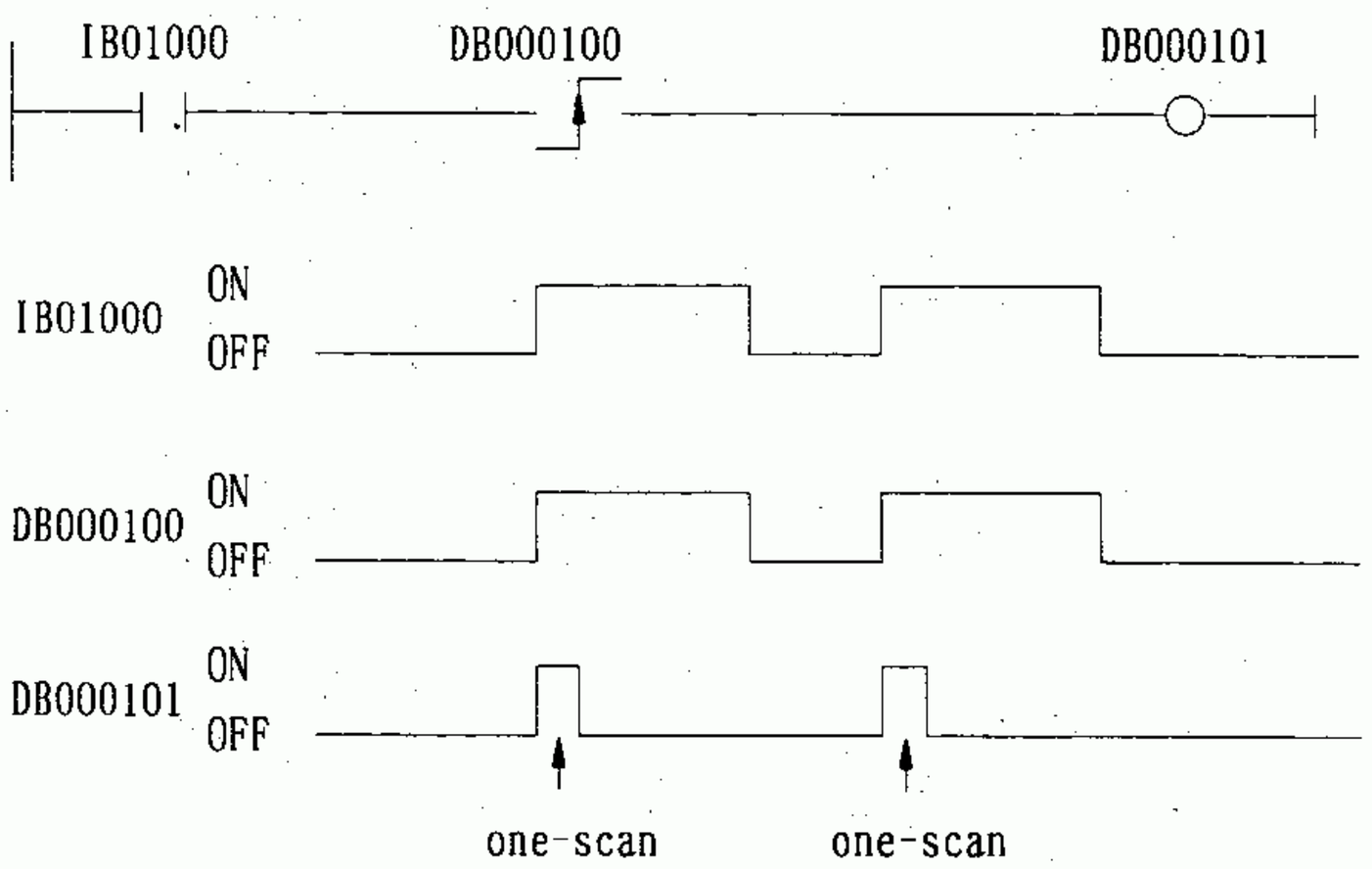
8.3.4 Rising pulse instruction

〔Format〕



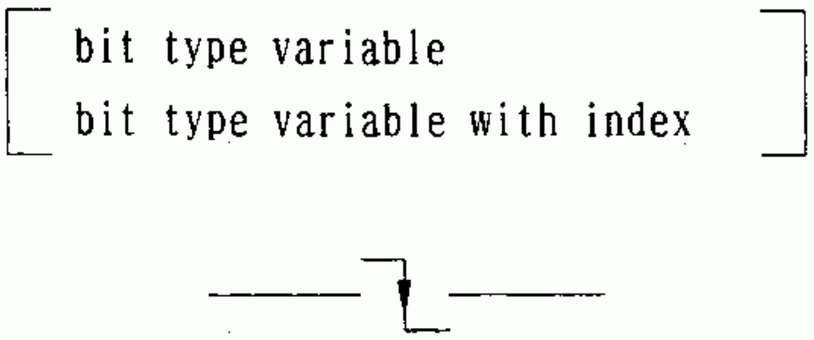
〔Explanation〕 The rising pulse instruction causes the B register status to be ON during one scan when the preceding B regaister status (IB01000) changes from OFF to ON. The previous B register value is retained in the reference variable.

〔Example〕 In the following sample program, when DB01000 is ON, DB000101 goes ON during one scan.



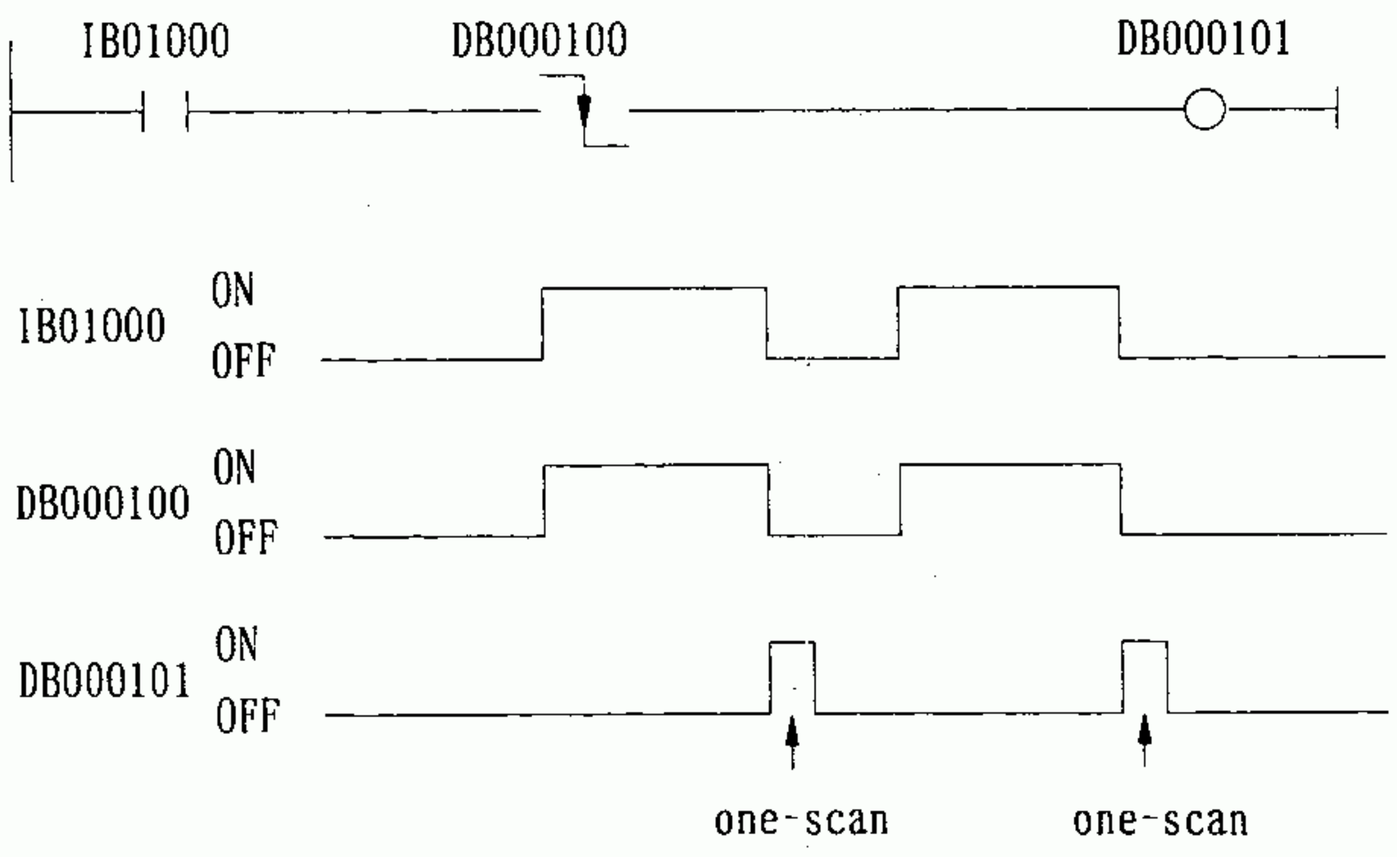
8.3.5 Falling pulse instruction

〔Format〕



〔Explanation〕 The falling pulse instruction causes the B register status to be ON during one scan when the preceding B register status (IB01000) changes from ON to OFF. The previous B register value is retained in the refeareance variable.

〔Example〕 In the following sample program, when IB01000 is OFF, DB000101 goes ON during one scan.



8.3.6 ON delay timer instruction

〔Format〕 ——{T preset value counted value }——

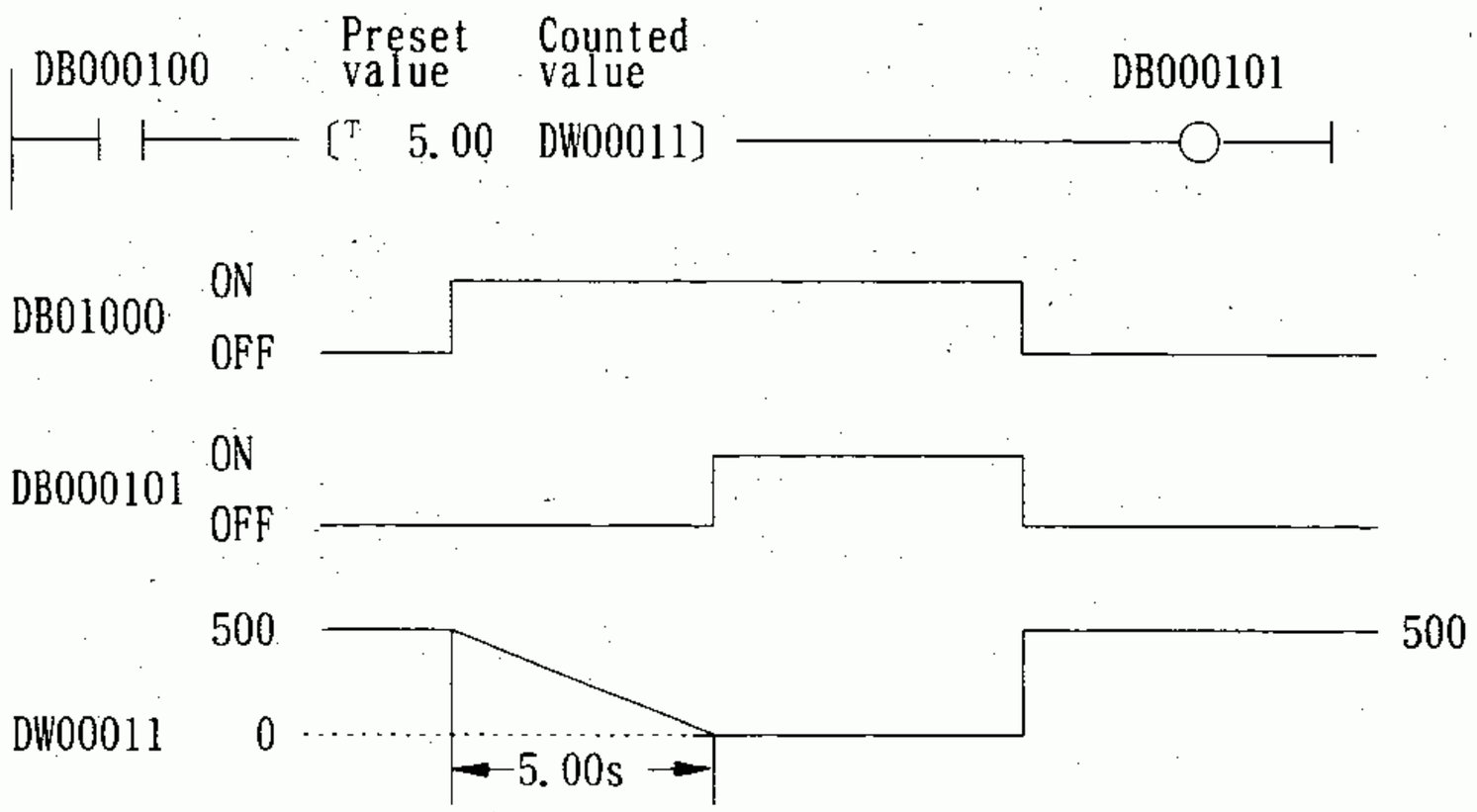
preset value : constant, integer type variable, integer type variable with index (0 to 655.35 s)

counted value: integer type variable (except #), integer type variable (except #) with index

〔Explanation〕 The ON delay timer counts the time when the preceding B register status is ON.

The B register status goes ON when the counted value = 0.

〔Example〕

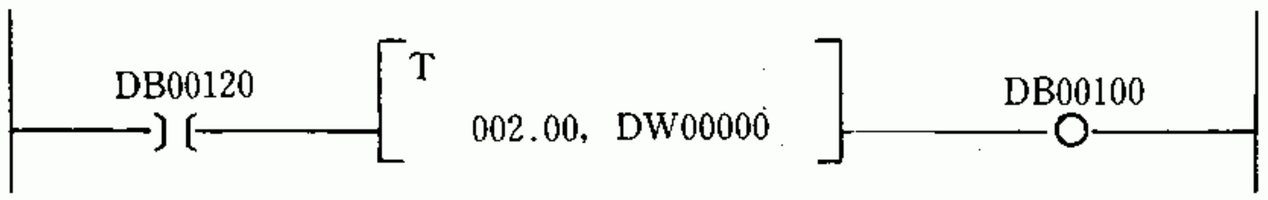


Precaution for using CP-3300 ON delay timer instruction

CPU (GL60V/VA) for single CPU system and CPU (GL60V1/V2) for dual CPU system differ in counting operation.

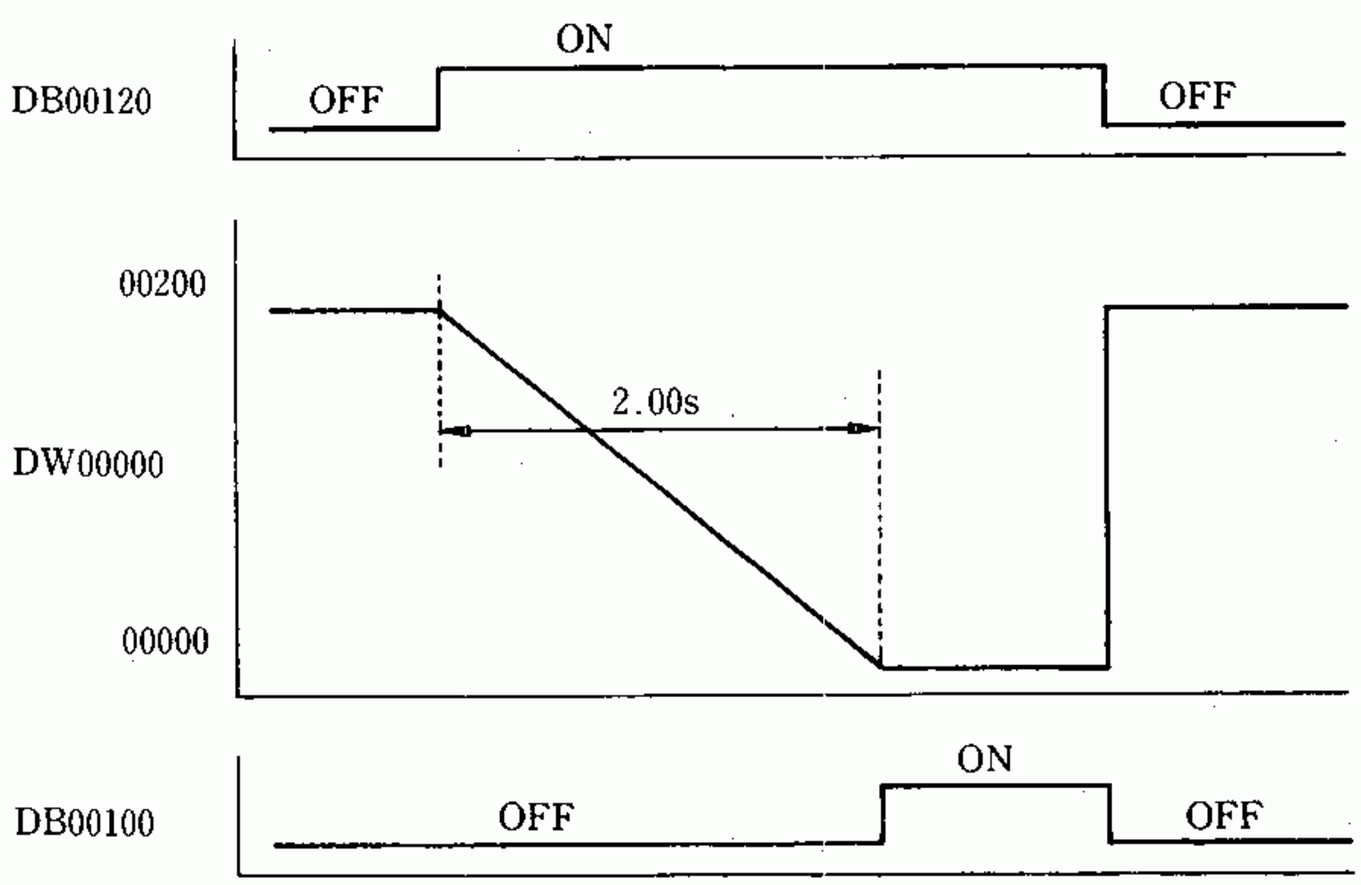
ON delay timer counting operation shows below.

〔Programming〕

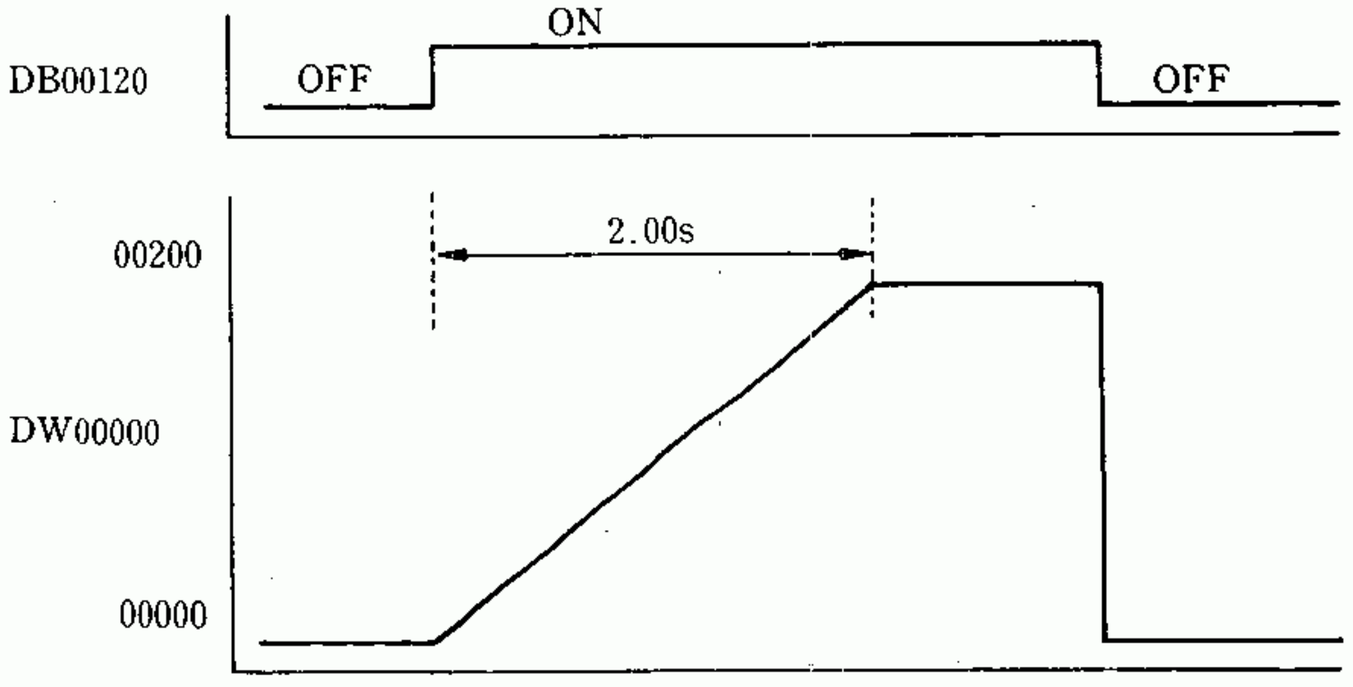


〔Counting operation〕

CPU(GL60V/VA) for single system



CPU(GL60V1/V2) for dual system



When creating the program that managing the time at CP-3300 start-up, be careful difference for setting of the counting register initial value.

- Single CPU : counting register = preset value
- Dual CPU : counting register = 00000

8.3.7 OFF delay timer instruction

〔Format〕 ——{ preset value counted value T }——

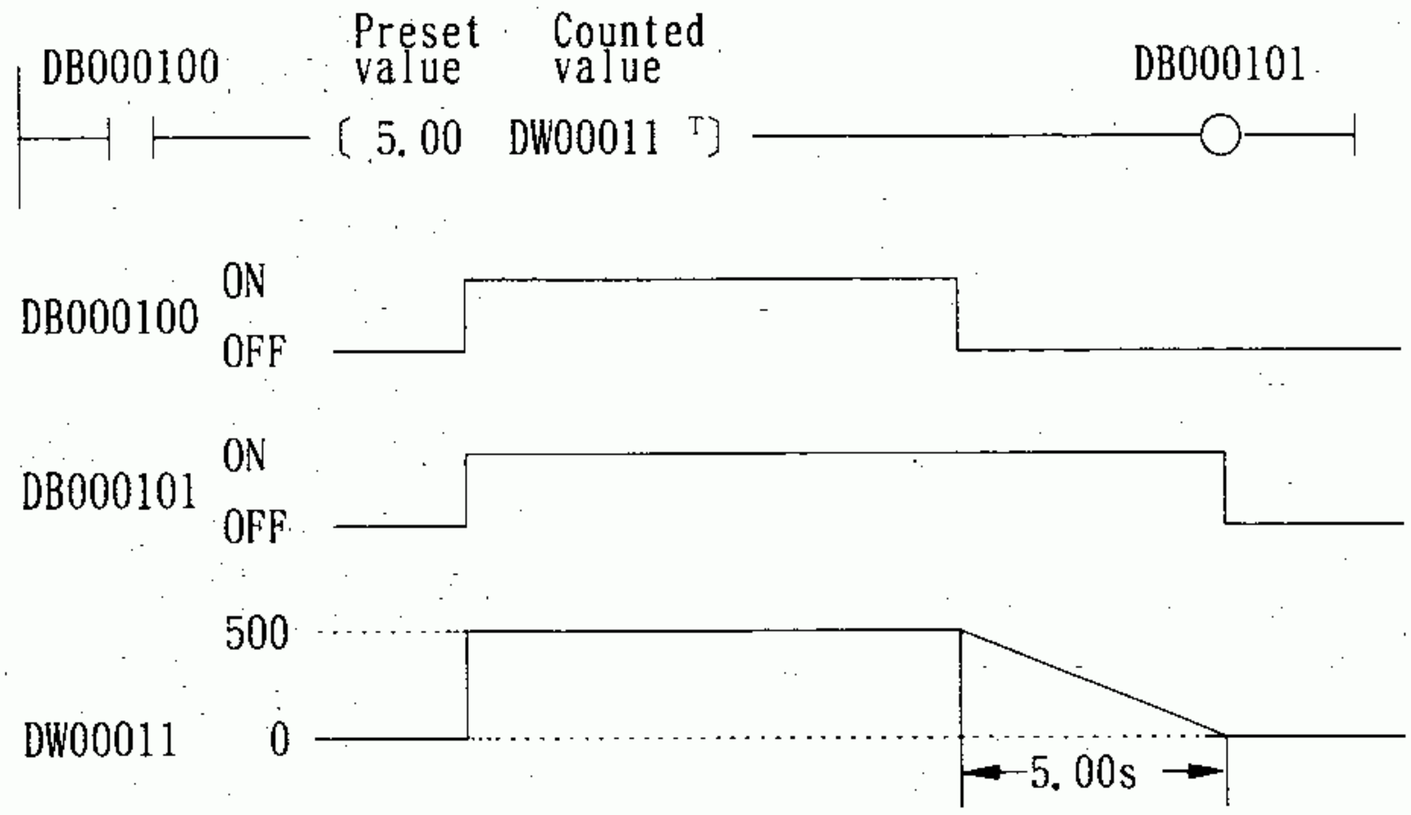
preset value : constant, integer type variable, integer type variable with index (0 to 655.35 s)

counted value: integer type variable (except #), integer type variable (except #) with index

〔Explanation〕 The OFF delay timer counts the time when the preceding B register status is OFF.

The B register status goes OFF when the counted value = 0.

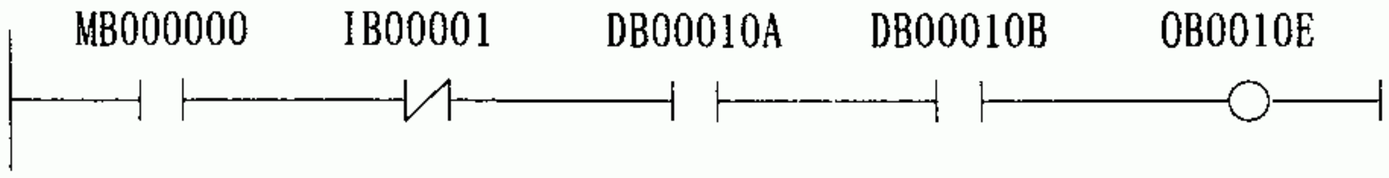
〔Example〕



8.3.8 Sequence circuit combination examples

(1) Serial circuit example

In the following example, relays are connected in series and their logical product is output to a coil.



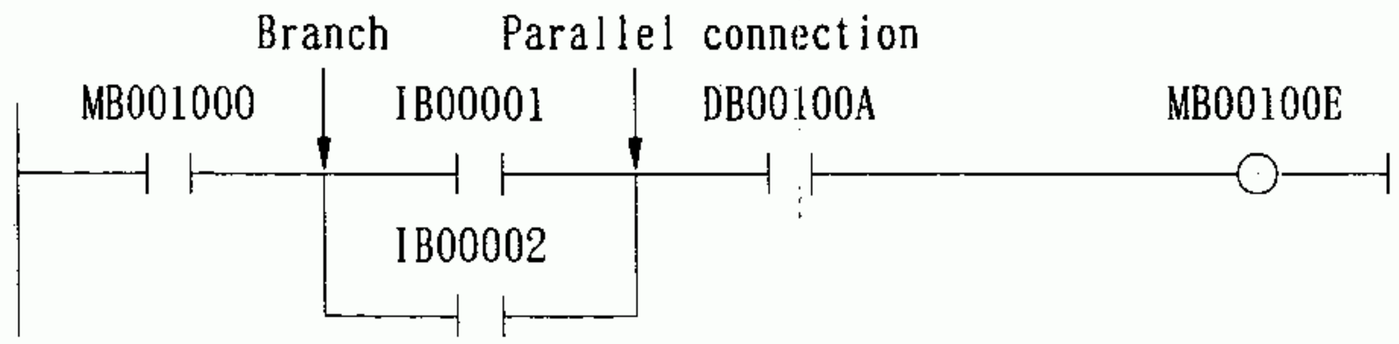
(2) Branch and parallel circuit examples

A branch indicating element is used to branch out the content of B register to multiple relays.

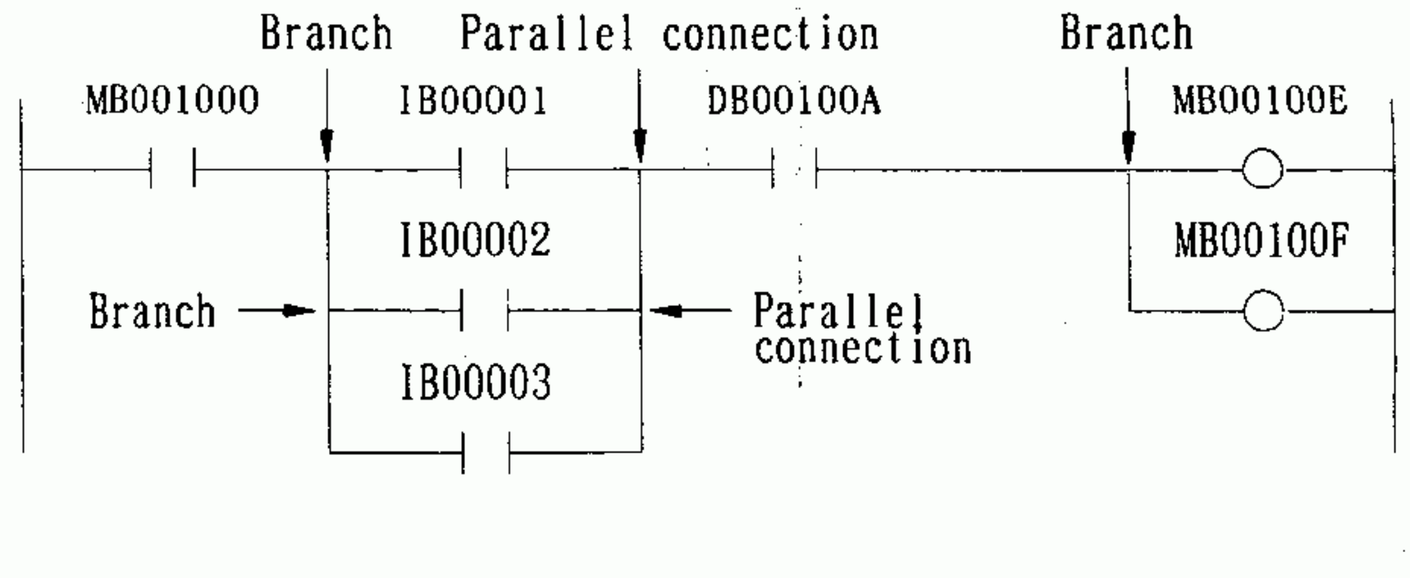
A parallel connection indicating element is used to obtain the logical sum of multiple relays.

In the following example, the result of serial and parallel connections of relays is output to a coil.

〔Example 1〕 Simple example of branch and paralled connection

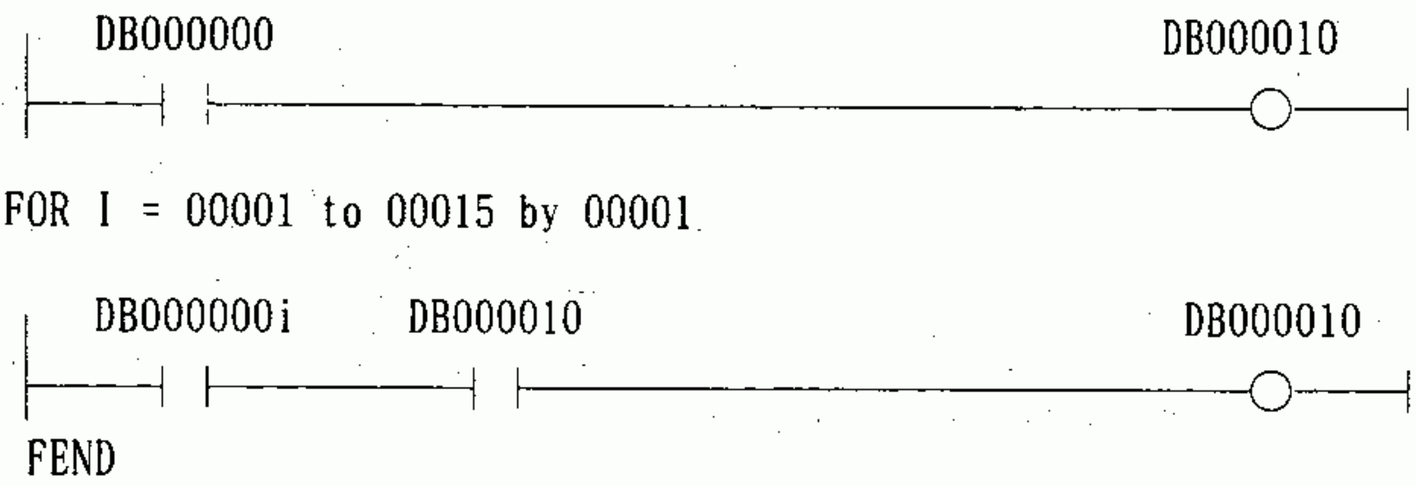


〔Example 2〕 Plural example of branch and paralled connection



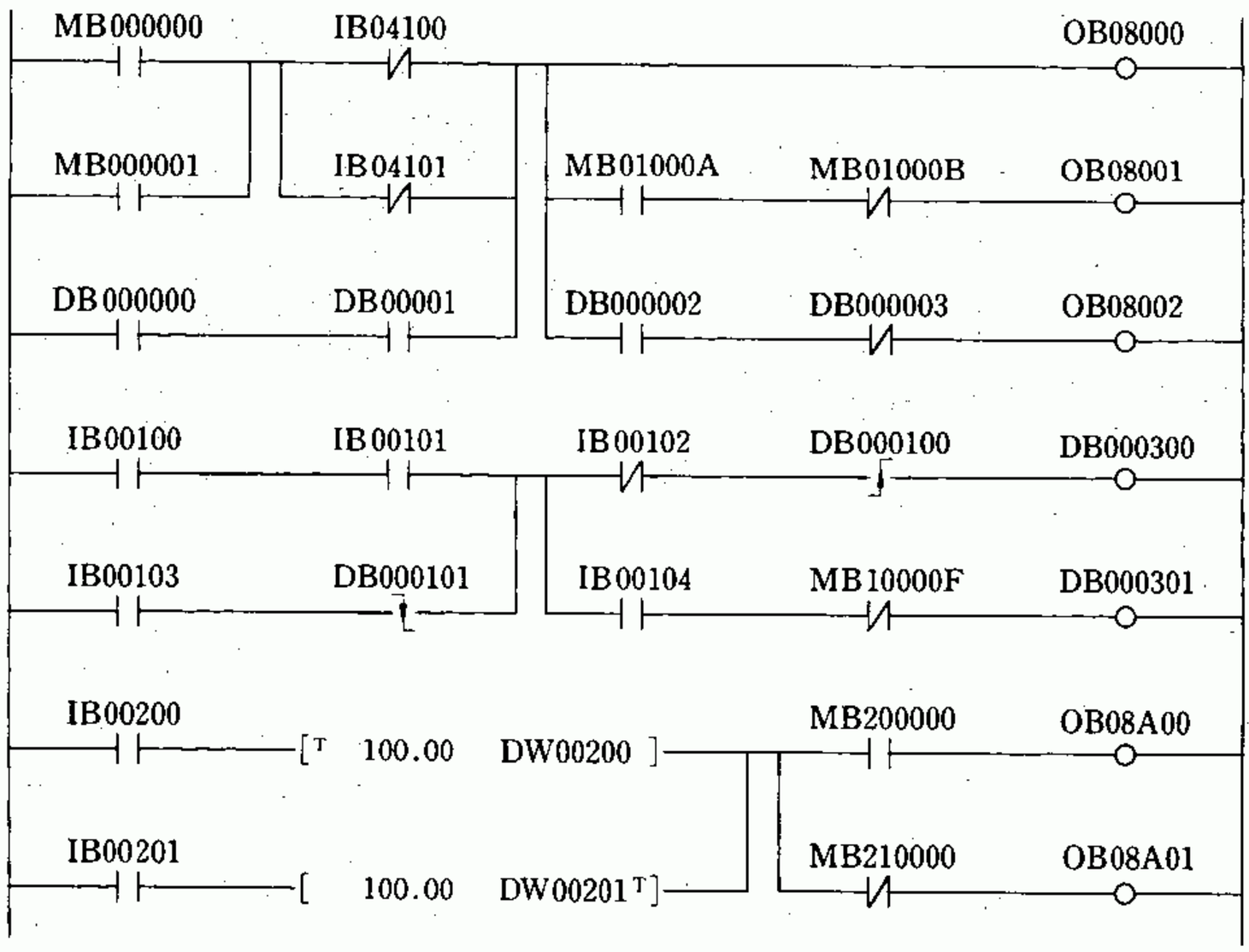
(3) Subscripted sequence circuit example

The relay numbers can be subscripted. In the following example, the logical product of relays DB000000 to DB000000F are obtained in DB000010.



(4) Complex sequence circuit example

A complex sequence circuit such as the one shown below can be created by a combination of sequence circuit elements and connection indicating elements.



8.4 NUMERICAL OPERATION INSTRUCTIONS

8.4.1 Load instructions

The CP-3300 can accommodate two kinds of data types: integer and real type. Accordingly, it provides two types of load instructions which begin an arithmetic expression.

(1) Integer type load instruction

〔Format〕

```
├─ [ integer type variable
   | integer type variable with index
   | index register
   | constant ]
```

〔Explanation〕 The integer type load instruction loads data to the A register and begins an integer type operation. Once the operation is started with an integer type load instruction, real type data cannot be specified until the next real type load instruction is entered.

〔Example〕
```
├─ MW00100  (The content of MW00100 is loaded to register A.)
├─ MF00100  (This use is not allowed.)
```

(2) Real type load instruction

〔Format〕

```
|├─ [ integer type variable
   | integer type variable with index
   | real type variable
   | real type variable with index
   | index register
   | constant ]
```

〔Explanation〕 The real type load instruction loads data in the F register and begins a real type operation. A program consisting of a series of operations that begin with a real type load instruction may use integer and real type variables. If an integer type variable is specified in a real type load instruction, the type is converted to real at the time execution.

〔Example〕
```
|├─ MF00200  (The content of MF00200 is loaded in register F.)
|├─ MW00100  (Integer type data in MW00100 are converted to real
              type and loaded in register F.)
```

8.4.2 Store instructions

〔Format〕

=> [integer type variable (except I, #)
integer type variable (except I, #) with index
real type variable (except I, #)
real type variable (except I, #) with index
index register]

〔Explanation〕 The store instruction stores the contents of the A or F register into a specified variable.

In the arithmetic expression beginning with an integer type load instruction, the contents of register A are stored.

In the arithmetic expression beginning with a real type load instruction, the contents of register F are stored.

In this case, if a specified variable is of real type, register F is stored as real type; and if a specified variable is of integer type, register F is converted (with a fraction rounded off) to integer type before being stored. Storage into a real type variable is possible only in an arithmetic expression beginning with a real type load instruction. Storage into a index register is possible only with an arithmetic expression beginning with an integer type load instruction.

〔Examples〕 (1) The content of register A is stored into MW00100.

```
    ├─  12345                                  => MW00100
                                                  (12345)
```

(2) The content of register F is stored into MF00200 as real type.

```
    ||─ 1.23456                                => MF00200
                                                  (1.23456)
```

(3) The content of register F is converted to real type and stored into MW00100.

```
    ||─ 1.23456                                => MW00100
                                                  (00001)
```

(4) The following uses are not allowed.

```
    ├─  12345                                  => MF00200

    ||─ 1.23456                                => I
```

8.4.3 Add instruction

〔Format〕

```
+ ┌ integer type variable
  │ integer type variable with index
  │ real type variable
  │ real type variable with index
  │ index register
  └ constant
```

〔Explanation〕 The add instruction adds integer or real type numeric values.

If the result of integer type addition is greater than 32767, an overflow operation error occurs.

No operation error occurs with a real type add instruction.

〔Examples〕 (1) Adding an integer type numerical value

```
├─  MW00100 + 12345          => MW00100
├─  MW00101 + MW00102        => MW00101
```

(2) Adding a real type numerical value

```
│├─ MF00200 + 1.23456         => MF00200
│├─ MF00202 + MF00204         => MF00202
```

8.4.4 Subtract instruction

〔Format〕

— [integer type variable
  integer type variable with index
  real type variable
  real type variable with index
  index register
  constant]

〔Explanation〕 The subtract instruction performs subtraction of an integer or real type numerical value.

If the result of integer type subtraction is less than -32767, an underflow operation error occurs.

No operation error occurs in a real type subtract instruction.

〔Examples〕 (1) Subtraction of integer type numerical values

```
├— MW00100 — 12345          => MW00100
├— MW00101 — MW00102        => MW00101
```

(2) Subtraction of real type numerical values

```
|├— MF00200 — 1.23456       => MF00200
|├— MF00202 — MF00204       => MF00202
```

8.4.5 Multiply instruction

〔Format〕

× [integer type variable
integer type variable with index
real type variable
real type variable with index
index register
constant]

〔Explanation〕 The multiply instruction performs multiplication of an integer or real type numerical value.

"×" can be used alone in a real type multiplication result, but "×" and "÷" must be paired in an integer type multiplication.

If the result of a real type multiplication is out of range of the F register value, an operation error occurs. (Letting X be a multiplication result, an exponent underflow occurs if $|X| < 3.4E-4932$ and an exponent overflow if $|X| > 1.2E+4932$.)

No operation error occurs in an integer type multiply instruction.

〔Examples〕 (1) Multiplication of an integer type numerical value

```
├─  MW00100 ×      3 ÷ 1          => MW00100
├─  MW00101 × MW00102 ÷ 1          => MW00101
```

(2) Multiplication of a real type numerical value

```
|├─  MF00200 ×    3.0              => MF00200
|├─  MF00202 × MF00204              => MF00202
```

8.4.6 Divide instruction

〔Format〕

÷ [integer type variable
  integer type variable with index
  real type variable
  real type variable with index
  index register
  constant]

〔Explanation〕 The divide instruction performs division of an integer or real type numerical value.

"÷" can be used alone in a real type division, but "÷" and "×" must be paired in an integer type division. If the value of specified variable is 0, a division-by-zero error occurs.

If the result of an integer type division is out of range of the A register value, an operation error occurs. (Letting X be the result of division, an exponent underflow occurs if X< -32768 and an overflow if X> 32767.)

If the result of a real type division is out of range of the F register value, an operation error occurs. (Letting X be a division result, an exponent underflow occurs if |X| < 3.4E-4932 and an exponent overflow if |X| > 1.2E+4932.)

〔Examples〕 (1) Division of an integer type numerical value

```
├─ MW00100 × 1 ÷ 3                    => MW00100
├─ MW00101 × 1 ÷ MW00101              => MW00101
```

(2) Division of a real type numerical value

```
||─ MF00200 ×     3.0                => MF00200
||─ MF00202 × MF00204                 => MF00202
```

8.4.7 MOD instruction

〔Format〕 MOD

〔Explanation〕 The MOD instruction outputs the remainder of an integer type division to register A.

〔Example〕 Find the quotient of an integer type division in MW00101 and the remainder in MW00102.

```
|—  MW00100 × 1  ÷  3                    => MW00101
    (00010)                                (00003)

    MOD                                  => MW00102
                                           (00001)
```

8.4.8 REM instruction

〔Format〕

REM [real type variable
      real type variable with index
      constant]

〔Explanation〕 The REM instruction outputs the remainder of a real type division to register F.

The remainder in this case refers to the residue obtained by subtracting a specified variable value from the F register repeatedly. In other words, suppose the F register value is A, the specified variable value X, the iterative subtraction count n, and the REM instruction output value Y. Then the following equation is established :

$$Y = A - (X * n) \qquad (0 \leqq Y < X)$$

〔Example〕 Find in MF00202 the remainder of division of a real type variable MF00200 by constant 1.5.

```
||—  MF00200 REM  1.5                  => MF00202
     (4.0)                               (1.0)
```

8.4.9 INC instruction

〔Format〕

INC [integer type variable (except I, #)
integer type variable (except I, #) with index
index register]

〔Explanation〕 The INC instruction adds 1 to a specified integer type variable. This instruction is not allowed in a real type operation.

No overflow operation error occurs even if the result of addition exceeds 32767.

Decimal : 0 → 1····32767 → -32768···· -1 → 0
Hexadecimal : 0000 → 0001···· 7FFF → 8000····FFFF → 0000

〔Examples〕 (1) Programs (a) and (b) below are equivalent :

(a) ⊢ MW00100 + 1 => MW00100
(b) INC MW00100

(2) The following uses are not allowed :

(a) INC IW0100 (I variable)
(b) INC #W00001 (# variable)
(c) INC MF00200 (real type variable)

8.4.10 DEC instruction

〔Format〕

DEC [integer type variable (except I, #)
integer type variable (except I, #) with index
index register]

〔Explanation〕 The DEC instruction subtracts 1 from a specified integer type variable. This instruction is not allowed in a real type operation.

No underflow operation error occurs even if the result of subtraction is less than -32768.

Decimal : 0 → -1·····-32768 → 32767···· 1 → 0

Hexadecimal : 0000 → FFFF···· 8000 → 7FFF····0001 → 0000

〔Examples〕 (1) Programs (a) and (b) below are equivalent :

(a) ⊢ MW00100 − 1 => MW00100

(b) DEC MW00100

(2) The following uses are not allowed

(a) DEC IW0100 (I variable)

(b) DEC #W00001 (# variable)

(c) DEC MF00200 (real type variable)

8.4.11 Extended add instruction

〔Format〕

```
++ [ integer type variable
     integer type variable with index
     index register
     constant ]
```

〔Explanation〕 The extended add instruction performs addition of an integer type numerical value. This instruction is not allowed in a real type operation.

No operation error occurs even when the operation result causes an overflow or underflow.

```
Decimal     :    0 → 1···32767 → -32768···· -1 →   0
Hexadecimal : 0000 → 0001···· 7FFF →  8000···FFFF → 0000
```

〔Example〕 Use this instruction to avoid an operation error as in the following addition of a hexadecimal :

```
├─ MW00100 ++ 2          => MW00100
   (H7FFF)                  (H8001)
```

8.4.12 Extended subtract instruction

〔Format〕

```
-- [ integer type variable
     integer type variable with index
     index register
     constant ]
```

〔Explanation〕 The exatended subtract instruction performs subtraction of an integer type numerical value. This instruction is not allowed in a real type operation.

No operation error occurs even when the operation result causes an overflow or underflow.

```
Decimal     :    0 → -1····-32767 → 32768····  1 →   0
Hexadecimal : 0000 →FFFF···· 8000 → 7FFF····0001 →0000
```

〔Example〕 Use this instruction to avoid an operation error as in the following subraction of a hexadecimal :

```
├─ MW00100 -- 2          => MW00100
   (H8000)                  (H7FFE)
```

8.5 LOGICAL OPERATION INSTRUCTIONS

The following logical operation instructions are used : logical product (AND ; ∧), logical sum (OR ; ∨), and exclusive logical sum (XOR ; ⊕).

8.5.1 Logical product instruction

〔Format〕

∧ [logical type variable
  logical type variable with index
  constant
  index register]

〔Explanation〕 Output the logical product of register A and a specified variable to register A.

This isntruction is not allowed in a real type operation. It is not combined with a NOT operation.

Logical product single-bit verification table (AND:A ∧ B=C)

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

〔Example〕 The logical product of MW00100 and a constant is stored in MW00101.

```
├─  MW00100 ∧ H00FF          => MW00101
    (H1234)   (H00FF)          (H0034)
```

8.5.2 Logical sum instruction

〔Format〕

$$\vee \left[\begin{array}{l} \text{logical type variable} \\ \text{logical type variable with index} \\ \text{constant} \\ \text{index register} \end{array}\right]$$

〔Explanation〕 Outputs the logical sum of register A and a specified variable to register A.

This isntruction is not allowed in a real type operation. It is not combined with a NOT operation.

Logical sum single-bit verification table
(OR : A ∨ B = C)

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

〔Example〕 The logical sum of MW00100 and a constant is stored in MW00101.

```
 ├─  MW00100  ∨  H00FF          => MW00101
      (H1234)    (H00FF)          (H12FF)
```

8.5.3 Exclusive logical sum instruction

〔Format〕

⊕ [logical type variable
  logical type variable with index
  constant
  index register]

〔Explanation〕 Output the logical sum of register A and a specified variable to register A.

This instruction is not allowed in a real type operation. It is not combined with a NOT operation.

Exclusive logical sum single-bit verification table
(XOR : A ⊕ B = C)

A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

〔Example〕 The exclusive logical sum of MW00100 and a constant is stored in MW00101.

```
    ⊢  MW00100 ⊕ H00FF          => MW00101
       (H5555)   (H00FF)          (H55AA)
```

8.6 COMPARE INSTRUCTIONS

Six types of compare instructions that compare numerical values and test an equivalent relationship are provided.

〔Format〕

<	integer type variable
≦	integer type variable with index
=	real type variable
≠	real type variable with index
≧	constant
>	index register

〔Explanation〕 The result of comparison is stored in register B.
(ON if verified.)

In an integer type operation, comparison with a real type variable is not allowed.

Note: To receive the result of comparison at a coil, the result is held and reflected on the next instruction.

〔Example〕 If the value of MW00100 is not 100, MB00010A is set to ON and the instructions from IFON are executed.

```
                                                    MB00010A
   ├─ MW00100  ≠ 00100                  ──────────○──────────┤
 ┌─ IFON
 │ ├─ MW00101  + MW00102  + MW00103   => MW00104
 │ ├─ MW00102
 │   :
 │   :
 └─ IEND
```

8.7 NUMERICAL CONVERSION INSTRUCTIONS

Six types of numerical value conversion instructions are provided that change the contents of register A or F, as listed below. In these instructions, register A (F) is used as input and the operation result remains in register A (F).

No.	Numerical conversion instruction	Register A	Register F	Explanation
1	Sign inversion INV	○	○	Inverts the sign of register A or F.
2	Complement of 1 COM	○	×	Determines, the complement of 1 of register A.
3	Absolute value conversion ABS	○	○	Determines the absolute value of register A or F.
4	BIN conversion BIN	○	×	Performs BIN conversion of register A.
5	BCD conversion BCD	○	×	Performs BCD conversion of register A.
6	Parity conversion PARITY	○	×	Computes the number of ON (=1) in the A register bits.

8.7.1 INV instruction

〔Format〕 INV

〔Explanation〕 Inverts the sign of register A or F.

The A register is the target of the instruction in an integer type operation and the F register in a real type operation.

〔Example〕

```
├─ MW00100   INV          => MW00101
   (00100)                   (-00100)

├┼─ MF00200   INV          => MF00202
   (1.0000)                  (-1.0000)
```

8.7.2 COM instruction

〔Format〕 COM

〔Explanation〕 Determines the complement of 1 of register A.

This instruction is allowed only in an integer type operation and not allowed in a real type operation.

〔Example〕

```
├─ MW00100   COM                  => MF00101
    (H 5555)                         (H AAAA)
```

8.7.3 ABS instruction

〔Format〕 ABS

〔Explanation〕 Determines the absolute value of register A or F.

The A reister is the target of the instruction in an integer type operation and the F register in a real type operation.

〔Example〕

```
├─ MW00100   ABS                  => MW00101
    (-00100)                         (00100)

├─ MF00200   ABS                  => MF00202
    (-1.0000)                        (1.0000)
```

8.7.4 BIN instruction

〔Format〕 BIN

〔Explanation〕 Converts a numeric value represented in BCD in the A register into a binary number (BIN conversion).

This instruction is allowed only in an integer type operation and not allowed in a real type operation.

Letting "abcd" be a numerical value (four digits) represented in BCD in the A register, the output value Y of BIN instruction is given by the following formula.

$$Y = (a * 1000) + (b * 100) + (c * 10) + d$$

If the value of register A is not BCD (such as H123F), the above formula is applicable, but a correct result will be not obtained.

〔Example〕

```
├─ MW00100   BIN                  => MW00101
    (H 1234)                         (D 01234)
```

8.7.5 BCD instruction

〔Format〕 BCD

〔Explanation〕 Converts a numerical value represented in binary in the A register into BCD (BCD conversion).

This instruction is allowed only in an integer type operation and not in a real type operation.

Letting "0abcd" be a numerical value (five digits) represented in decimal in register A, the value Y of output of BCD instruction is given by following formula.

$$Y = (a * 4096) + (b * 256) + (c * 16) + d$$

If a value in the A register cannot be represented in BCD (a number exceeding (D09999 or a negative number), the formula above is applicable, but a correct result is not obtained.

〔Example〕

```
├─ MW00100   BCD                  => MW00101
   (D 01234)                        (H 1234)
```


8.7.6 PARITY instruction

〔Format〕 PARITY

〔Explanation〕 Computes the number of ON (=1) in the binary-represented bits in register A.

This instruction is allowed only in an integer type operation and not in a real type operation.

〔Example〕

```
├─ MW00100   PARITY               => MW00101
   (H F0F0)                         (D 00008)
```

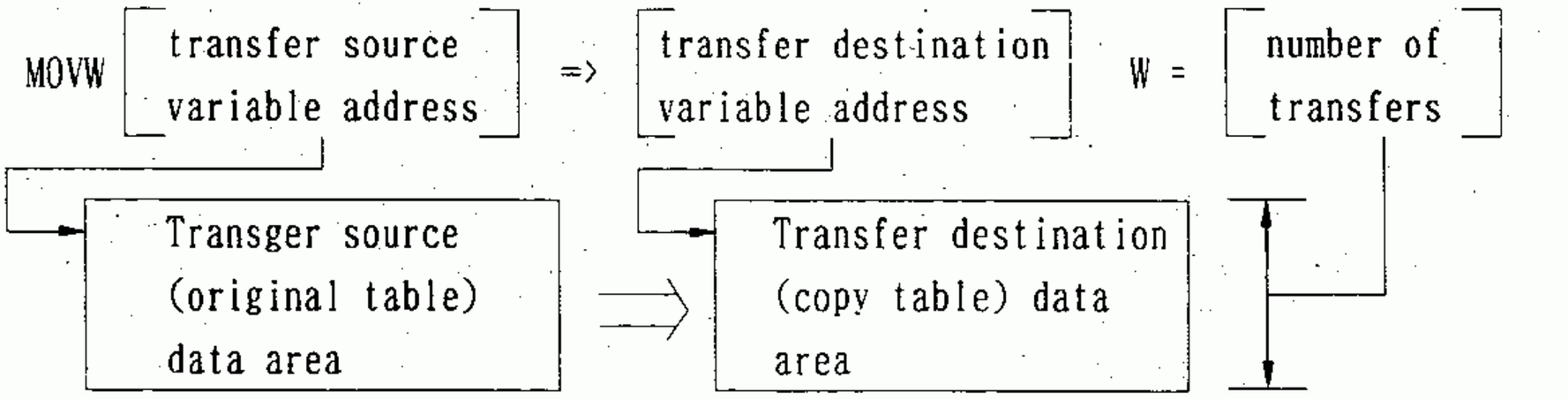
8.8 DATA TRANSFER INSTRUCTIONS

8.8.1 MOVW instruction

〔Format〕

	transfer source variable address		transfer destination variable address		〔number of transfers〕
MOVW	integer type variable integer type variable with index	->	integer type variable integer type variable with index	W =	integer type variable integer type variable with index constant

〔Explanation〕 The MOVW instruction creates a copy of a data table. The copy operation is performed one word at a time in the direction of increasing the register number. The original table is retained as long as the original table does not overlap the copy table. Care must be taken to prevent overlapping because the original table will be lost if overlap occurs.



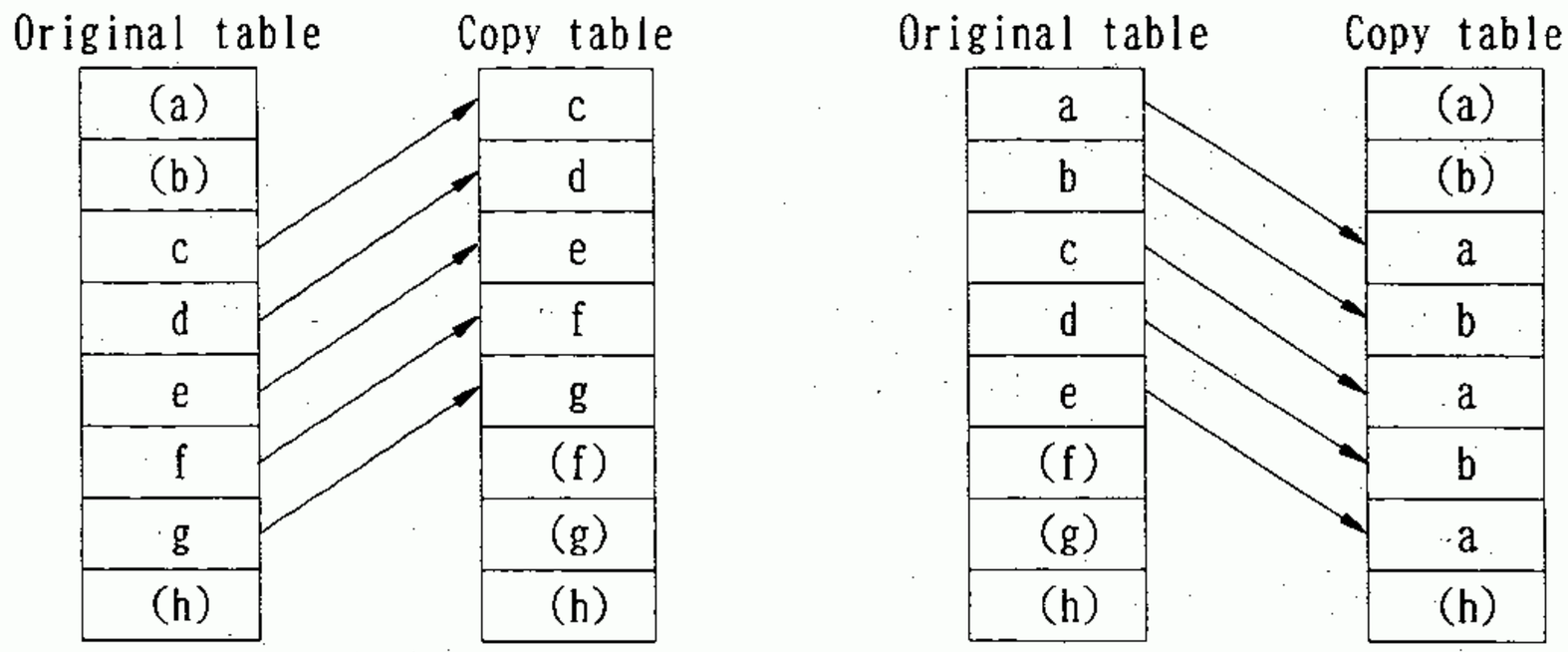


Table overlapping (1) Table overlapping (2)

〔Examples〕 (1) DW00000 to DW00009 are copied to MW00100 to MW00109.

```
MOVW   DW00000 -> MW00100   W = 00010
```

(2) MW00100 to MW00199 are turned to 0 by using an overlap.

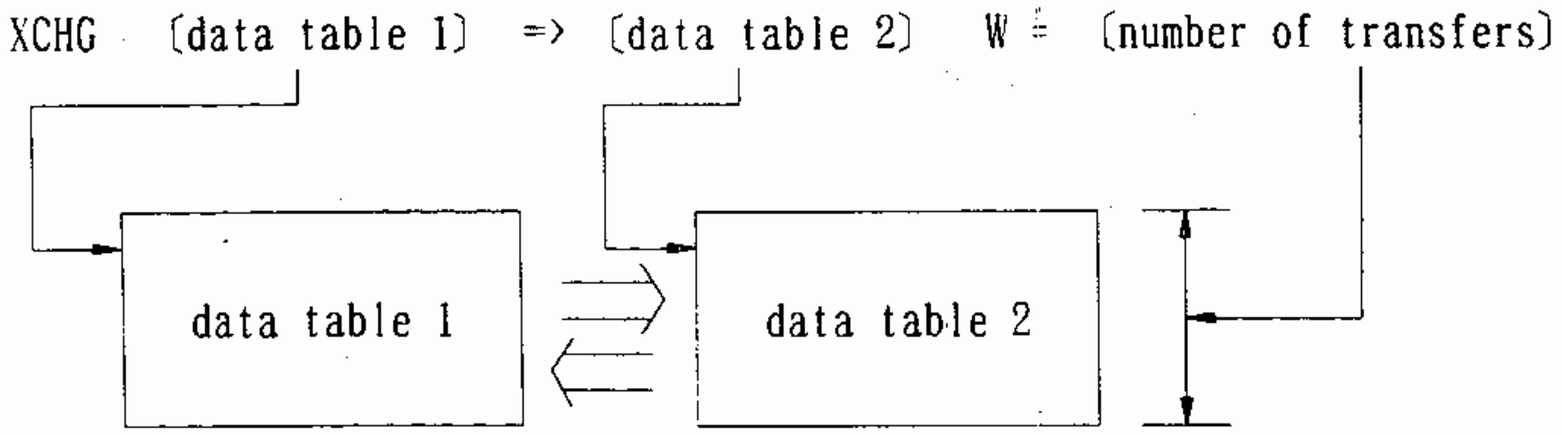
```
├─ 00000                                  => MW00100
   MOVW    MW00100 -> MW00101   W = 00099
```

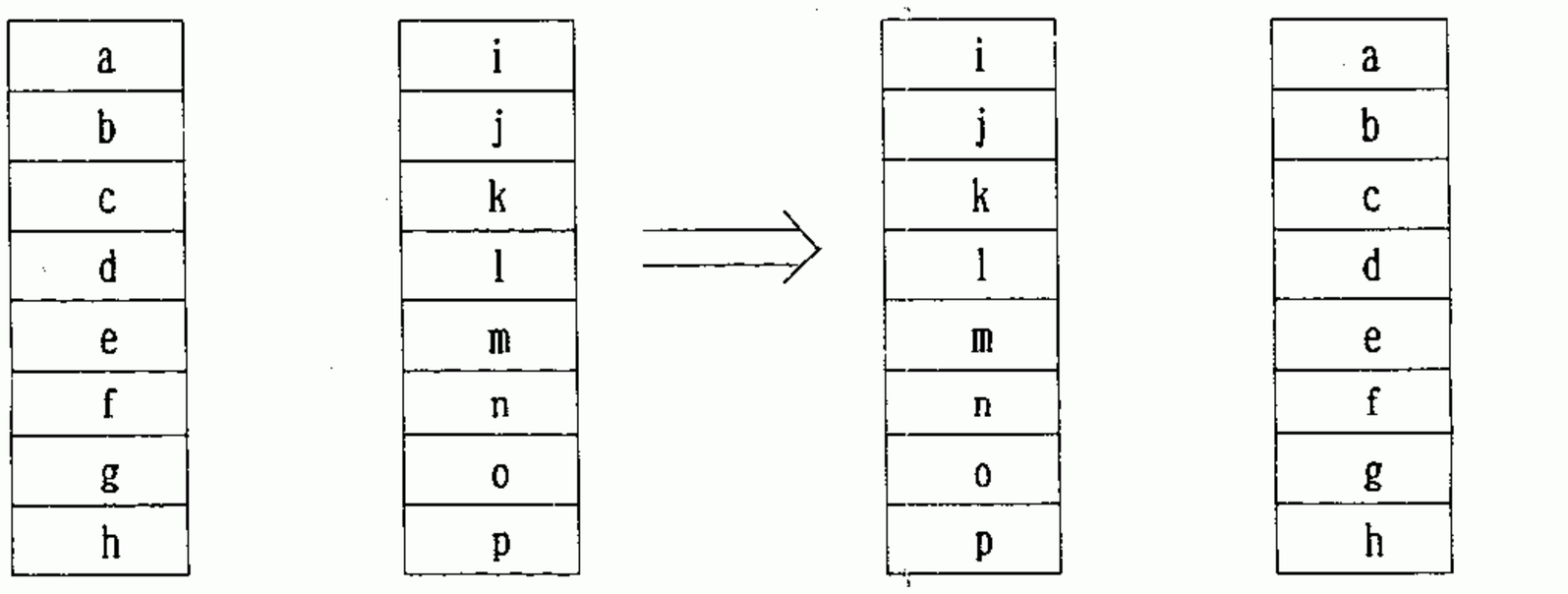
8.8.2 XCHG instruction

〔Format〕

	〔data table 1〕		〔data table 2〕		〔number of transfers〕
XCHG	integer type variable integer type variable with index	->	integer type variable integer type variable with index	W =	integer type variable integer type variable with index constant

〔Explanation〕 The XCHG instruction exchanges the data contents between data tables 1 and 2.





〔Example〕 The contents of DW00000 to DW00009 are exchanged with those of MW00100 to MW00109.

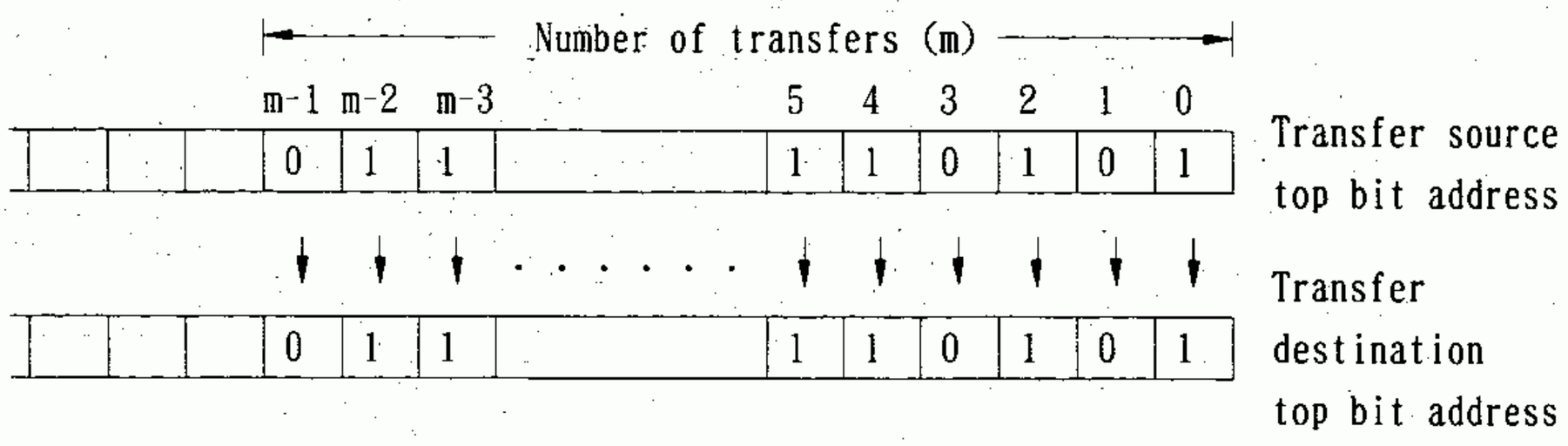
```
XCHG   DW00000 -> MW00100   W = 00010
```

8.8.3 MOVB instruction

〔Format〕

	transfer source bit address		transfer destination bit address		〔number of transfers〕
MOVB	bit type variable bit type variable with index	->	bit type variable bit type variable with index	W =	integer type variable integer type variable with index constant

〔Explanation〕 The MOVB instruction transfers a specified amount of bit data from the beginning of an original bit table to the beginning of a copy bit table. The transfer is performed one bit at a time in the direction of increasing the relay number. The original bit table is retained as long as there is no overlap between the original and copy bit table. Care must be taken to prevent an overlap.



〔Example〕 DB000008 to DB0000F are copied to MB001000 to MB001007.

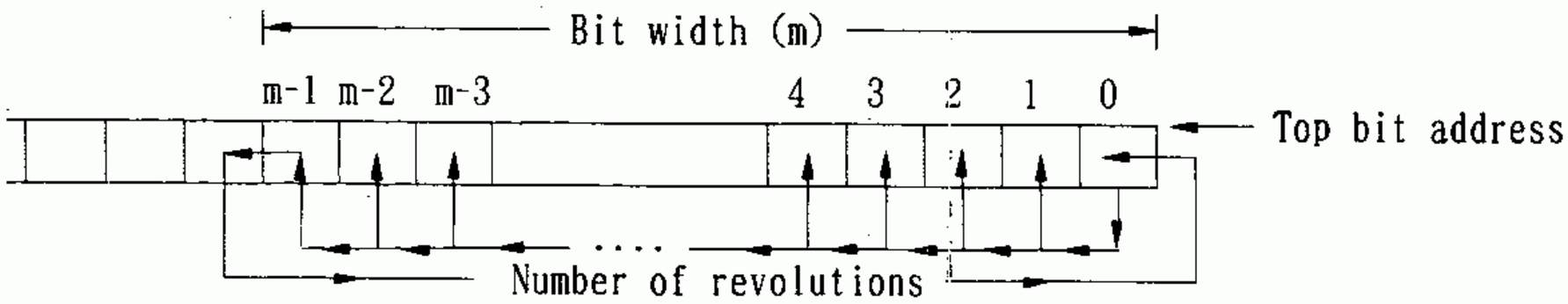
```
MOVB    DB000008  ->  MB001000    W = 00008
```

8.8.4 ROTL instruction

〔Format〕

	〔top bit address〕		〔number of revolutions〕		〔bit width〕
ROTL	integer type variable integer type variable with index	N =	integer type variable integer type variable with index	W =	integer type variable integer type variable with index constant

〔Explanation〕 The ROTL instruction rotates a bit table specified by a top bit address and bit width to the left a specified number of revolutions.



ROTL operation

〔Example〕 DB000000 to DB00000F are rotated one bit to the left.

ROTL DB000000 N = 00001 W = 00016

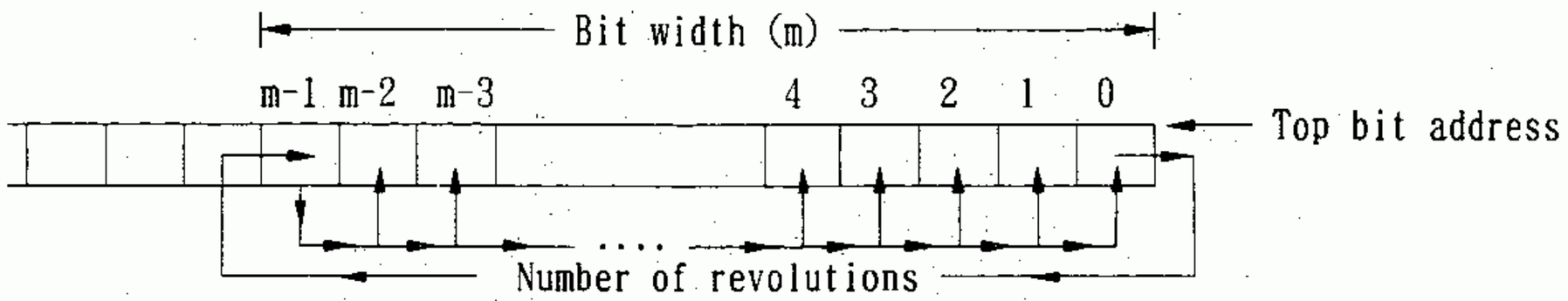
(Before executing ROTL)	→	(After executing ROTL)
DW00000 = H 0001		DW00000 = H 0002
= H 8000		= H 0001

8.8.5 ROTR instruction

〔Format〕

	〔top bit address〕		〔number of revolutions〕		〔bit width〕
ROTR	integer type variable integer type variable with index	N =	integer type variable integer type variable with index	W =	integer type variable integer type variable with index constant

〔Explanation〕 The ROTR instruction rotates a bit table specified by a top bit address and bit width to the right a specified number of revolutions.



ROTR operation

〔Example〕 DB000000 to DB0000F are rotated one bit to the right.

ROTL DB000000 N = 00001 W = 00016

(Before executing ROTR)	→	(After executing ROTR)
DW00000 = H 0001		DW00000 = H 8000
= H 8000		= H 4000

8.9 BASIC FUNCTION INSTRUCTIONS

8.9.1 SIN instruction

〔Format〕 SIN

〔Explanation〕 The SIN instruction takes the result of the preceding operation (F register) as an input (unit : degrees) and leaves its sine in the F register. It can be used only in a real type operation.

〔Example〕 The sine 〔SIN (θ) = 0.5 〕 of an input value (θ = 30.0) is computed.

θ		SIN(θ)
⊢ MF00200	SIN	=> MF00202
(30.000)		(0.5000)

8.9.2 COS instruction

〔Format〕 COS

〔Explanation〕 The COS instruction takes the result of the preceding operation (F register) as an input (unit : degrees) and leaves its cosine in the F register. It can be used only in a real type operation.

〔Example〕 The cosine 〔COS (θ) = 0.5 〕 of an input value (θ = 60.0) is computed.

θ		COS(θ)
⊢ MF00200	COS	=> MF00202
(60.000)		(0.5000)

8.9.3 TAN instruction

〔Format〕 TAN

〔Explanation〕 The TAN instruction takes the result of the preceding operation (F register) as an input (unit : degrees) and leaves its tangent in the F register. It can be used only in a real type operation.

〔Example〕 The tangent 〔TAN (θ) = 1.0〕 of an input value (θ = 45.0) is computed.

θ		TAN(θ)
⊢ MF00200	TAN	=> MF00202
(45.000)		(1.0000)

8.9.4 ASIN instruction

〔Format〕

〔denominator specified value(y)〕

ASIN [real type variable / real type variable with index / constant]

〔Explanation〕 The ASIN instruction takes the result of the preceding operation 〔F register : numerator (x)〕 and a denominator specified value (y) as inputs (x/y) and leaves its arcsine (unit : degrees) in the F register. It can be used only in a real type operation.

〔Example〕 The arcsine 〔30.0 = θ = ASIN (x/y)〕 of an input value (1.0/2.0) is computed.

	x		y		(θ)
⊢	MF00200	ASIN	MF00202	=>	MF00204
	(1.0000)		(2.0000)		(30.000)

8.9.5 ACOS instruction

〔Format〕

〔denominator specified value(y)〕

ACOS [real type variable / real type variable with index / constant]

〔Explanation〕 The ACOS instruction takes the result of the preceding operation 〔F register : numerator (x)〕 and a denominator specified value (y) as inputs (x/y) and leaves its arccosine (unit : degrees) in the F register. It can be used only in a real type operation.

〔Example〕 The arccosine 〔60.0 = θ = ACOS (x/y)〕 of an input value (1.0/2.0) is computed.

	x		y		
⊢	MF00200	ACOS	MF00202	=>	MF00204
	(1.0000)		(2.0000)		(60.000)

8.9.6 ATAN instruction

〔Format〕

〔denominator specified value(y)〕

ATAN [real type variable / real type variable with index / constant]

〔Explanation〕 The ATAN instruction takes the result of the preceding operation 〔F register : numerator (x)〕 and a denominator specified value (y) as inputs (x/y) and leaves its arctangent (unit : degrees) in the F register. It can be used only in a real type operation.

〔Example〕 The arctangent 〔45.0 = θ = ATAN (x/y)〕 of an input value (1.0/1.0) is computed.

	x		y		(θ)
||—	MF00200	ACOS	MF00202	=>	MF00204
	(1.0000)		(1.0000)		(45.000)

8.9.7 SQRT instruction

〔Format〕 SQRT

〔Explanation〕 The SQRT instruction takes the result of the preceding operation (F register) as an input (unit : degrees) and leaves its square root in the F register. If the input is a negative number, the square root of the absolute value is computed and the result is retained in the F register as a negative number. This instruction can be used only in a real type operation.

〔Examples〕 (1) When the input is a positive number

||—	MF00200	SQRT	=>	MF00202
	(64.000)			(8.0000)

(2) When the input is a negative number

||—	MF00200	SQRT	=>	MF00202
	(-64.000)			(-8.0000)

8.9.8 EXP instruction

〔Format〕 EXP

〔Explanation〕 The EXP instruction takes the result of the preceding operation (F register) as an input and leaves the natural logarithm base (e) raised to the power of the input as an operation result in the F register. It can be used only in a real type operation.

〔Example〕 With an input value (x=1.0), the "e" raised to the power of the input (e^x = 2.7183) is computed.

```
    ||—  MF00200   EXP      => MF00202
         (1.0000)              (2.7183)
```

8.9.9 LN instruction

〔Format〕 LN

〔Explanation〕 The LN takes the result of the preceding operation (F register) as an input and leaves its natural logarithm as an operation result in the F register. It can be used only in a real type operation.

〔Example〕 With an input value (x=10.0), its natural logarithm 〔Log e(x) = 2.3026〕 is computed.

```
    ||—  MF00200   LN       => MF00202
         (10.000)              (2.3026)
```

8.9.10 LOG instruction

〔Format〕 LOG

〔Explanation〕 The LOG takes the result of the preceding operation (F register) as an input and leaves its common logarithm as an operation result in the F register. It can be used only in a real type operation.

〔Example〕 With an input value (x=10.0), its common logarithm 〔Log 10(x) = 1.0〕 is computed.

```
    ||—  MF00200   LOG      => MF00202
         (10.000)              (1.0000)
```

8.10 DDC INSTRUCTIONS

8.10.1 DZA instruction

〔Format〕

〔dead zone specified value〕

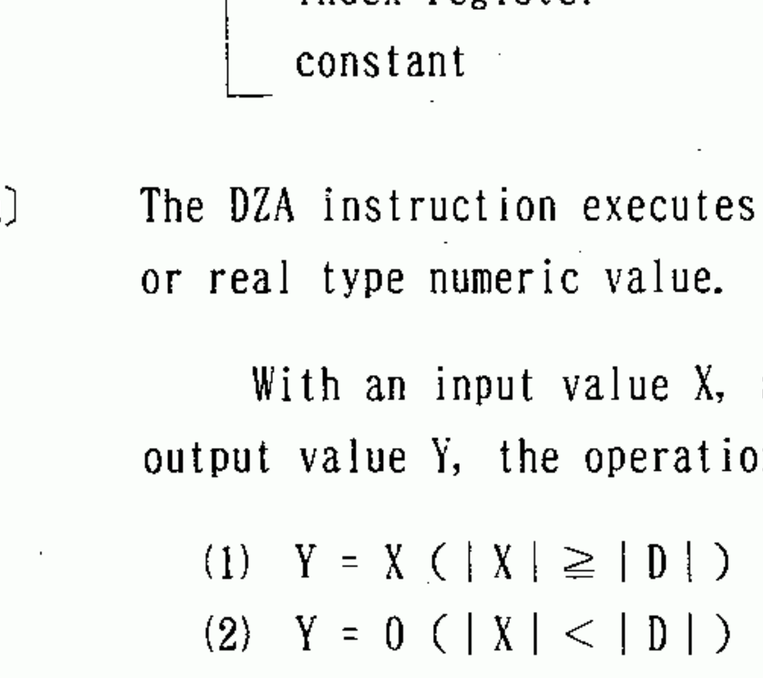
```
        ┌ integer type variable
        │ integer type variable with index
DZA     │ real type variable
        │ real type variable with index
        │ index register
        └ constant
```

〔Explanation〕 The DZA instruction executes a dead zone operation on an integer or real type numeric value.

With an input value X, a dead zone specified value D, and an output value Y, the operation is performed as follows :

(1) $Y = X \ (|X| \geqq |D|)$

(2) $Y = 0 \ (|X| < |D|)$



DZA instruction operation

〔Examples〕 (1) Integer type operation

```
          X              D
├─  MW00100  DZA   00100  =>  MW00101
    (00150)                  (00150) ←Outside dead zone area
    (00050)                  (00000) ←Inside dead zone area
```

(2) Real type operation

```
          X              D
├┼─  MF00200  DZA   100.00  =>  MF00202
    (150.00)                   (150 00) ←Outside dead zone area
    ( 50.00)                   (  0 00) ←Inside dead zone area
```

8.10.2 DZB instruction

〔Format〕

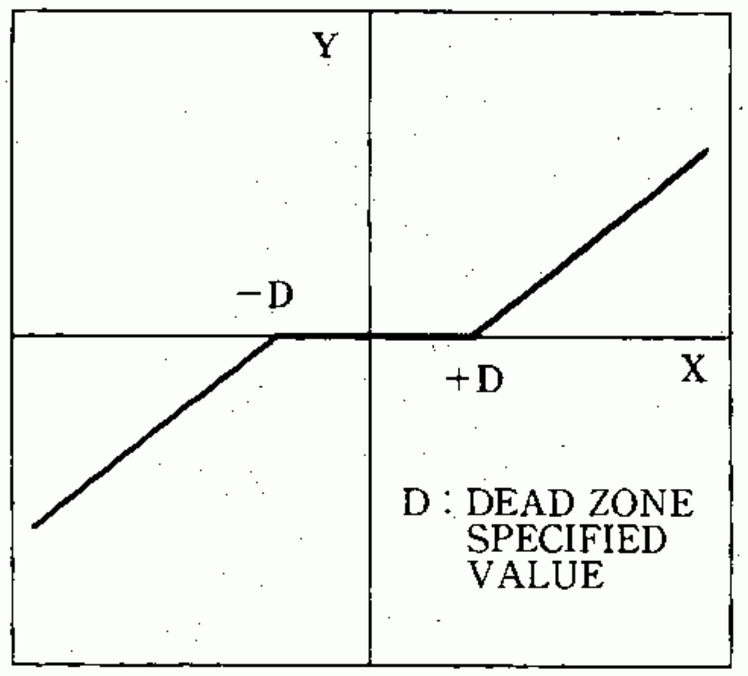
〔dead zone specified value〕

```
DZB [ integer type variable
      integer type variable with index
      real type variable
      real type variable with index
      index register
      constant ]
```

〔Explanation〕 The DZB instruction executes a dead zone operation on an integer or real type numerical value.

With an input value X, a dead zone specified value D, and an output value Y, the operation is performed as follows :

(1) $Y = X - D$ ($|X| \geqq |D|$)
(2) $Y = 0$ ($|X| < |D|$)



DZB instruction operation

〔Examples〕 (1) Integer type operation

```
          X              D          Y
 ├─  MW00100  DZB   00100  =>  MW00101
     (00150)                   (00150) ←Outside dead zone area
     (00050)                   (00000) ←Inside dead zone area
```

(2) Real type operation

```
          X              D          Y
 ||─  MF00200  DZB  100.00  =>  MF00202
     (150.00)                  (50.00) ←Outside dead zone area
     ( 50.00)                  ( 0.00) ←Inside dead zone area
```

8.10.3 LIM instruction

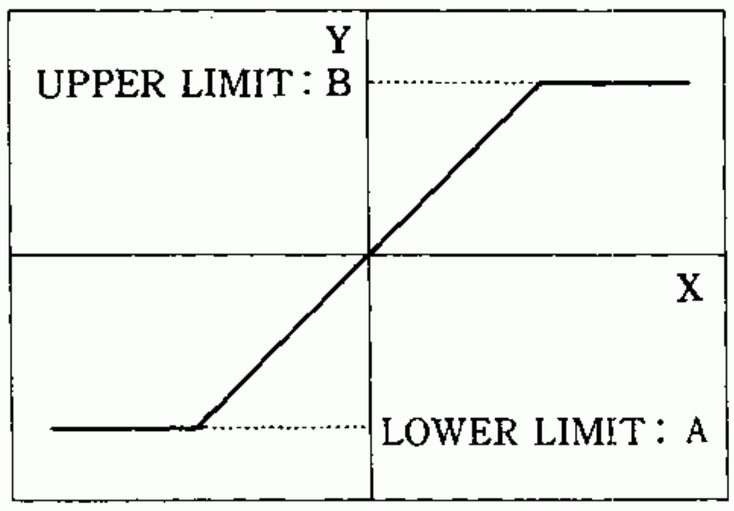
〔Format〕

	〔lower limit〕	〔upper limit〕
LIM	integer type variable	integer type variable
	integer type variable with index	integer type variable with index
	real type variable	real type variable
	real type variable with index	real type variable with index
	index register	index register
	constant	constant

〔Explanation〕 The LIM instruction executes an upper and lower limit operation on an integer or real type numerical value.

With an input value X, lower limit A, upper limit B, and output value Y, the operation proceeds as follows :

(1) Y = A (X < A)
(2) Y = X (A ≦ X ≦ B)
(1) Y = B (B < X)



LIM instruction operation

〔Examples〕 (1) Integer type operation

	X		A	B		Y	
├—	MW00100	LIM	-00100	00100	=>	MW00101	
	(-00150)					(-00100)	←Lower limit exceeded
	(00050)					(00050)	←Within upper and lower limit
	(00200)					(00100)	←Upper limit exceeded

(2) Real type operation

	X		A	B		Y	
├—	MF00200	LIM	-100.00	100.00	=>	MW00202	
	(150.00)					(-100.00)	←Lower limit exceeded
	(50.00)					(50.00)	←Within upper and lower limit
	(200.00)					(100.00)	←Upper limit exceeded

8.10.4 PI instruction

〔Format〕

〔parameter table top address〕

PI [variable address (except I, #)
     variable address (except I, #) with index]

〔Explanation〕 The PI instruction performs PI operation according to the contents of a preset parameter table.

This instruction is available in both integer and real type operations. The integer and real type operations differ in the structure of the parameter table.

(1) Integer type PI instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Relay I/O	Relay input and relay output *	IN/OUT
1	W	Kp	P gain	P compensation gain (100 for gain factor 1)	IN
2	W	Ki	Integral adjustment gain	A gain of integrating circuit input (100 for gain factor 1)	IN
3	W	Kd	Integral time	Integral time (ms)	IN
4	W	IUL	Integral upper LIMIT	Upper limit value for I compensation value	IN
5	W	ILL	Integral lower LIMIT	Lower limit value for P+I compensation value	IN
6	W	UL	PI upper LIMIT	Upper limit value for P+I compensation value	IN
7	W	LL	PI lower LIMIT	Lower limit value for I compensation value	IN
8	W	DB	PI output dead zone	Dead zone width for P+I compensation value	IN
9	W	Y	PI output	PI compensation output (Also output to register A)	OUT
10	W	Yi	I compensation value	I compensation value saving	OUT
11	W	IREM	I remainder	I remainder saving	OUT

(2) Real type PI instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Relay I/O	Relay input and relay output *	IN/OUT
1	W	—	(Not used)	Reserved register	—
2	F	Kp	P gain	P compensation gain	IN
4	F	Ki	Integral adjustment gain	A gain of integrating circuit input	IN
6	F	Ti	Integral time	Integral time (ms)	IN
8	F	IUL	Integral upper LIMIT	Upper limit value for I compensation value	IN
10	F	ILL	Integral lower LIMIT	Lower limit value for I compensation value	IN
12	F	UL	PI upper LIMIT	Upper limit value for P+I compensation value	IN
14	F	LL	PI lower LIMIT	Lower limit value for P+I compensation value	IN
16	F	DB	PI output dead zone	Dead zone width for P+I compensation value	IN
18	F	Y	PI output	PI compensation output (Also output to register F.)	OUT
20	F	Yi	I compensation value	I compensation value saving	OUT

* Relay I/O bit allocation

BIT	Symbol	Name	Specification	I/O
0	IRST	Integration reset	Enter "close" at integration reset.	IN
1 to 7	—	(Not used)	Reserved relay for input	IN
8 to F	—	(Not used)	Reserved relay for output	OUT

The PI operation is expressed as follows

$$\frac{Y}{X} = Kp + Ki * \frac{1}{Ti * S}$$

X : deviation input value

Y : output value

The following operation takes place inside the PI instruction :

$$Y = Kp * X + \{(Ki * X + IREM) / \frac{Ti}{Ts} + Yi'\}$$

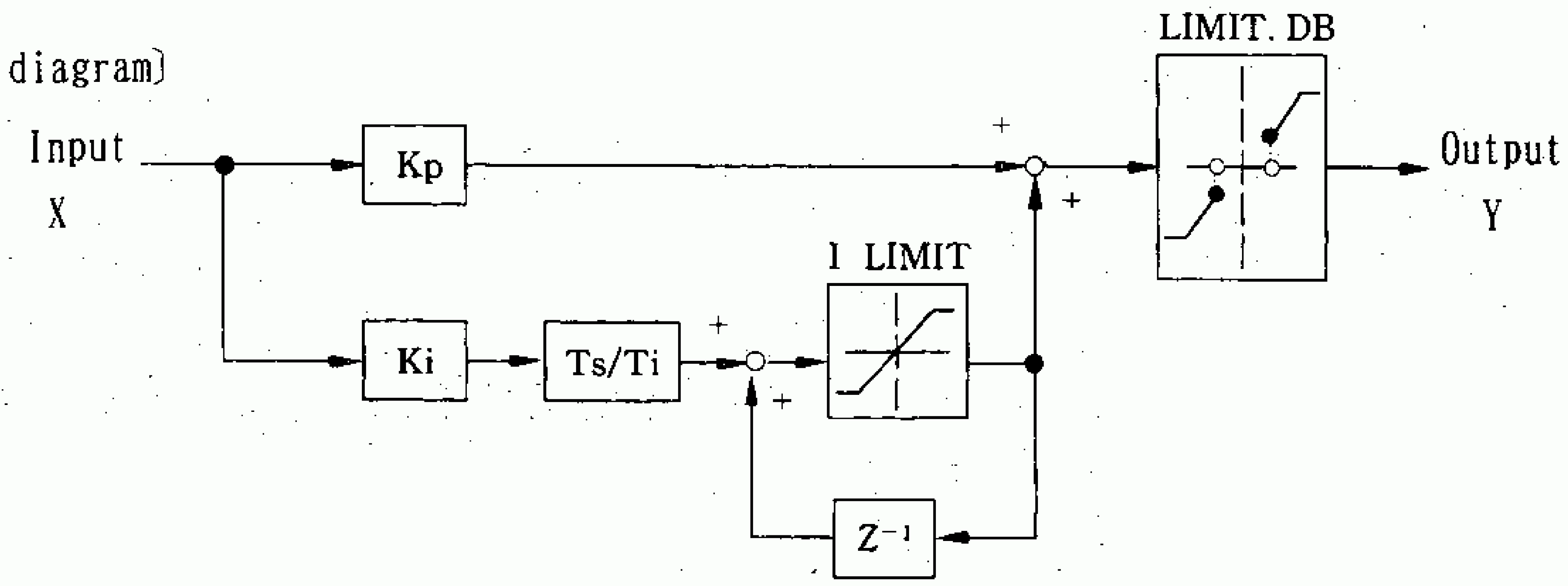
X' : previous input value

Y' : previous output value

Yi' : previous I output value

Ts : scan time preset value

〔Block diagram〕



When the P+I compensation touches the PI upper or lower limit (UL,LL) or the PI dead zone (DB), if the current P compensation and the I compensation are of the same sign (divergent), the I compensation is not updated but the previous value is used. Conversely, if they are of the opposite sign (convergent to 0), the I compensation is updated by the present value.

If the integration reset (IRST) is "closed", Yi = 0 and IREM = 0 are output.

〔Examples〕

(1) When a PI instruction is used in an integer type operation, MW00100 to MW00111 are used as a parameter table in the example below :

```
├─     DW00010   PI    MA00100          =>  DW00011
       ↑                ↑
       └─ Deviation     └─ Parameter table
          input            top address
```

(2) When an PI instruction is used in a real type operation, MF00200 to MF00220 are used as a parameter table in the example below :

```
||─    DF00020   PI    MA00200          =>  DF00022
       ↑                ↑
       └─ Deviation     └─ Parameter table
          input            top address
```

8.10.5 PD instruction

〔Format〕

〔parameter table top address〕

PD [variable address (except I, #) / variable address (except I, #) with index]

〔Explanation〕 The PD instruction executes a PD operation according to the content of a preset parameter table.

This instruction is available in both integer and real type operations. The integer and real type operations differ in the parameter table structure.

(1) Integer type PD instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Relay I/O	Relay input and relay output *	IN/OUT
1	W	Kp	P gain	P compensation gain (100 for gain factor of 1)	IN
2	W	Kd	D gain	Differential circuit input gain (100 for gain factor of 1)	IN
3	W	Td1	Divergent differential time	Differential time (ms) used when the input is divergent	IN
4	W	Td2	Convergent differential time	Differential time (ms) used when the input is convergent	IN
5	W	UL	PD upper LIMIT	Upper limit value for P+D compensation	IN
6	W	LL	PD lower LIMIT	Lower limit value for P+D compensation	IN
7	W	DB	PD output dead zone	Dead zone width for P+D compensation	IN
8	W	Y	PD output	PD compensation output (Also output to register A.)	OUT
9	W	X	Input value retention	Current deviation input value retention	OUT

(2) Real type PD instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Relay I/O	Relay input, relay output *	IN/OUT
1	W	—	(Reserve)	Reserve register	—
2	F	Kp	P gain	P compensation gain	IN
4	F	Kd	D gain	Differential circuit input gain	IN
6	F	Td1	Divergent differential time	Differential time (s) used when the input is divergent	IN
8	F	Td2	Convergent differential time	Differential time (s) used when the input is convergent	IN
10	F	UL	PD upper LIMIT	Upper limit value for P+D compensation	IN
12	F	LL	PD lower LIMIT	Lower limit value for P+D compensation	IN
14	F	DB	PD output dead zone	Dead zone width for P+D compensation	IN
16	F	Y	PD output	PD compensation output (Also output to register F.)	OUT
18	F	X	Input value retention	Current deviation input value retention	OUT

* Relay I/O bit allocation

BIT	Symbol	Name	Specification	I/O
0 to 7	—	(Not used)	Reserved relay for input	IN
8 to F	—	(Not used)	Reserved relay for output	OUT

The PD operation is expressed as follows :

$$\frac{Y}{X} = Kp + Kd * Td * S$$

X : deviation input value

Y : output value

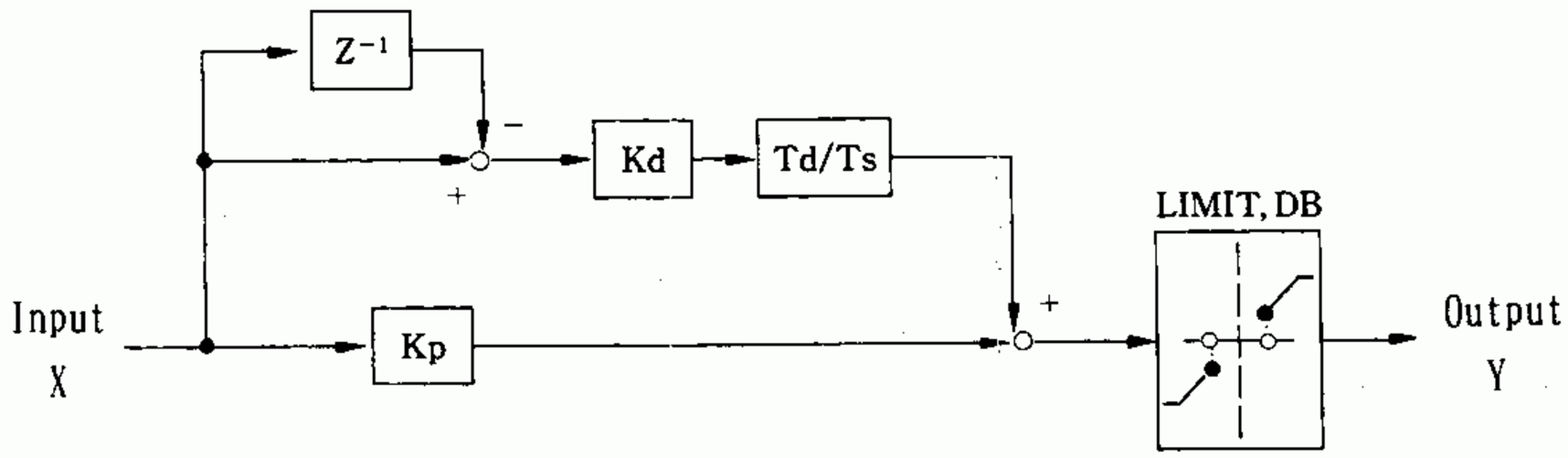
The following operation takes place inside the PD instruction :

$$Y = Kp * X + Kd * (X - X') * \frac{Td}{Ts}$$

X' : previous input value

Ts : scan time preset value

〔Block diagram〕



In the differentiation (D), if the deviation input change (X - X') has the same sign (in the divergent direction) as the previous deviation input (X), the divergent differentiation time (Td1) is used as a differential time. Conversely, if it has a different sign (in the convergent direction toward 0), the convergent differential time (Td2) is used.

〔Examples〕 (1) When a PD instruction is used in an integer type operation, MW00100 to MW00109 are used as a parameter table in the following example :

```
├─  DW00010   PD   MA00100        => DW00011
      └ Deviation      └ Parameter table
        input            top address
```

(2) When an PD instruction is used in a real type operation, MF00200 to MF00218 are used as a parameter table in the following example :

```
|├─ DF00020   PD   MA00200        => DF00022
      └ Deviation      └ Parameter table
        input            top address
```

8.10.6 PID instruction

〔Format〕

〔parameter table top address〕

```
PID [ variable address (except I, #)
    [ variable address (except I, #) with index ]
```

〔Explanation〕 The PID instruction performs PID operation according to the content of a preset parameter table.

This instruction can be usaed in both integer and real type operation. The integer and real type operations differ in the parameter table structure.

(1) Integer type PID instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Relay I/O	Relay input and relay output *	IN/OUT
1	W	Kp	P gain	P compensation gain (100 for gain factor 1)	IN
2	W	Ki	I gain	A gain of integrating circuit input (100 for gain factor 1)	IN
3	W	Kd	D gain	A gain of differentiating circuit input (100 for gain factor 1)	IN
4	W	Ti	Integral time	Integral time (ms)	IN
5	W	Td1	Divergent differential time	Differential time (ms) used when the input is divergent	IN
6	W	Td2	Convergent differential time	Differential time (ms) used when the input is convergent	IN
7	W	IUL	Integral upper LIMIT	Upper limit value for I compensation value	IN
8	W	ILL	Integral lower LIMIT	Lower limit value for I compensation value	IN
9	W	UL	PID upper LIMIT	Upper limit value for P+I+D compensation value	IN
10	W	LL	PID lower LIMIT	Lower limit value for P+I+D compensation value	IN
11	W	DB	PID output dead zone	Dead zone width for P+I+D compensation value	IN
12	W	Y	PID output	PID compensation output (Also output to register A)	OUT
13	W	Yi	I compensation value	I compensation value retention	OUT
14	W	IREM	I remainder	I remainder saving	OUT
15	W	X	Input value saving	Current deviation input value saving	OUT

(2) Real type PID instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Relay I/O	Relay input and relay output *	IN/OUT
1	W	—	(Not used)	Reserved register	—
2	F	Kp	P gain	P compensation gain	IN
4	F	Ki	I gain	A gain of integrating circuit input	IN
6	F	Kd	D gain	A gain of differentiating circuit input	IN
8	F	Ti	Integral time	Integral time (s)	IN
10	F	Td1	Divergent differential time	Differential time (s) used when the input is divergent	IN
12	F	Td2	Convergent differential time	Differential time (s) used when the input is convergent	IN
14	F	IUL	Integral upper LIMIT	Upper limit value for I compensation value	IN
16	F	ILL	Integral lower LIMIT	Lower limit value for I compensation value	IN
18	F	UL	PID upper LIMIT	Upper limit value for P+I+D compensation value	IN
20	F	LL	PID lower LIMIT	Lower limit value for P+I+D compensation value	IN
22	F	DB	PID output dead zone	Dead zone width for P+I+D compensation valve	IN
24	F	Y	PID output	PID compensation output (Also output to register F.)	OUT
26	F	Yi	I compensation value	I compensation value saving	OUT
28	F	X	Input value saving	Current deviation input value saving	OUT

* Relay I/O bit allocation

BIT	Symbol	Name	Specification	I/O
0	IRST	Integration reset	Enter "close" at integration reset.	IN
1 to 7	—	(Not used)	Reserved relay for input	IN
8 to F	—	(Not used)	Reserved relay for output	OUT

The PID operation is expressed as follows :

$$\frac{Y}{X} = Kp + Ki * \frac{1}{Ti * S} + Kd * Td * S$$

X : deviation input value

Y : output value

The following operation takes place inside the PID instruction :

$$Y = Kp * X + \{(Ki * X + IREM) \diagup \frac{Ti}{Ts} + Yi'\} + Kd * (X - X') * \frac{Td}{Ts}$$

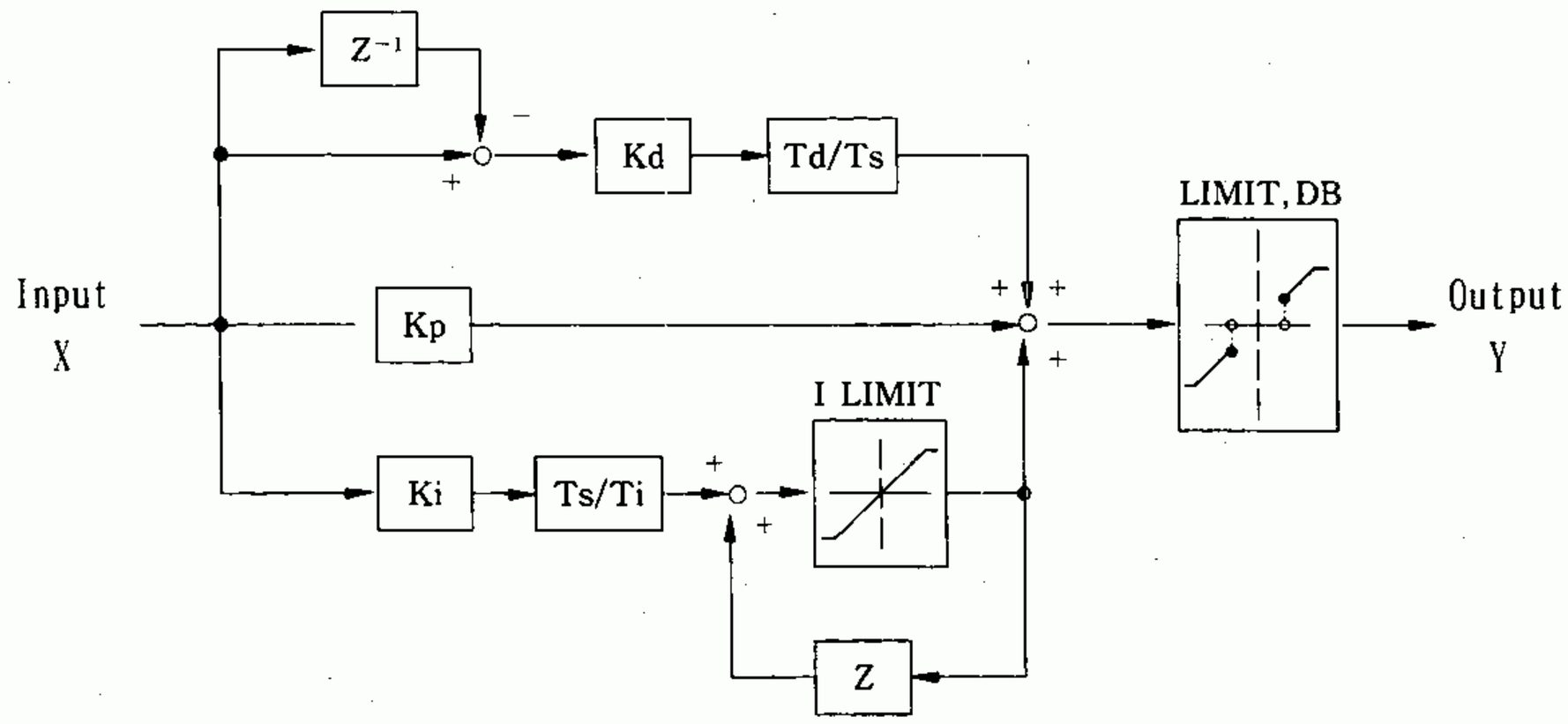
X' : previous input value

Y' : previous output value

Yi' : previous I output value

Ts : scan time preset value

〔Block diagram〕



When the P+I+D compensation reaches the PID upper or lower limit (UL, LL) or the PID dead band (DB), if the current P compensation and the I compensation are of the same sign (divergent), the I compensation is not updated but the previous value is used. Conversely, if they are of the opposite sign (convergent to 0), the I compensation is updated by the current value.

In the differentiation (D), if the deviation input change (X - X') has the same sign (in the divergent direction) as the previous deviation input (X), the divergent differential time (Td1) is used as a differential time. Conversely, if it has a different sign (in the convergent direction toward 0), the convergent differential time (Td2) is used.

If the integration reset (IRST) is "closed", Yi = 0 and IREM = 0 are output.

〔Examples〕 (1) When a PID instruction is used in an integer type operation, MW00100 to MW00115 are used as a parameter table in the example below :

```
├─ DW00010   PID   MA00100        => DW00011
     ↑              ↑
     └ Deviation    └ Parameter table
       input          top address
```

(2) When an PID instruction is used in a real type operation, MF00200 to MF00228 are used as a parameter table in the example below ;

```
├─ DF00020   PID   MA00200        => DF00022
     ↑              ↑
     └ Deviation    └ Parameter table
       input          top address
```

8.10.7 LAG instruction

〔Format〕

〔parameter table top address〕

LAG [variable address (except I, #) / variable address (except I, #) with index]

〔Explanation〕 The LAG instruction performs a primary delay operation according to the content of a preset parameter table.

This instruction is available in an integer or real type operations. The integer and real type operations differ in the parameter table structure.

(1) Integer type LAG instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Relay I/O	Relay input, relay output *	IN/OUT
1	W	T	Primary delay time constant	Primary delay time constant (ms)	IN
2	W	Y	LAG output	LAG output (Also output to register A.)	OUT
3	W	REM	Remainder	Remainder retention	OUT

(2) Real type LAG instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Relay I/O	Relay input, relay output *	IN/OUT
1	W	—	(Not used)	Reserved register	—
2	F	T	Primary delay time constant	Primary delay time constant (s)	IN
4	F	Y	LAG output	LAG output (Also output to register F.)	OUT

* Relay I/O bit allocation

BIT	Symbol	Name	Specification	I/O
0	RST	LAG reset	Enter "closed" at LAG reset.	IN
1 to 7	—	(Not used)	Reserved relay for input	IN
8 to F	—	(Not used)	Reserved relay for output	OUT

The primary delay is expressed as follows :

$$\frac{Y}{X} = \frac{1}{1 + T * S} \qquad \{T * (dY/dt) + Y = X\}$$

The following operation takes place inside the LAG instruction with dt = Ts and dY= Y - Y' :

$$Y = \frac{T * Y' + Ts * X + REM}{T + Ts}$$

X : Input value
Y : Output value
Y' : Previous output value
Ts: Scan time preset value
(REM : A remainder is used only for an integer type.)

When the LAG reset (RST) is "closed", Y = 0 and REM = 0 are output.

〔Examples〕 (1) When a LAG instruction is used in an integer type operation, MW00100 to MW00103 are used as a parameter table in the example below :

```
|—  DW00010   LAG   MA00100        => DW00011
     ↑              ↑
     └ Input value  └ Parameter table top address
```

(2) When an LAG instruction is used in a real type operation, MF00200 to MF00204 are used as a parameter table in the example below :

```
||—  DF00020   LAG   MA00200        => DF00022
      ↑              ↑
      └ Input value  └ Parameter table top address
```

8.10.8 LLAG instruction

〔Format〕 〔parameter table top address〕

LLAG [variable address (except I, #) / variable address (except I, #) with index]

〔Explanation〕 The LLAG instruction performs a primary delay operation according to the contents of a preset parameter table.

This instruction is available in an integer or real type operation. The integer and real type operations differ in the parameter table structure.

(1) Integer type LLAG instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Relay I/O	Relay input, relay output *	IN/OUT
1	W	T2	Phase advance time constant	Phase advance time constant (ms)	IN
2	W	T1	Phase delay time constant	Phase delay time constant (ms)	IN
3	W	Y	LLAG output	LLAG output (Also output to register A.)	OUT
4	W	REM	Remainder	Remainder retention	OUT
5	W	X	Input value retention	Input value retention	OUT

(2) Real type LLAG instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Relay I/O	Relay input , relay output *	IN/OUT
1	W	—	(Reserve)	Reserve register	—
2	F	T2	Phase advance time constant	Phase advance time constant (s)	IN
4	F	T1	Phase delay time constant	Phase delay time constant (s)	IN
6	F	Y	LLAG output	LLAG output (Also output to register F.)	OUT
8	F	X	Input value saving	Input value saving	OUT

* Relay I/O bit allocation

BIT	Symbol	Name	Specification	I/O
0	RST	LAG reset	Enter "closed" at LAG reset.	IN
1 to 7	—	(Not used)	Reserved relay for input	IN
8 to F	—	(Not used)	Reserved relay for output	OUT

The phase advance and delay is expressed as follows :

$$\frac{Y}{X} = \frac{1 + T2 * S}{1 + T1 * S} \quad \{T1 * (dY/dt) + Y = T2 * (dX/dt) + X\}$$

The following operation takes inside the LLAG instruction with dt = Ts, dY = Y - Y', and dX = X - X' :

$$Y = \frac{T1 * Y' + (T2 + Ts) * X - T2 * X' + REM}{T1 + Ts}$$

X : Input value
Y : Output value
X' : Previous input value
Y' : Previous output value
Ts: Scan time preset value
(REM : A remainder is used only for an integer type.)

When the LLAG reset (RST) is "closed", Y = 0, REM = 0 and X = 0 are output.

〔Examples〕 (1) When a LLAG instruction is used in an integer type operation, MW00100 to MW00105 are used as a parameter table in the example below :

```
├─ DW00010   LLAG   MA00100        => DW00011
     ↑              ↑
     └ Input value  └ Parameter table top address
```

(2) When an LLAG instruction is used in a real type operation, MF00200 to MF00208 are used as a parameter table in the example below :

```
|├─ DF00020   LLAG   MA00200        => DF00022
      ↑              ↑
      └ Input value  └ Parameter table top address
```

8.10.9 FGN instruction

〔Format〕

〔parameter table top address〕

FGN [variable address / variable address with index]

〔Explanation〕 The FGN instruction generates function curve according to the contents of a preset parameter table.

This instruction is available in both integer and real type operations. The integer and real type operations differ in the parameter table structure.

(1) Integer type FGN instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	N	Number of data	Number of X, Y pairs	IN
1	W	X1	Data 1		IN
2	W	Y1	Data 1		IN
3	W	X2	Data 2		IN
4	W	Y2	Data 2		IN
:	:	:	:	:	
:	:	:	:	:	
2N-1	W	XN	Data N		IN
2N	W	YN	Data N		IN

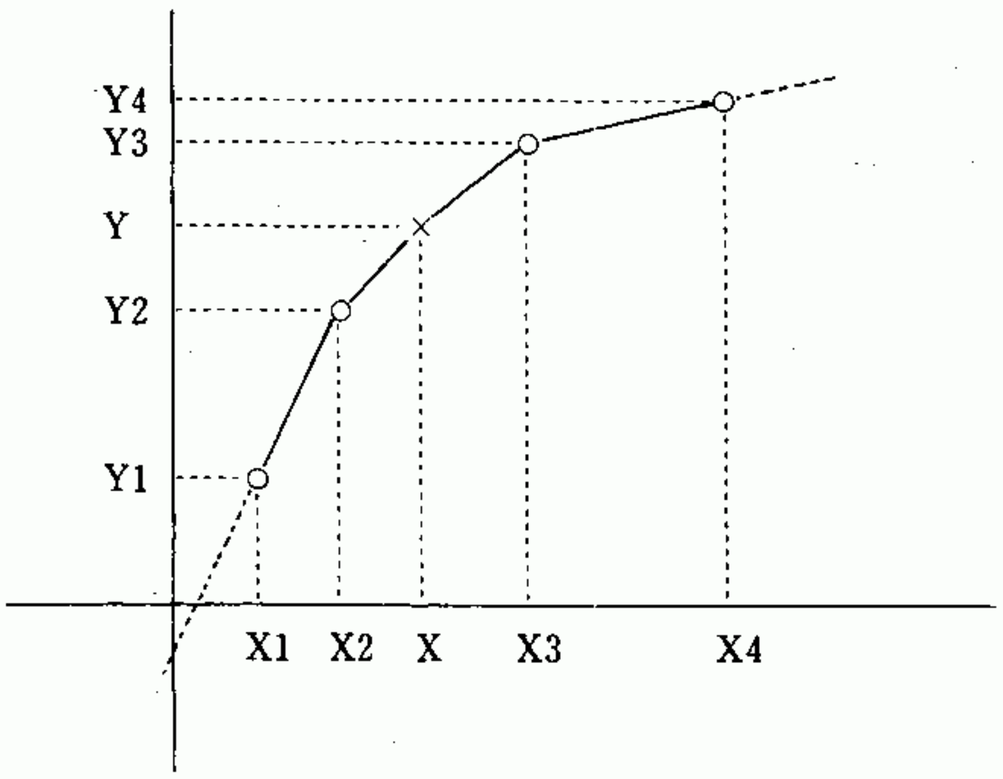
(2) Real type FGN instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	N	Number of data	Number of X, Y pairs	IN
1	W	—	(Not used)	Reserved register	IN
2	F	X1	Data 1		IN
4	F	Y1	Data 1		IN
6	F	X2	Data 2		IN
8	F	Y2	Data 2		IN
:	:	:	:	:	
:	:	:	:	:	
4N-2	F	XN	Data N		IN
4N	F	YN	Data N		IN

Let Xn, Yn be data set up by the parameter table of an FGN instruction. Then the data must be such Xn ≦ Xn+1. The FGN instruction searches the parameter table for Xn, Yn pairs such that Xn ≦ X ≦ Xn+1 for the input value X and computes the output value Y by the following formula :

$$Y = Yn + \frac{Yn+1 - Yn}{Xn+1 - Xn} \times (X - Xn)$$

The following diagram shows the relationship between the parameter table setup data, input value X, and output value Y :



〔Examples〕 (1) When an FGN instruction is used in an integer type operation, #W00000 to #W00040 are used as a parameter table in the example below (with the number N of data = 20 assumed) :

```
|— DW00010   FGN   #A00000        => DW00011
     ↑              ↑
     └ Input value  └ Parameter table top address
```

(2) When an FGN instruction is used in a real type operation, #W00000 to #F00080 are used as a parameter table in the example below (with the number N of data = 20 assumed) :

```
|— DF00020   FGN   #A00000        => DF00022
     ↑              ↑
     └ Input value  └ Parameter table top address
```

8.10.10 IFGN instruction

〔Format〕

〔parameter table top address〕

IFGN [variable address
       variable address with index]

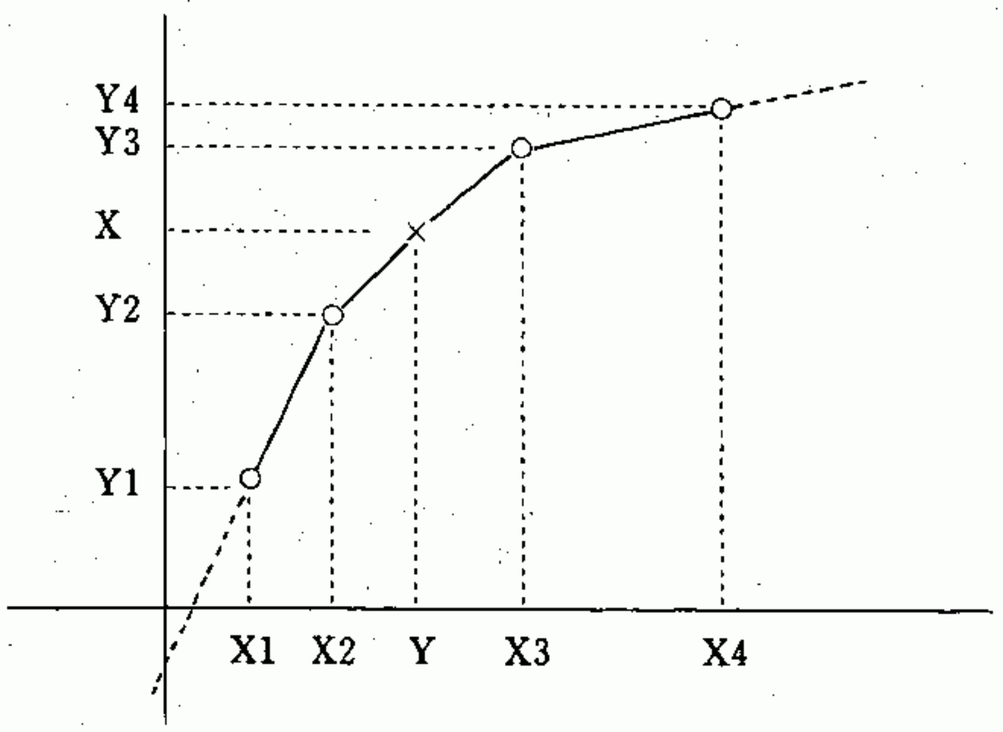
〔Explanation〕 The IFGN instruction uses a parameter table generated by an FGN instruction in the opposite direction.

This instruction is available in both integer and real type operations.

Let Xn, Yn be data set up by the parameter table of an IFGN instructaion. Then the data must be set up so that $Yn \leqq Yn+1$. The IFGN instruction searches the parameter table for Xn, Yn pairs such that $Yn \leqq Y \leqq Yn+1$ for the input value X and computes the output value Y by the following formula :

$$Y = Xn + \frac{Xn+1 - Xn}{Yn+1 - Yn} \times (X - Yn)$$

The following diagram shows the relationship between the parameter table setup data, input value X, and output value Y :



〔Examples〕 (1) When an IFGN instruction is used in an integer type operation, #W00000 to #W00040 are used as a parameter table in the example below (with the number N of data = 20 assumed) :

```
├─ DW00010   IFGN   #A00000        =>  DW00011
     ↑              ↑
     └─ Input value └─ Parameter table top address
```

(2) When an IFGN instruction is used in a real type operation, #W00000 to #F00080 are used as a parameter table in the example below (with the number N of data = 20 assumed) :

```
├─ DF00020   IFGN   #A00000        =>  DF00022
     ↑              ↑
     └─ Input value └─ Parameter table top address
```

8.10.11 LAU instruction

〔Format〕

〔parameter table top address〕

LAU [variable address (except I, #)
variable address (except I, #) with index]

〔Explanation〕 The LAU instruction performs acceleration or deceleration at a constant rate on a speed input (value of register A or F). The operation follows the content of a preset parameter table.

This instruction is available in both integer and real type operations. The integer and real type operations differ in the parameter table structure.

(1) Integer type LAU instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Ralay I/O	Relay input, relay output *	IN/OUT
1	W	LV	Input 100% level	Scale of 100% of input value	IN
2	W	AT	Acceleration time	Time to accelerate from 0 to 100% (0.1s)	IN
3	W	BT	Deceleration time	Time to decelerate from 100 to 0% (0.1s)	IN
4	W	QT	Quick stop time	Time for quick stop from 100 to 0% (0.1s)	IN
5	W	V	Current speed	LAU output (Also output to register A.)	OUT
6	W	DVDT	Current accel/decel time	Scaling with normal acceleration rate at 5000	OUT

(2) Real type LAU instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Ralay I/O	Relay input, relay output *	IN/OUT
1	W	—	(Reserve)	Reserve register	—
2	F	LV	Input 100% level	Scale of 100% of input value	IN
4	F	AT	Acceleration time	Time to accelerate from 0 to 100% (s)	IN
6	F	BT	Deceleration time	Time to decelerate from 100 to 0% (s)	IN
8	F	QT	Quick stop time	Time for quick stop from 100 to 0% (s)	IN
10	F	V	Current speed	LAU output (Also output to register F.)	OUT
12	F	DVDT	Current accel/decel time	Output the current accel/decel speed	OUT

* Relay I/O bit allocation

BIT	Symbol	Name	Specification	I/O
0	RN	Line in operation	Enter "closed" when line operation is in progress	IN
1	QS	Quick stop	Enter "open" at quick stop	IN
2 to 7	—	(Not used)	Reserved relay for input	IN
8	ARY	Accelerating	Output "closed" during acceleration	OUT
9	BRY	Decelerating	Output "closed" during deceleration	OUT
A	SLP	Zero speed	Output "closed" at zero speed	OUT
B B	EQU	Matched	Output "closed" when input value = output value	OUT
C to F	—	(Not used)	Reserved relay for output	OUT

(3) The accel/decel rate inside an integer type LAU instruction is operated as follows :

Ts : scan time preset value ; V' : previous speed output

① Acceleration rate (ADV) = $\dfrac{LV}{\dfrac{AT\ (0.1s) * 100}{Ts\ (ms)}}$

When V1 > V' (V' > 0)
V = V' + ADV
When V1 < V' (V' < 0)
V = V' - ADV

② Deceleration rate (BDV) = $\dfrac{-LV}{\dfrac{BT\ (0.1s) * 100}{Ts\ (ms)}}$

When V1 < V' (V' > 0)
V = V' + BDV
When V1 > V' (V' < 0)
V = V' - BDV

③ Quick stop rate (QDV) = $\dfrac{-LV}{\dfrac{QT\ (0.1s) * 100}{Ts\ (ms)}}$

When QS = ON (V' > V1 ≧ 0)
V = V' + QDV
When QS = ON (V' < V1 ≦ 0)
V = V' - QDV

The current acceleration or deceleration speed (DVDT) is output after the following operation :

$$DVDT = \frac{V - V'}{ADV} * 5000$$

When the line in operation (RN) is "open", V = 0 and DVDT = 0 are output.

(4) The accel/decel rate inside a real type LAU instruction is operated as follows :

Ts : scan time preset value, V' : previous speed output

$$\text{Acceleration rate (ADV)} = \frac{LV}{\frac{AT\ (s) * 1000}{Ts\ (ms)}}$$

When V1 > V' (V' > 0)
V = V' + ADV
When V1 < V' (V' < 0)
V = V' - ADV

$$\text{Deceleration rate (BDV)} = \frac{-LV}{\frac{BT\ (s) * 1000}{Ts\ (ms)}}$$

When V1 < V' (V' > 0)
V = V' + BDV
When V1 > V' (V' < 0)
V = V' - BDV

$$\text{Quick stop rate (QDV)} = \frac{-LV}{\frac{QT\ (s) * 1000}{Ts\ (ms)}}$$

When QS = ON (V' > V1 ≧ 0)
V = V' + QDV
When QS = ON (V' < V1 ≦ 0)
V = V' - QDV

The actual value is output for the current acceleration or deceleration speed (DVDT) :

DVDT = V - V'

When the line in operation (RN) is "open", V = 0 and DVDT = 0 are output.

〔Examples〕 (1) When an LAU instruction is used in an integer type operation, MW00100 to MW00106 are used as a parameter table in the example below :

```
├─ DW00010   LAU   MA00100        => DW00011
     ↑              ↑
     └ Speed command input   └ Parameter table top address
```

(2) When a LAU instruction is used in a real type operation, MF00200 to MF00212 are used as a parameter table in the example below :

```
├─ DF00020   LAU   MA00200        => DF00022
     ↑              ↑
     └ Speed command input   └ Parameter table top address
```

8.10.12 SLAU instruction

〔Format〕

〔parameter table top address〕

SLAU [variable address (except I, #)
       variable address (except I, #) with index]

〔Explanation〕 The SLAU instruction performs acceleration or deceleration at a variable rate on a speed command input (value of register A or F). The operation follows the content of a preset parameter table.

This instruction is available in both integer and real type operations. The integer and real type operations differ in the parameter table structure.

(1) Integer type SLAU instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Relay I/O	Relay input, relay output *	IN/OUT
1	W	LV	Input 100% level	Scale of 100% of input value	IN
2	W	AT	Acceleration time	Time to accelerate from 0 to 100% (0.1s)	IN
3	W	BT	Deceleration time	Time to decelerate from 100 to 0% (0.1s)	IN
4	W	QT	Quick stop time	Time for quick stop from 100 to 0% (0.1s)	IN
5	W	AAT	Acceleration S-shape time	S-shape area time during acceleration (0.01s)	IN
6	W	BBT	Deceleration S-shape time	S-shape area time during deceleration (0.01s)	IN
7	W	V	Current speed	SLAU output (Also output to register A.)	OUT
8	W	DVDT 1	Current accel/decel 1	Scaling with normal acceleration rate at 5000	OUT
9	W	DVDT 2	Current accel/decel 2	Actual accel/decel times 100	OUT
10	W	ABMD	Speed increase during hold	Speed change before setting following a hold command	OUT
11	W	REM	Remainder	Remainder saving	OUT

(2) Real type SLAU instruction parameter table

ADR	Type	Symbol	Name	Specification	I/O
0	W	RLY	Relay I/O	Relay input, relay output *	IN/OUT
1	W	—	(Not used)	Reserved register	—
2	F	LV	Input 100% level	Scale of 100% of input value	IN
4	F	AT	Acceleration time	Time to accelerate from 0 to 100% (s)	IN
6	F	BT	Deceleration time	Time to decelerate from 100 to 0% (s)	IN
8	F	QT	Quick stop time	Time for quick stop from 100 t0 0% (s)	IN
10	F	AAT	Acceleration S-shape time	S-shape area time during acceleration (s)	IN
12	F	BBT	Deceleration S-shape time	S-shape area time during deceleration (s)	IN
14	F	V	Current speed	SLAU output (Also output to register F.)	OUT
16	F	DVDT	Current accel/decel	Output the current accel/decel	OUT
18	F	ABMD	Speed increase during hold	Speed change before setting following a hold command	OUT

* Relay I/O bit allocation

BIT	Symbol	Name	Specification	I/O
0	RN	Line in operation	Enter "closed" when line operation is in progress	IN
1	QS	Quick stop	Enter "open" at quick stop	IN
2 to 7	—	(Not used)	Reserved relay for inupt	IN
8	ARY	Accelerating	Output "closed" during acceleration	OUT
9	BRY	Decelerating	Output "closed" during deceleration	OUT
A	LSP	Zero speed	Output "closed" at zero speed	OUT
B	RQU	Matched	Output "closed" when input value = output value	OUT
C to F	—	(Reserve)	Reserved relay for output	OUT

(3) The accel/decel rate inside an integer type SLAU instruction is operated as follows :

Ts : scan time current value ; V' : previous speed output

$$\text{Acceleration rate (ADV)} = \frac{LV * 100}{\frac{AT\ (0.1s) * 100}{Ts\ (ms)}}$$

When VI > V' (V' > 0) outside the S-shape section (ADVS > ADV):
V = V' + (ADV + REM) / 100

$$\text{Deceleration rate (BDV)} = \frac{-LV * 100}{\frac{BT\ (0.1s) * 100}{Ts\ (ms)}}$$

When VI < V' (V' > 0) outside the S-shape section (BDVS < BDV):
V = V' + (BDV + REM) / 100

$$\text{Quick stop rate (QDV)} = \frac{-LV * 100}{\frac{QT\ (0.1s) * 100}{Ts\ (ms)}}$$

When QS = ON (V' > VI ≧ 0) :
V = V' + (QDV + REM) / 100

$$\text{S-shape section acceleration rate (ADVS)} = ADVS' \pm \frac{ADV}{\frac{AAT\ (0.1s) * 10}{Ts\ (ms)}}$$

When VI > V' (V' > 0) inside the S-shape section (ADVS < ADV)
V = V' + (ADVS + REM) / 100

$$\text{S-shape section deceleration rate (BDVS)} = BDVS' \pm \frac{BDV}{\frac{BBT\ (0.1s) * 10}{Ts\ (ms)}}$$

When VI < V' (V' > 0) inside the S-shape section (BDVS > BDV)
V = V' + (BDVS + REM) / 100

The current accel/decel speed 1 (DVDT1) is output after the following operation :

$$DVDT1 = \frac{V - V'}{ADV} * 5000$$

The current accel/decel 2 (DVDT2) is output as follows :

(1) Inside S-shape section during acceleration : DVDT2 = ADVS
Outside S-shape section during acceleration : DVDT2 = ADV

(2) Inside S-shape section during deceleration : DVDT2 = BDVS
Outside S-shape section during deceleration : DVDT2 = BDV

The speed increase during hold (ABMD) is output after the following operation :

$$ABMD = \frac{DVDT2 * DVDT2}{2 * AADVS\ (BBDVS)}$$

When the line operation in progress (RN) is "open", V = 0, DVDT1 = 0, DVDT2 = 0, ABMD = 0 and REM = 0 are output.

(4) The accel/decel rate inside a real type SLAU instruction is operated as follows :
Ts : scan time current value ; V' : previous speed output

Acceleration rate (ADV) = $\dfrac{LV}{\dfrac{AT\ (s) * 1000}{Ts\ (ms)}}$

When V1 > V' (V' > 0) outside the S-shape section (ADVS > ADV) :
V = V' + ADV

Deceleration rate (BDV) = $\dfrac{-LV}{\dfrac{BT\ (s) * 1000}{Ts\ (ms)}}$

When V1 < V' (V' > 0) outside the S-shape section (BDVS < BDV) :
V = V' + BDVS

Quick stop rate (QDV) = $\dfrac{-LV}{\dfrac{QT\ (s) * 1000}{Ts\ (ms)}}$

When QS = ON (V' > V1 ≧ 0) :
V = V' + QDV

S-shape section acceleration rate (ADVS) = ADVS' ± $\dfrac{ADV}{\dfrac{AAT(s) * 1000}{Ts(ms)}}$

When V1 > V' (V' > 0) outside the S-shape section (ADVS < ADV) :
V = V' + ADVS

S-shape section deceleration rate (BDVS) = BDVS' ± $\dfrac{BDV}{\dfrac{BBT(s) * 1000}{Ts(ms)}}$

When V1 < V' (V' < 0) inside the S-shape section (BDVS > BDV) :
V = V' + BDV

The current accel/decel (DVDT) is output as follows :

(1) Inside S-shape section during acceleration : DVDT = ADVS
Outside S-shape section during acceleration : DVDT = ADV

(2) Inside S-shape section during deceleration : DVDT = BDVS
Outside S-shape section during deceleration : DVDT = BDV

The speed increase during hold (ABMD) is output after the following operation :

$$ABMD = \frac{DVDT * DVDT}{2 * AADVS\ (BBDVS)}$$

When the line operation in progress (RN) is "open", V = 0, DVDT = 0 and ABMD = 0 are output.

〔Examples〕 (1) When an SLAU instruction is used in an integer type operation, MW00100 to MW00111 are used as a parameter table in the example below :

```
├─ DW00010   SLAU   MA00100        => DW00011
   └ Speed command   └ Parameter table
     input             top address
```

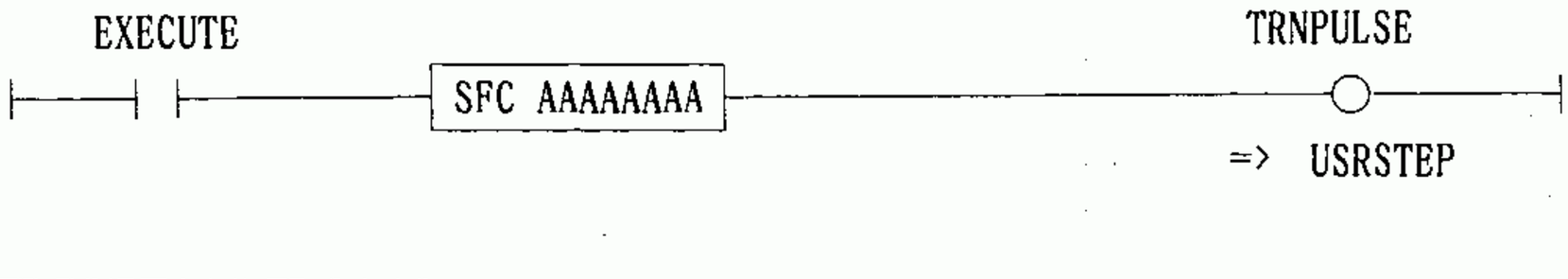
(2) When an SLAU instruction is used in a real type operation, MF00200 to MF00218 are used as a parameter table in the example below :

```
├─ DF00020   SLAU   MA00200        => DF00022
   └ Speed command   └ Parameter table
     input             top address
```

8.11 SFC PROGRAM

(1) SFC execution

The SFC program is performed by a DWG program using an SFC instruction shown below :



No.	Item	Content
1	AAAAAAAA	SFC name. Used as a comment.
2	EXECUTE	ON : Controls step change. OFF: Sets an initial step.
3	TRNPULSE	Goes ON when the step is changed.
4	USRSTEP	A user step corresponding to the current system step is output.

(2) Variable area in SFC execution

The variable area necessary for SFC execution is determined as shown in the table below. These variables must not be used for other purposes in a DWG using SFC.

Variable	Variable name	Content
DW00000	Step current value	Current step number
DW00001	Step previous value	Step no. preceding to the current step
DW00002	Change timer current value	Change timer current value (10 ms)
DW00003	Step search input	Search for a system step matching the user step
DW00004	SFC output-1	Output data from SFC output timing chart
DW00005	SFC output-2	Output data from SFC output timing chart
DW00006	SFC output-3	Output data from SFC output timing chart
DW00007	SFC output-4	Output data from SFC output timing chart
DW00008	System work SFC control	DW00008: Control the SFC processing sequence
DW00009		DW00009: Control concurrent processing
DW00010 to DW00029	System work concurrent processing control register	DW00010-DW00019: Hold the current step of each branch step for concurrent processing DW00020-DW00029: Used for counting a change timer for concurrent processing

(3) SFC flowchart

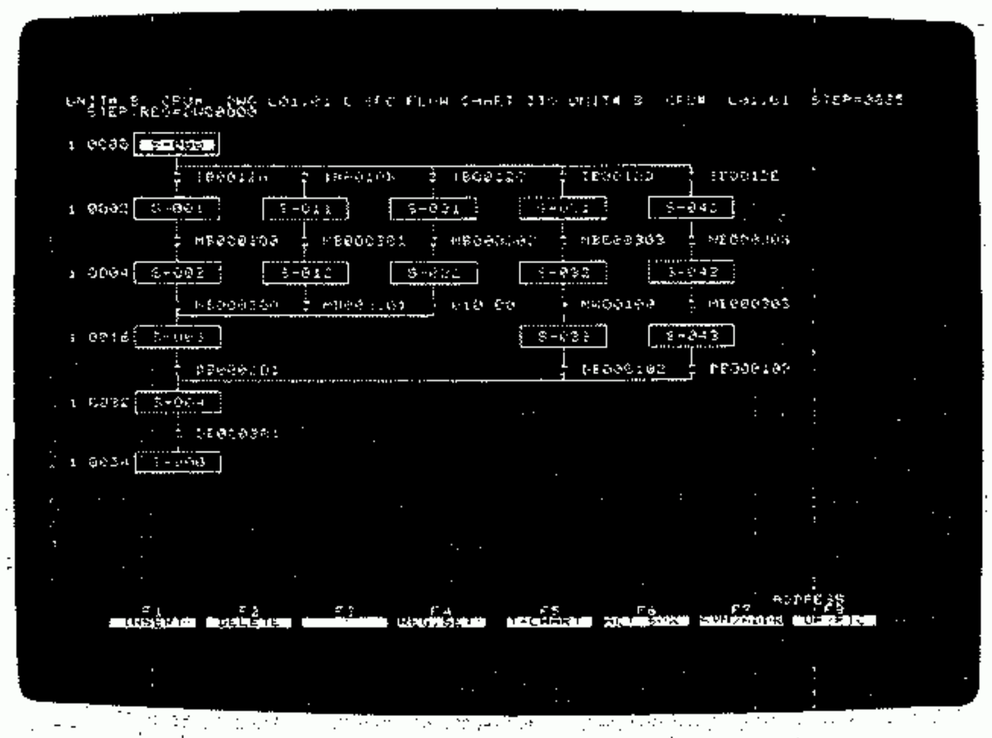
An SFC flowchart is created by using a step, change condition, and connection specification.

If one flowchart is insufficient, two or more flowcharts can be used.

Step : Represented by a box and step no.

Change condition : Contact A condition (⊣⊢), contact B condition (⊣/⊢), and change timer condition (+) are available.

Connection specification : A branch, join, and join and connect can be specified.



690-6

(4) Handling SFC step names

The step name of each step input when creating an SFC flowchart is handled as follows :

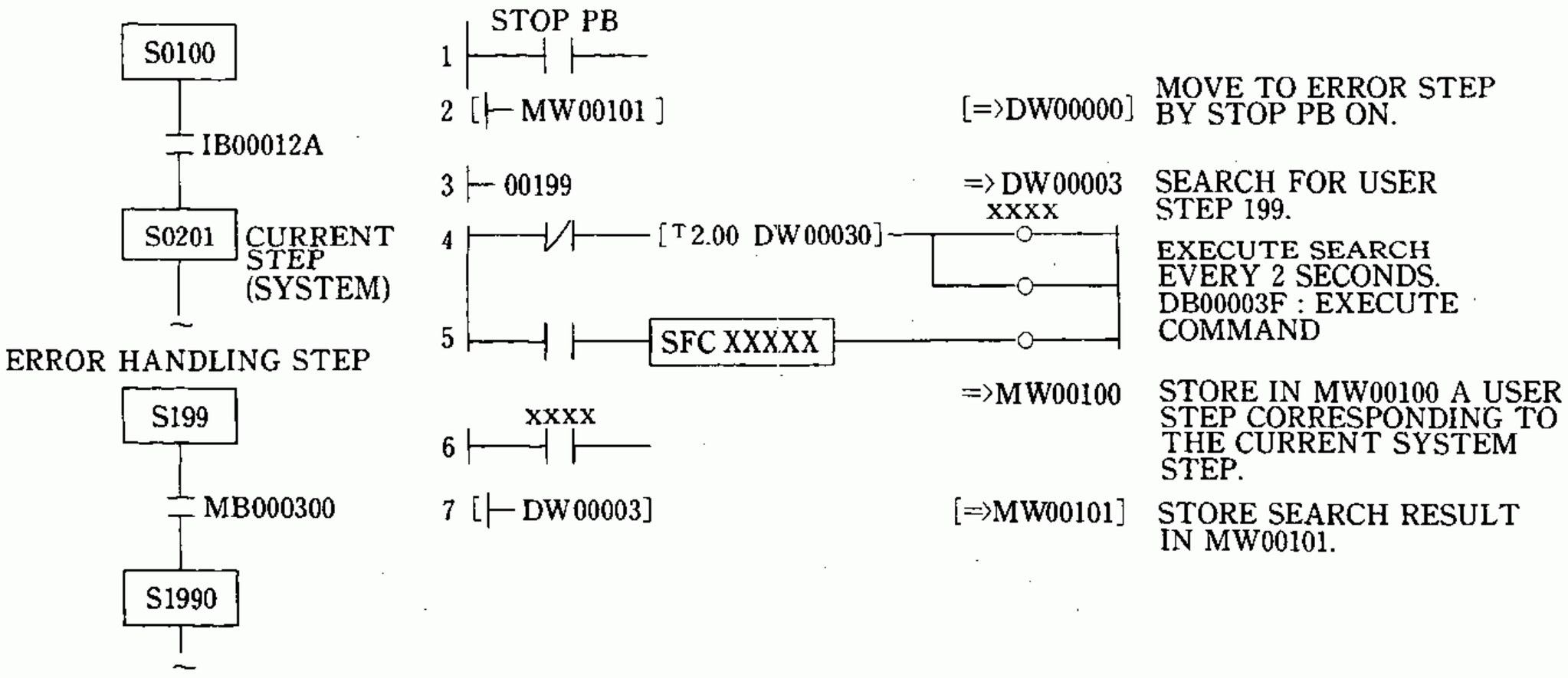
(a) For a name beginning with a letter "S", the character string following "S" is interpreted as a numeric value (user step number).

(b) If the first character is not "S" or the character string following "S" contains a non-numeric character, the step is considered as a symbol and the user step 0000 assumed.

For example, the step name "S0100" is interpreted as user step number = 100 and the step name "S-0100" as usear step number = 000.

The value allowed as a user number ranges from 00000 to 32767.

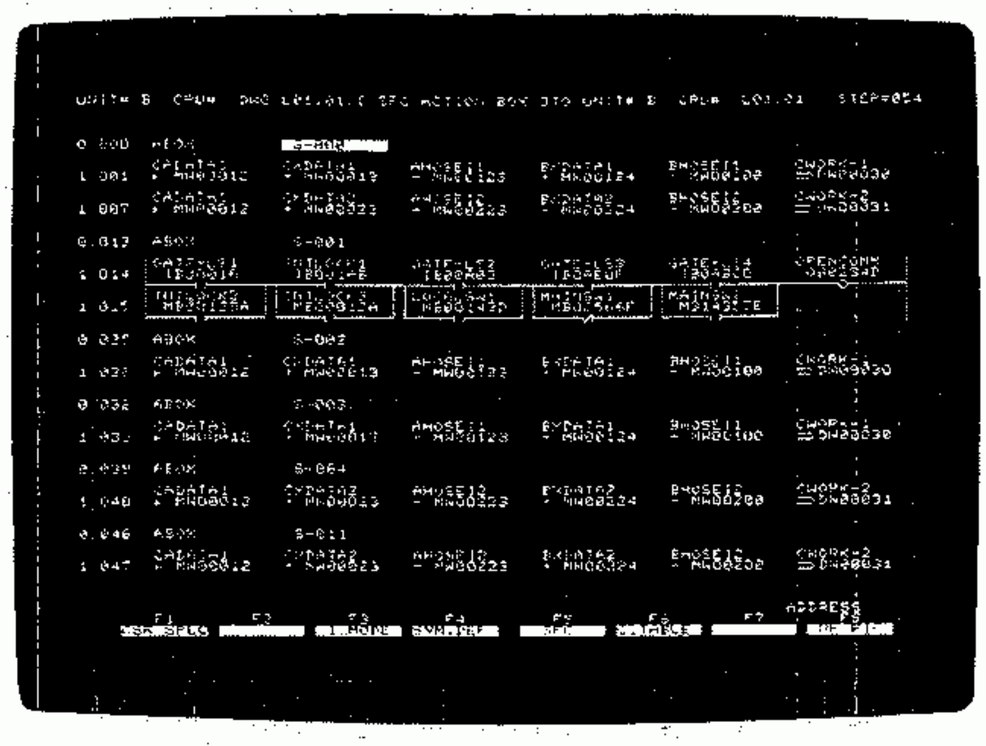
As explained above, the user step number is a value input as a step name. The step number stored in DW00000 (DW00010 to DW00019 in concurrent processing) means the system-managed SFC step number. The system step number is assigned automatically by the system to each step name when the SFC flowchart is edited. Consequently, the value of the step number (stored in DW00000) of the same step name changes by the editing, which may make processing in an error handling sequence, for example, complex. To avoid this, a user-defined user step number may be used as follows. Then an error sequence may be created easily.



Note: 00199 is used as a user step number of an error handling step.

(5) SFC action box

The SFC action box specifies a step no. by using an ABOX or SBOX instruction and creates a program for each SFC step by using a ladder diagram and text language.



690-7

The following is a difference between ABOX ans SBOX :

In ABOX, once the change comes back to itself, scan is repeated until the change moves to the next. In SBOX, execution is performed only once.

(6) SFC output timing chart

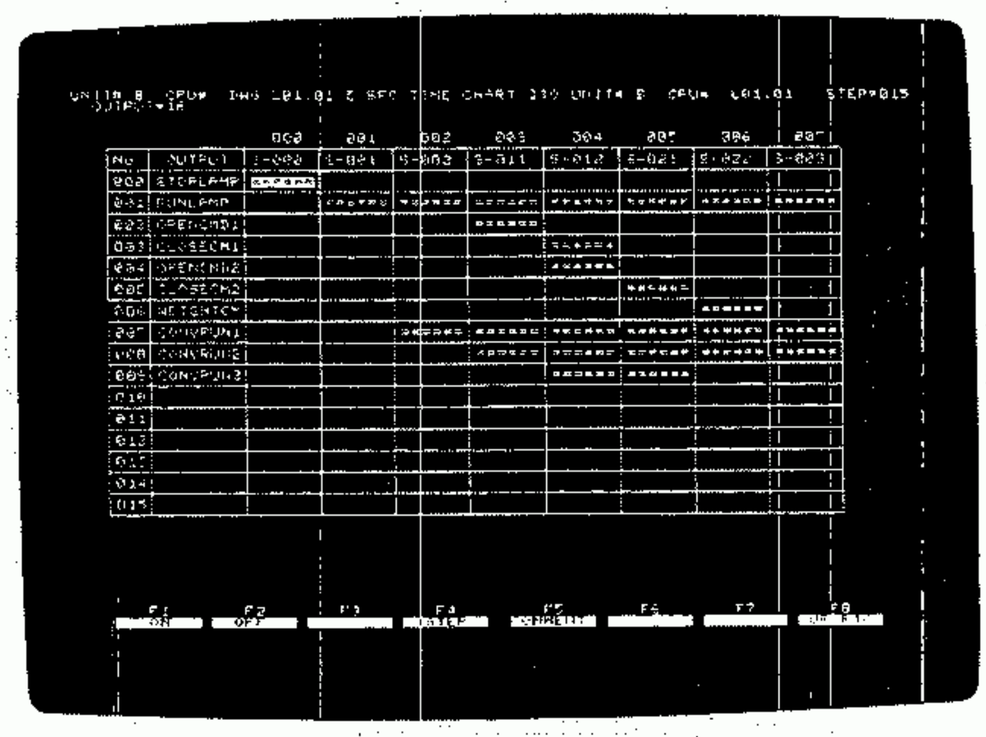
In an SFC output timing chart, any sequence of step numbers and number of output points are specified and ON/OFF of output data in the relevant step specified by using the following timing chart.

The output data specified by this timing chart are output to the variable area (DW00004 to DW00007) during SFC execution.

Step no. : Specify a sequence of step numbers by any combination.

Number of output points : Can be specified in 16-point units. (Max. 128 points)

Output name : Can be defined with not more than eight alphanumeric characters. Used as a comment.



690-8

SFC output timing chart

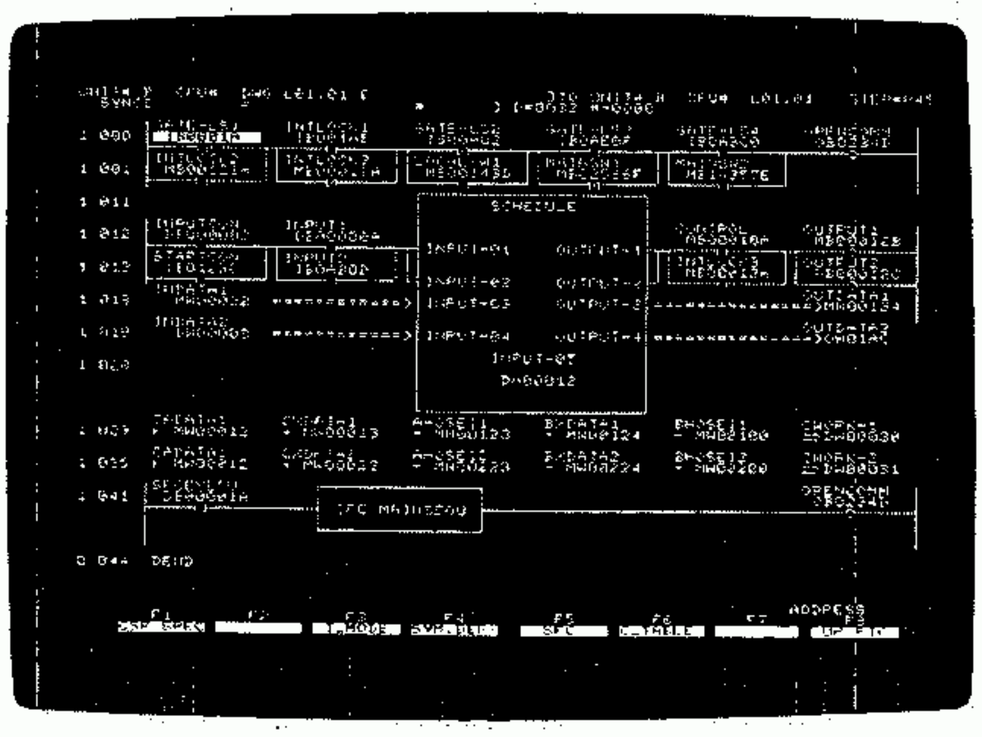
8.12 PROGRAMMING EXAMPLES

(1) DWG program

A DWG program is created by using a ladder diagram and text language.

The programming panel provides a 50-line by 80-character display.

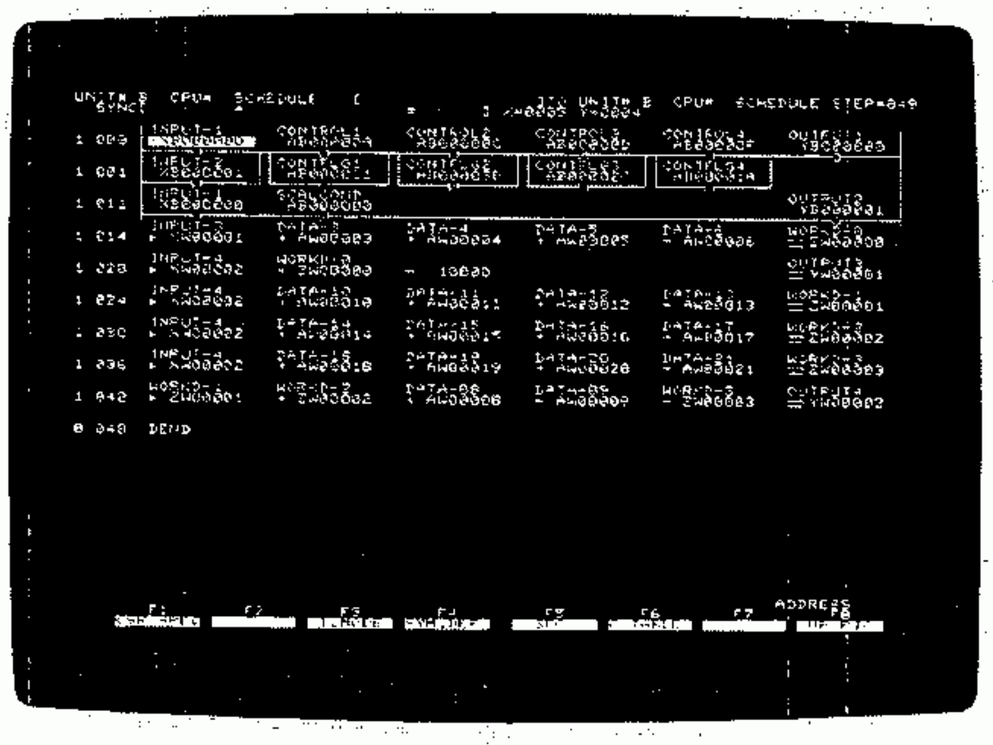
The numerals represent the syntax depth and the step number.



690-4

(2) Function program

A function program is created using a ladder diagram and text language as for DWG program.



690-5

9 SCAN SETTING

This section describes the scan processing for program execution and gives the precautions to be taken.

CONTENTS

9.1 PROGRAM EXECUTION

(1) Basic drawing execution

Basic drawings are executed based on the individual priority and execution condition, as shown in Fig. 9.1.

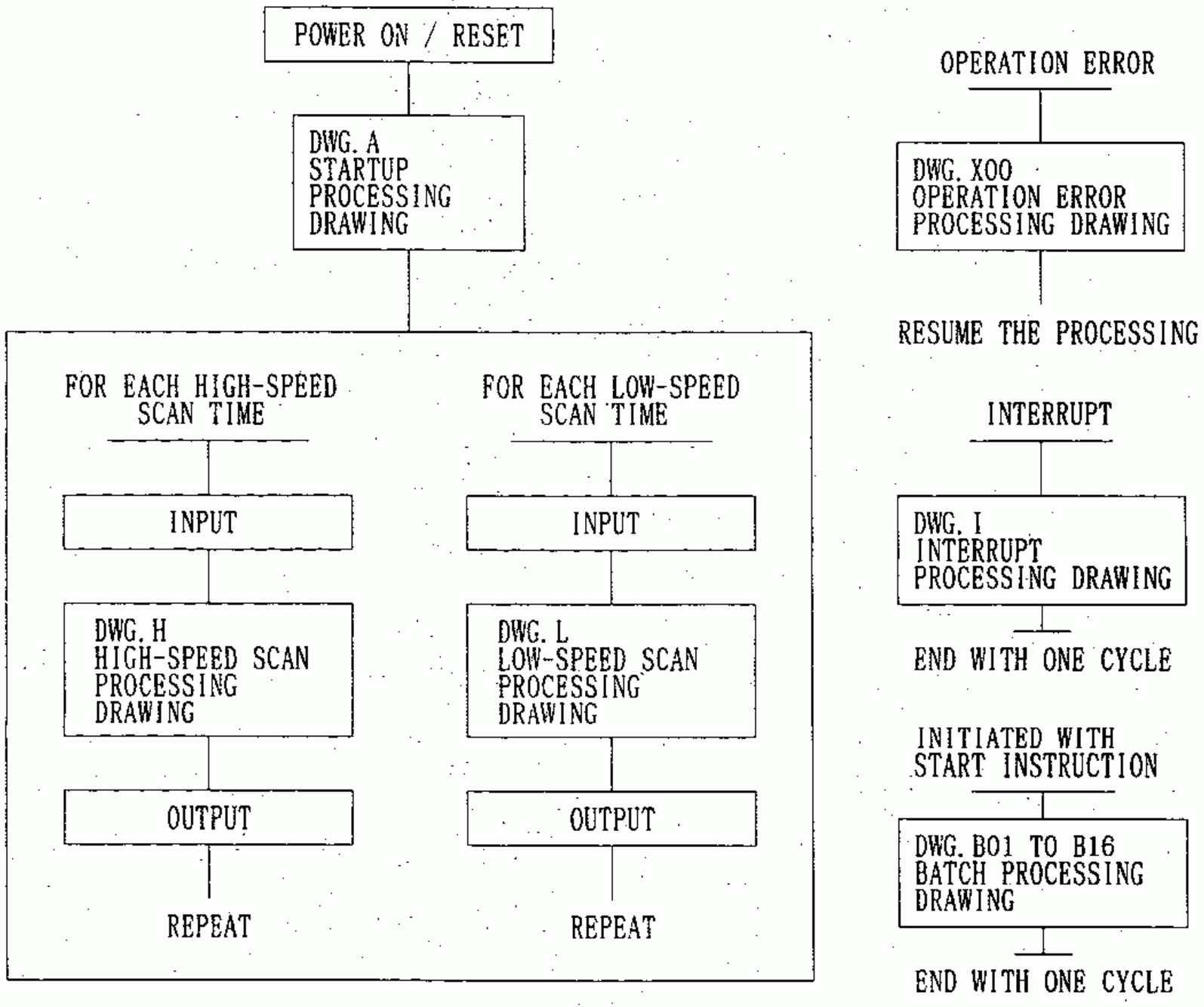


Fig. 9.1 Basic drawing execution

(2) Scan processing drawing execution scheduling

Scan processing drawings are not executed simultaneously, but scheduled according to their priority as shown in Fig. 9.2.

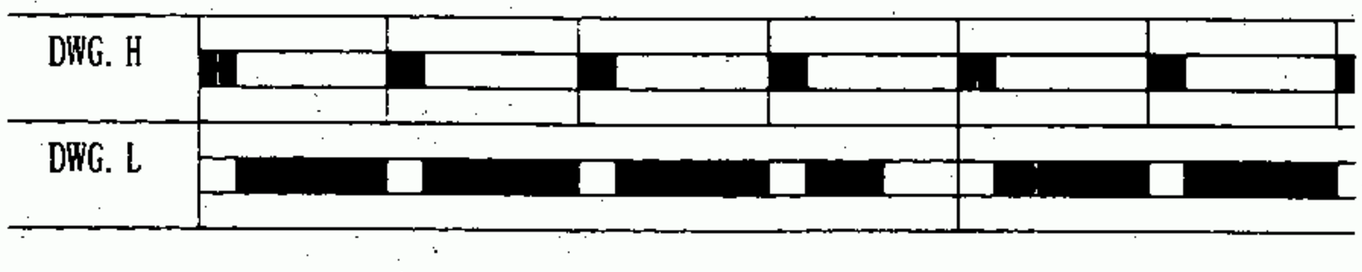


Fig. 9.2 Execution scheduling of scan processing drawings

9.2 PRECAUTIONS ON SETTING THE SCAN TIME

High-speed and low-speed scan times are set with a programming panel. Note the following when setting the times:

(a) The scan time should not be less than 3 ms and not more than 300 ms.

(b) Make the low-speed scan time an integral multiple of the high-speed scan time unless otherwise dictated.

If not, observe the following condition:

High-speed scan time + low-speed scan time ≦ 500 ms (If this condition is not satisfied, the watchdog timer may operate to cause the CPU to go down.)

10 INPUT/OUTPUT ALLOCATION

This section describes usage of CP-3300-provided I/O functions and allocation of I/O registers.

The I/O functions comprise local I/O, remote I/O, CP-213 transmission, and link transmission.

CONTENTS PAGE

The CP-3300 I/O functions are grouped by the signal transmission method into the following :

- Local I/O
- Remote I/O
- CP-213 transmission
- Link transmission

Their allocation is explained below.

Refer to the Control Pack CP-3300 Programming Panel Operator's Manual for the actual operation of I/O allocation from the programming panel.

10.1 LOCAL I/O

The local I/O consists of a maximum of five racks as shown in Fig. 10.1. The I/O modules in each rack have a free location. Namely, an I/O module can be inserted in any slot on any rack. Normally, 2000 series I/O modules are used.

A COMM module, 213IF module, etc. when used are inserted in slots 1 to 6 on rack 1. Consequently, in this case, the number of local I/O modules that can be inserted on rack 1 is limited.

The capacity of local I/O is 512 words each for input and output.

Input : IW0000 to IW01FF (input variables)
Output : OW0000 to OW01FF (output variables)

I/O allocation is performed by associating the I/O module location (rack number, slot number) with the input or output variable. In actual allocation, the I/O modules are classified into discrete and register. The number of bits must be set up in the discrete and the number of words in the register.

The I/O operation is synchronized with high- or low-speed scanning. This scan mode must also be specified.

Fig. 10.1 Local I/O configuration

10.2 REMOTE I/O

The remote I/O consists of a maximum of 31 stations from station 1 to 31. Each station consists of up to four racks as shown in Fig. 10.2. The I/O modules on each rack have a free location. Namely, an I/O module can be inserted in any slot on any rack. Normally, 2000 series I/O modules are used.

The capacity of remote I/O is 512 words each for input and output.

Input : IW0200 to IW03FF (input variables)

Output : OW0200 to OW03FF (output variables)

I/O allocation is performed by associating the I/O module location (station number, rack number, slot number) with the input or output variable. In actual allocation, the I/O modules are classified into discrete and register. The number of bits must be set up in the discrete and the number of words in the register.

The I/O operation is synchronized with high- or low-speed scanning. This scan mode must also be specified.

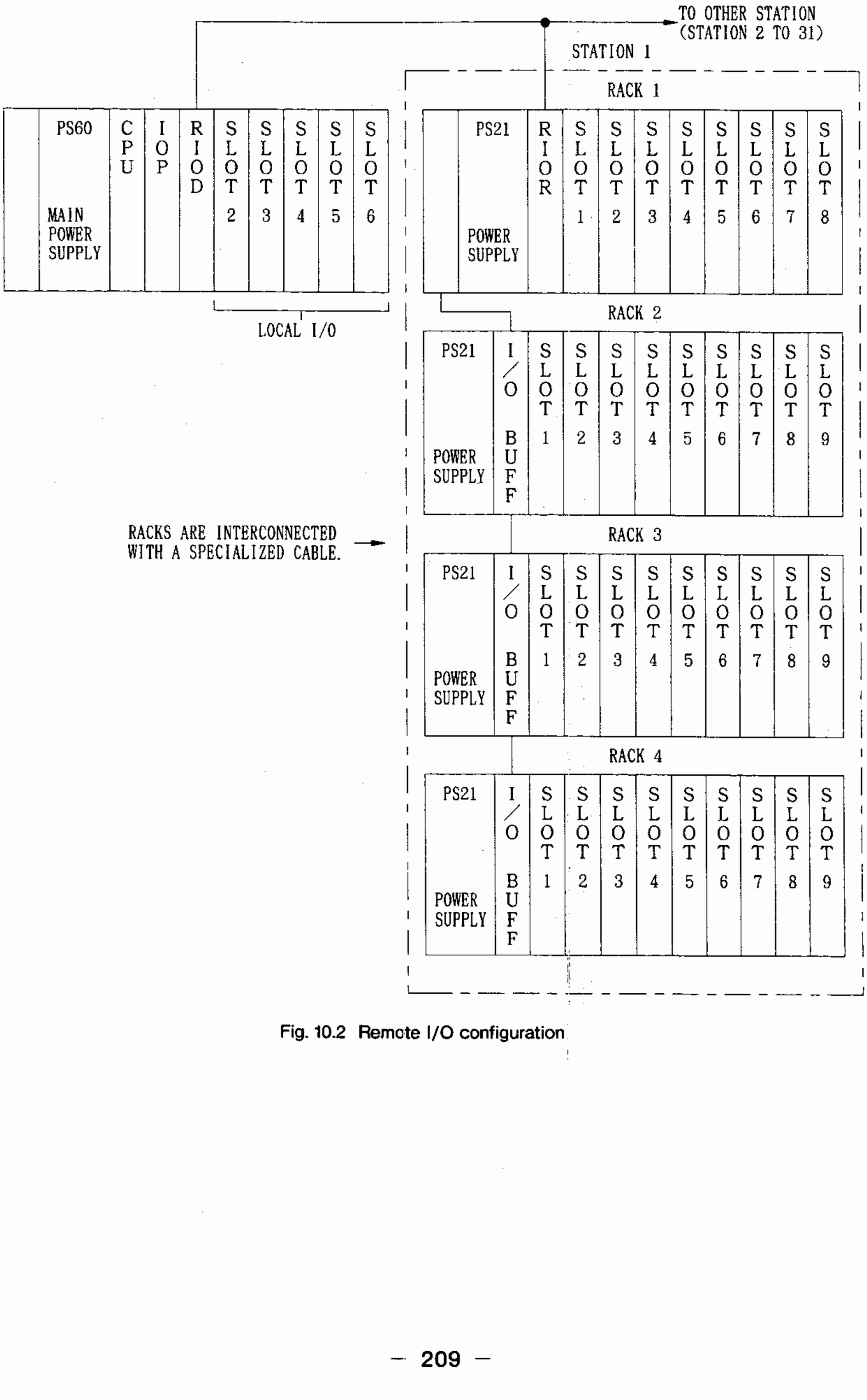


Fig. 10.2 Remote I/O configuration

10

10.3 CP-213 TRANSMISSION

Using a 213IF module allows you to connect I/O equipment provided with a CP-213 transmission port. These include the following :

- CP-313M series ····· CP-313M, CP-313MA, CP-313MB, CP-313MC, CP-313MD
- CP-813 series ····· RIO-01M, DMC-02M
- CP-815 ····· Equipment with BCM-815M mounted as a transmission interface module
- VS-676
- VS-686TYM

Up to four 213IF modules can be inserted into the CPU rack. A total of four lines can be served for each module line. The transmission capacity is 512 words per line for each of input and output. Therefore, a total for four lines is 2 k words for each of input and output. The registers used are

Input variables : IW0400 to IW0BFF
Output variables : OW0400 to OW0BFF

Up to 63 stations (1 to 63) can be connected to a line. Station number 0 is not allowed.

I/O allocation is performed by associating the line number and station number to the CP-213 station. As CP-213 station details, the top I/O register, the number of words in this register, and the device name are specified.

There are three types of transmission repeat periods to be selected according to the usage :

- Synchronizing with high-speed scan
- Synchronizing with low-speed scan
- Not synchronizing with high-speed nor low-speed scan, but repeating for CP-213 itself

Some equipment with a CP-213 port requires initial data. Transmission of initial data is performed using a built-in function ISET-213.

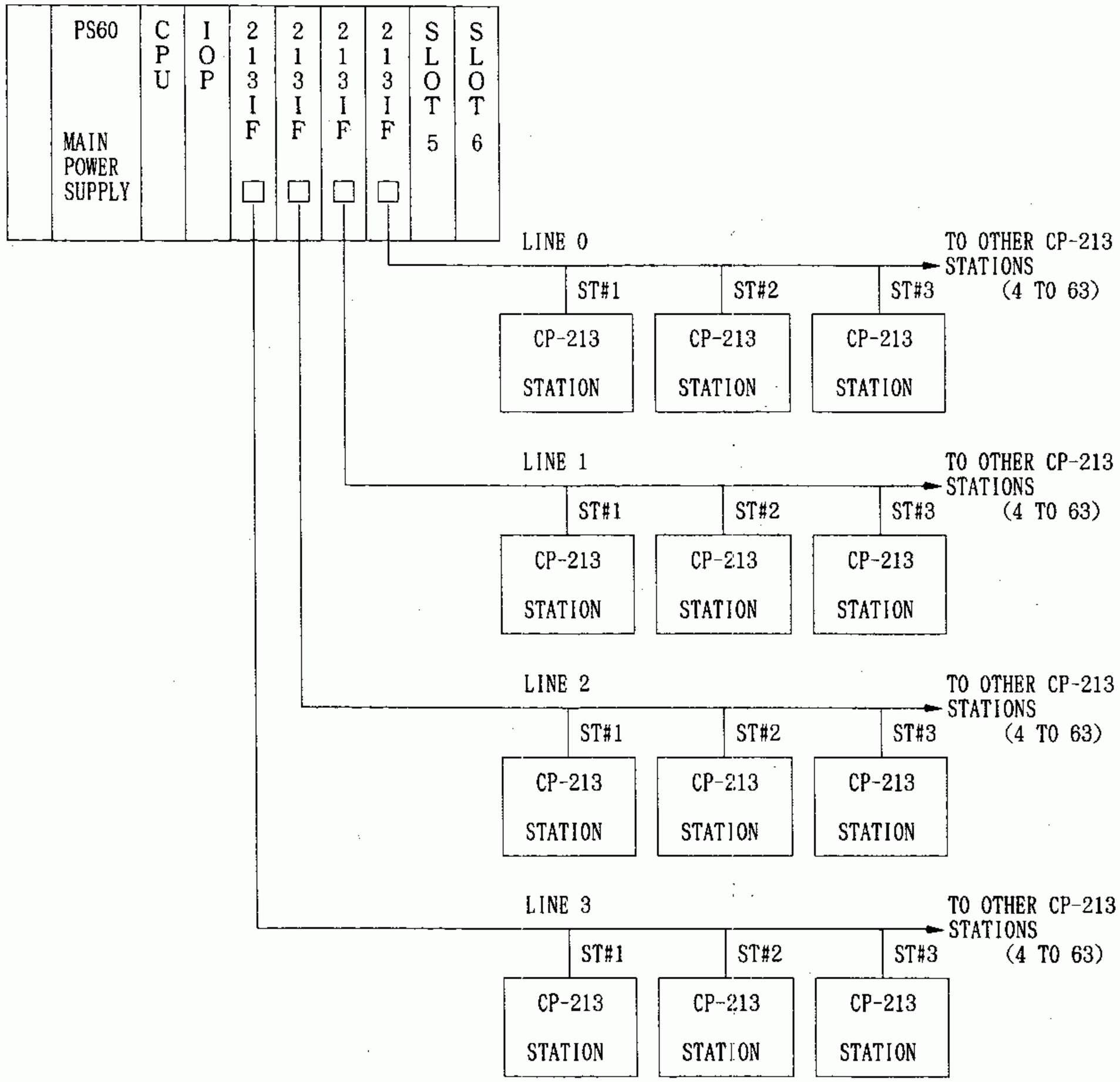


Fig. 10.3 CP-213 transmission configuration

10.4 LINK TRANSMISSION

Using a LINK module allows you to perform data I/O with CP-3500, CP-5500, and GL60S. Only one LINK module can be inserted into the CPU rack.

The transmission capacity is 1k words for each of input and output.
Input variables : IW0C00 to IW0FFF
Output variables : OW0C00 to OW0FFF

Up to 32 stations (1 to 32) can be connected to a line.

The I/O is allocated by associating the station number with the link station.

There are two types of I/O data : discrete and register. The number of bits is set up in the discrete and the number of words in the register. The maximum number of words of an output register is 256.

The input and output operation is executed in synchronizing with high-speed or low-speed scan. This scan mode must also be set up.

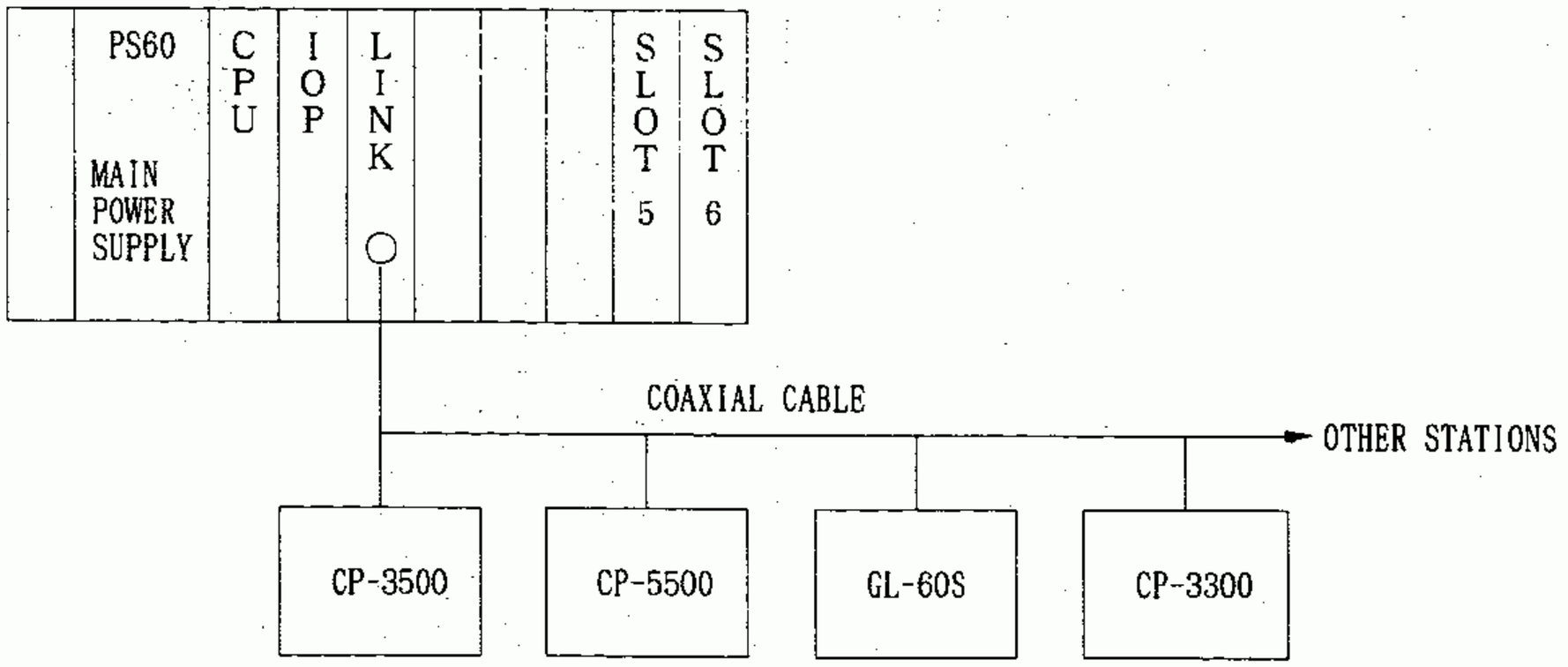


Fig. 10.4 Link transmission configuration

10.5 YENET TRANSMISSION

Using a 3200IF module allows you to execute data I/O for CP-3300, GL60S, CP-3500 or CP-5500. (For CP-3300 and CP-5500, data I/O is executed via network service unit.)

Up to 126 stations combining 3200IF unit and network service unit can be inserted into a network. This network is also connected to other networks via a bridge on the line.

I/O operation is executed by system functions such as MEMOBUS protocol and general-purpose protocol.

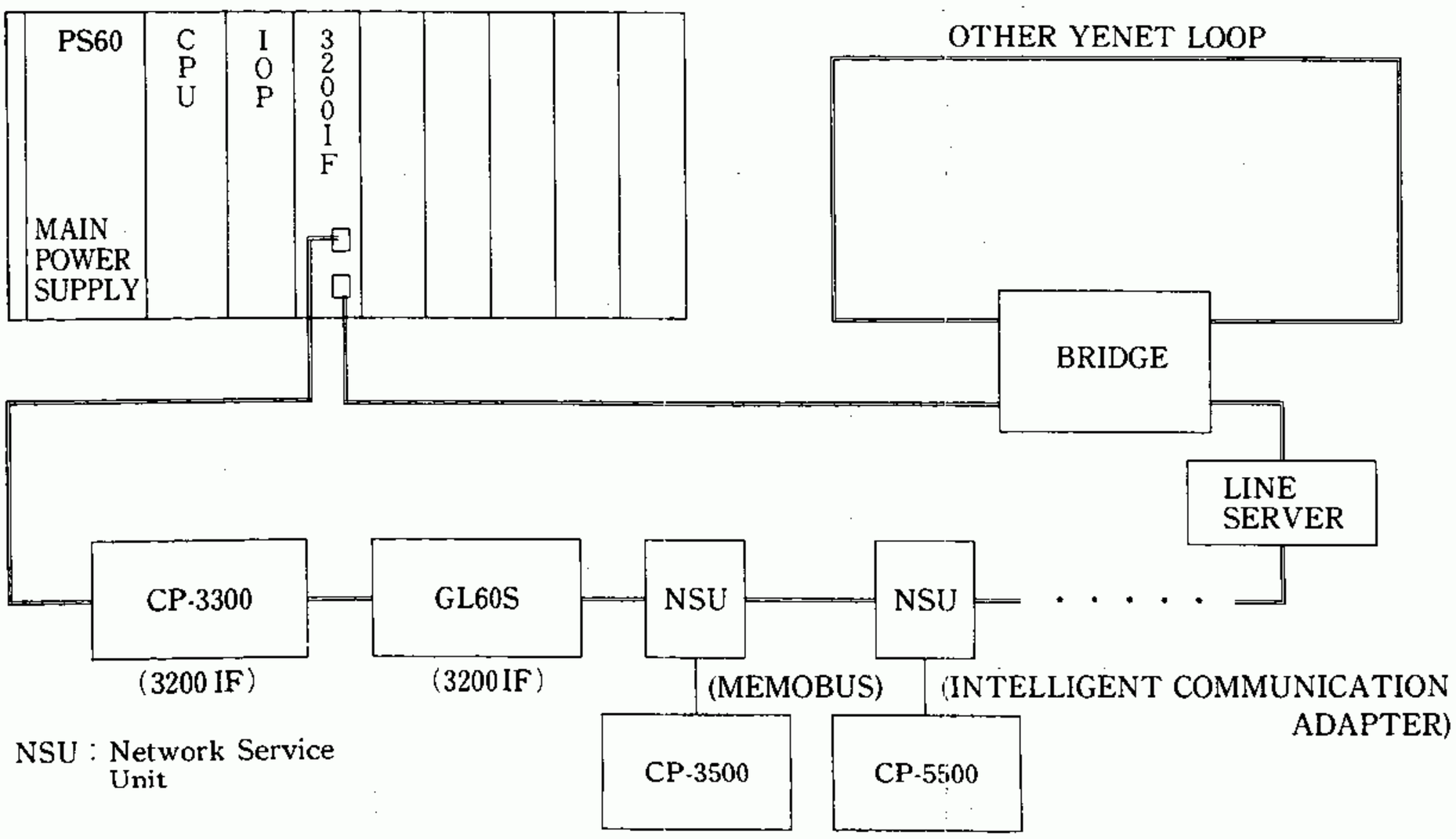


Fig. 10.5 YENET transmission configuration

11 TRANSMISSION INTERFACE

This section describes the functions and uses of the transmission interface of CP-3300.

The transmission interface comes in three types : MEMOBUS, FABUS-II, and ASCII transmission.

CONTENTS PAGE

11.1 MEMOBUS TRANSMISSION INTERFACE

The CP-3300 can be used as a MEMOBUS master/slave to perform communication through an RS-232C communication port of the IOP and/or COMM module. For communication as a MEMOBUS master, a system standard function COMM is used. For communication as a slave, a transmission definition defined from the programming panel is used. The IOP or COMM module has two communication ports. Since the CP-3300 can use up to two IO/COMM modules, maximum four ports can be used per system for MEMOBUS transmission.

The transmission status at the time the MEMOBUS master operates is stored in the COMM function parameter COMM STATUS (PARM01). The transmission status during slave operation is stored in SW00670 to SW00685.

11.2 FABUS-II TRANSMISSION INTERFACE

Using a LINK module allows you to perform communication through an FABUS-II transmission with CP-3500, CP-5500, CP-3300, and GL60S. The FABUS-II transmission has message and link transmission functions. See section 10 for link transmission. GL60S can be connected only by link transmission. Up to 32 stations can be connected on a transmission line.

Two types of transmission procedures are available: MEMOBUS and universal data protocol. A system standard function LINK-MST/LINK-SLV function is used in a master/slave operation under the MEMOBUS protocol and a LINK-SND/LINK-RCV in a master/slave operation under the universal data protocol. The universal data protocol can be use only in the CP-3500 and CP-3300.

11.3 ASCII TRANSMISSION INTERFACE

Using RIOD and ASCII modules allows you to connect equipment under RS-232C (called ASCII devices) such as a serial printer, CRT terminal, keyboard, and bar code reader. A system standard function WRIT is used for output equipment such as a serial printer and CRT terminal and a READ function for input equipment such as a keyboard.

The ASCII module has two ASCII ports. The CP-3300 can connect up to eight ASCII modules for each RIOD line. Therefore, the system as a whole can connect a maximum of 32 ASCII devices using two RIOD lines.

11.4 YENET TRANSMISSION INTERFACE

Using a 3200IF module allows you to perform communication through a YENET transmission with CP-3300, GL60S, CP-3500*, and CP-5500*. Up to 126 stations can be connected in a network.

Two types of transmission procedures are available

- MEMOBUS protocol
  A system standard function NET-MBUS is used in a master operation. No function is used in a slave operation because the response is made automatically.
- Universal data protocol (valid between 3200IF modules only)
  Uses functions NET-PEER, NET-BROD, NET-BOOK, NET-POLL, NET-SND and NET-RCV.
  A function NET-DIAG is also used for network self-diagnosis.

* When CP-3500 and CP-5500 is used, YENET transmission can be perform via a network service unit having two MEMOBUS ports. The ports can be set to master or slave. In this case, only MEMOBUS protocol is used as transmission procedure.

12 ERROR HANDLING

This section describes handling of operation errors produced in user programs and errors produced during I/O execution.

This section also explains the function of an operation error handling drawing performed following an operation error and the content of the handling.

CONTENTS PAGE

12.1 OPERATION ERRORS HANDLING

If an integer operation error (underflow, overflow, and division error) or a real operation error (integer store, real store, division by zero, etc.) occurs in a user program (DWG, function), the operation error handling drawing (DWG. X00) for the DWG in question is executed. Then, the execution of the original user program is resumed by using a value specified in the operation error handling drawing.

If a relevant operation error handling drawing is not found, the original user program execution is continued by using a system-defined value (default).

(1) Gathering operation error data

When an operation error occurs, error data shown in Tables 12.1 and 12.2 are gathered and stored in a relevant variable area. The error data shown in Table 12.2 are stored only if the relevant operation error handling drawing exists and the required variable area (D variable area) inside DWG is allocated.

(2) System default handling

If a relevant operation error handling drawing is not found or a necessary variable area inside the DWG is not allocated, or an error cannot be handled, system-defined default processing as shown in Table 12.3 is performed and then the user program execution is continued.

(3) Real operation error causes

In real number operation, an underflow, overflow, divide-by-zero error (X/0, 0/0), etc. occurs when storing data to an F register integer type variable or real type variable. An operation error also occurs when an operation with a nonnumeric value, infinity, or de-normal is performed as shown in Table 12.4.

(4) Processing in the operation error handling drawing

In the operation error handling drawing, the user should check the error code stored in DW00001 and store a value to be substituted in DW00003/DW00006/DF00006.

Fig. 12.1 shows a sample program of operation error handling drawing.

DW00003/DW00006/DF00006 stores values (see Table 12.3) defined by the system before execution of an operation error handling drawing. If no numeric value is stored in DW00003/DW00006/DF00006 on the operation error handling drawing, the same result as in the program shown in Fig. 12.1 will be obtained.

Table 12.1 Operation error status (system variable area)

No.	Name	Data address	Symbol	Remarks
1	DWG. A error count	SW00080		Error code: see Table 12.3.
	error code	SW00081		
2	DWG. I error count	SW00082		
	error code	SW00083		
3	DWG. H error count	SW00084		
	error code	SW00085		
4	(System reserved)	SW00086		
		SW00087		
5	DWG. L error count	SW00088		
	error code	SW00089		

Table 12.2 Operation error status (DWG internal variable area)

No.	Name	Data address	Symbol	Remarks
0	Error count	DW00000		
1	Error code	DW00001		
2	Error A register	DW00002		
3	Change A register	DW00003		Error DWG No. Basic drawing: C800H Detailed drawing: xx00H Expanded drawing: xxyyH xxH: detailed drawing number yyH: expanded drawing number Add 64H for a batch detailed drawing
4	Error F register	DW00004		
5		DW00005		
6	Stored integer value (1 wd) Stored real value (2 wds)	DW00006		
7	Change F register (2 wds)	DW00007		Error DWG #step DWG function reference step for an operation error inside a function
8	Error IP	DW00008		
9	Error CS	DW00009		
10	Error DWG #step	DW00010		
11	Error DWG No.	DW00011		

Table 12.3 (a) Operation error codes and system defaults

	Err-Code	Error	User	System default
Integer operation	H0001	Underflow in integer operation	○	-32768 〔-32768〕 *
	H0002	Overflow in integer operation	○	+32767 〔32767〕 *
	H0003	Division error in integer operation	○	Error itself 〔A Reg.〕 *
	H010X	Integer operation error inside operation error drawing	×	Default above
Real operation	H0010	Non-numeric value error in integer store	○	Store not executed 〔00000〕 *
	H0011	Underflow in integer store	○	Store not executed 〔-32768〕
	H0012	Overflow in integer store	○	Store not executed 〔+32768〕
Real operation	H0021	Underflow in real store	○	Store not executed 〔-1.0E+38〕 *
	H0022	Overflow in real store	○	Store not executed 〔1.0E+38〕 *
	H0023	Division-by-zero error in real operation	○	Operation not executed 〔F Reg.〕 *
Real operation	H0030	Invalid operation (non-numeric value) in real operation	×	Operation not executed
	H0031	Exponent underflow in real operation	×	0.0 (F register)
	H0032	Exponent overflow in real operation	×	Max. value (F register)
	H0033	Division error in real operation (non-numeric value, 0/0)	×	Operation not executed
	H0034	Exponent underflow in real store	×	0.0 stored
Real operation	H0040 to H0056	Operation error in real type system function H 0040: SQRT H 0041: SIN H 0042: COS H 0043: TAN H 0044: ASIN H 0045: ACOS H 0046: ATAN H 0047: EXP H 0048: LN H 0049: LOG H 004A: DZA H 004B: DZB H 004C: LIM H 004D: PI H 004E: PD H 004F: PID H 0050: LAG H 0051: LLAG H 0052: FGN H 0053: IFGN H 0054: LAU H 0055: SLAU H 0056: REM Add H1000 or H2000 for an error.	×	Operation stopped and output = 0.0 made (F register)

* The value in 〔 〕 is set in DW00003/DW00006/DF00006 by the system as a default before execution of a user operation error drawing.

Table 12.3 (b) Error codes at occurrence of index errors

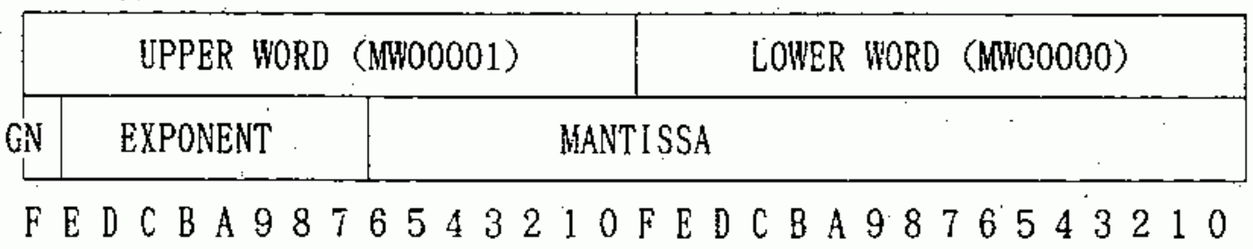
Err-Code	Error	User	System default
H1000	Index error in the DWG	×	Reexecute with i, j = 0
H2000	Index error in the function	×	Reexecute with i, j = 0
HX060 to HX077 X: 1 or 2	Index error in the integer type system function	×	Operation stopped. A register remains unchanged.
	H X060: START H X061: IN H X062: OUT H X063: BIN H X064: BCD H X065: PARITY H X066: ROTL H X067: ROTR	H X068: MOVB H X069: MOVW H X06A: XCHG H X06B: DZA H X06C: DZB H X06D: LIM H X06E: PI H X06F: PD	H X070: PID H X071: LAG H X072: LLAG H X073: FGN H X074: IFGN H X075: LAU H X076: SLAU H X077: SMOV

Note: When an index error occurs, the system temporarily sets the index value (i, j) to 0 and continues the operation. This condition (i=j=0) lasts until the indexed numeric value operation is switched to indexed relay operation or vice versa. The pre-error value of index register (i, j) is retained, but can be changed in the operation error handling drawing.

Table 12.4 Real value range and operation error

	Numeric value class	Sign (1 bit)	Exponent (8 bits)	Mantissa (23bits)	Remarks
Positive	Non-numeric value	0 to 0	11111111 to 11111111	1111111·····111111 to 0000000·····000001	Operation error cause
	Infinity	0	1111111	0000000·····000000	Operation error cause
	Normal	0 to 0	11111110 to 00000001	1111111·····111111 to 0000000·····000000	+3.402823 E + 38 to +1.175494 E - 38
	De-normal	0 to 0	00000000 to 00000000	1111111·····111111 to 0000000·····000001	Operation error cause
	Zero	0	00000000	0000000·····000000	+0.000000
Negative	Zero	1	00000000	0000000·····000000	-0.000000
	De-normal	1 to 1	00000000 to 00000000	0000000·····000001 to 1111111·····111111	Operation error cause
	Normal	1 to 1	00000001 to 11111110	0000000·····000000 to 1111111·····111111	-1.175494 E-38 to -3.402823 E+38
	Infinity	1	11111111	0000000·····000000	Operation error cause
	Non-numeric value	1 to 1	11111111 to 11111111	0000000·····000001 to 1111111·····111111	Operation error cause

INTERNAL FORMAT OF A REAL NUMBER



```
UNIT#1 CPU#1 DWG L00
 ├─  DW00001 < 00016                    : ANY INTEGER OPERATION ERROR?
IFON                                    : IF ANY
 ├─  DW00001 = 00001                    : IS IT AN UNDERFLOW?
   〔├─ -32,768 〕                       : IF SO (UNDERFLOW), SET -32,768.
            = 00002                     : OR AN OVERFLOW?
   〔├─ 32,767〕                         : IF SO (OVERFLOW), SET 32,767.
            = 00003                     : OR A DIVISION ERROR?
   〔├─ DW00002〕            => DW00003  : IF SO (DIVISION ERROR), REPLACE THE CONTENT
                                          WITH THAT OF DW00002.
                                        : UPDATE WITH THE ABOVE VALUE.


ELSE                                    : IF NOT AN INTEGER OPERATION ERROR, IT WILL
                                          PROBABLY BE A REAL OPERATION ERROR.
 ├─  DW00001 = 00016                    : A NON-NUMERIC VALUE ERROR DURING INTEGER STORE?

   〔├─ 00000〕                          : IF SO (NON-NUMERIC VALUE ERROR), SET 0.
            = 00017                     : OR AN UNDERFLOW DURING INTEGER STORE?
   〔├─ -32,768〕                        : IF SO (UNDERFLOW), SET -32,768.
            = 00018                     : OR AN OVERFLOW DURING INTEGER STORE?
   〔├─ -32,767〕            => DW00006  : IF SO (OVERFLOW), SET 32,767.
                                        : UPDATE WITH THE ABOVE VALUE.
 ├─  DW00001 = 00033                    : IF AN UNDERFLOW DURING REAL NUMBER STORE,
   〔├─ -1.000000E + 38〕  〔=>DF00006〕  : CHANGE WITH -1.000000E + 38.
 ├─  DW00001 = 00034                    : IF AN OVERFLOW DURING REAL NUMBER STORE,
   〔├─ 1.000000E + 38〕  〔=>DF00006〕   : CHANGE WITH 1.000000E + 38.
 ├─  DW00001 = 00035                    : IF A DIVISION ERROR IN REAL NUMBER OPERATION,
   〔├─ DF00004〕  〔=>DF00004〕          : CHANGE WITH THE CONTENT OF DF00004.
IEND
DEND
```

Fig. 12.1 Sample program of operation error handling drawing

12.2 I/O ERROR HANDLING

The CP-3300 has two I/O modes, namely system I/O and direct I/O. The system I/O is synchronized with scanning. It is applied to I/O modules specified by I/O allocation.

On the other hand, in the direct I/O, target I/O modules are determined by coding an IN and an OUT instruction in the user program (DWG). The I/O is executed when the instruction (IN, OUT) is executed. The direct I/O is valid only for the local I/O.

If an error occurs in the I/O, the system is involved differently depending on the I/O mode.

(1) Errors in system I/O

If an error occurs in the execution of system I/O, "system I/O status" (SW00100 to SW00655) shown in Table 1.10 of Appendix 1 is set. More precisely, the result of system I/O is recorded. The user can learn from this status information whether the I/O had an error. Moreover, this information can be used to have I/O error handling performed in the user program.

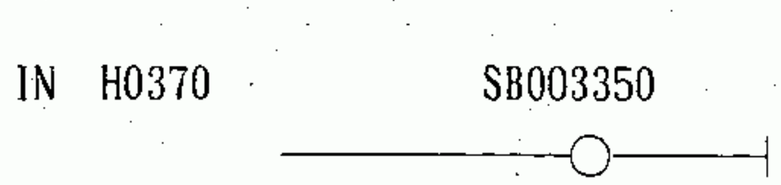
When the system input has an error, the current input data retains the previous value if the input in the previous scan is normal. (Holding the previous value) The input data is set to 0 when an error occurred previously.

(2) Errors in direct I/O

If an error occurs in the cxecution of an IN, OUT instruction, the system performs nothing except to set the B register to ON. (The register is set to OFF if no error occurs.)

The user can learn the presence or absence of an error by checking the content of B register just after execution of the IN or OUT instruction.

The following program outputs data to a 16-bit discrete output module mounted on slot 7, rack 3 and sets SB003350 if there is an error:



13 SYSTEM PROCESSING

This section describes the system processing at the startup and stop of the CP-3300.

The system processing is selected by setting the switches on the CPU module front.

CONTENTS PAGE

13

13.1 SYSTEM PROCESSING AT STARTUP AND STOP

Fig. 13.1 shows the flowchart of CP-3300 CPU module operation.

The following describes the CPU operation, centering on the startup and stop, according to Fig. 13.1.

(1) Offline self diagnosis

With DIAG on switch 1SW on the CPU front turned ON, turning power the ON causes the CPU to enter the offline self diagnostic mode. It will not execute a user program.

In this mode, the following diagnoses are repeated and the results stored in system variables SW00071 and SW00072.

- Memory (RAM) read/write diagnosis
- System program (ROM) diagnosis
- Main processor functional diagnosis
- Numeric coprocessor functional diagnosis
- Boolean coprocessor functional diagnosis

To move from the offline self diagnostic mode to normal operation mode, turn OFF power, turn OFF the DIAG switch, and then turn ON power again.

(2) Memory initialization

If DIAG on switch 1SW on the CPU module front is OFF and X3 ON, the CPU initializes all program memory and data memory areas and then executes the following self diagnostic items:

- Memory (RAM) read/write diagnosis
- System program (ROM) diagnosis
- Main processor functional diagnosis
- Numeric coprocessor functional diagnosis
- Boolean coprocessor functional diagnosis

If the result of self diagnosis is acceptable, check the setting of RUN switch on 1SW. Subsequent operation is the same as in the startup of new or continued operation.

(3) Startup of new operation

If the setting of switch 1SW on the CPU module front and the operation status prior to power failure satisfy the following conditions, startup of new operation is performed:

(DIAG = OFF) and (X3 = OFF) and

{ (NEW = ON) or (RUN = OFF), or (operation preceding the power failure is "stopped state")}

In the new operation startup, immediately after power-on, self diagnosis is performed and the RUN switch setting is checked. If the operation prior to the power failure is "stopped state" due to a major malfunction processing will stall unless the ERST switch on 1SW is turned ON for reset.

The operation after this self diagnosis is the same as in the startup of continued operation. See the explanation below of startup of continued operation for details.

(4) Startup of continued operation

The startup of continued operation takes place only if all the following conditions are satisfied:

- DIAG = OFF
- X3 = OFF
- NEW = OFF
- RUN = ON
- Operation preceding the power failure = Scanning being executed normally

In the startup of continued operation, after power-on, the rest of scanning executed at the time of power off is executed.

Then the following self diagnosis is executed:

- Main processor functional diagnosis
- Numeric coprocessor functional diagnosis
- Boolean coprocessor functional diagnosis

If, after the above self diagnosis, no error is detected, check the setting of the RUN switch. If the RUN switch is OFF, wait until it goes ON. In the startup of continued operation, the RUN switch cannot actually be OFF. This is because it makes sense only in the startup of new operation and memory initialization mode.

If the RUN switch is ON or changes from OFF to ON, the CPU starts up the watchdog timer, and executes DWG. A.

At completion of DWG. A execution begins scanning, and RUN indicaton lamp goes ON. The first scanning is executed after a passage of high-speed and low-speed scan time following the end of DWG. A. The system input is executed from the first scan, but the system output executed from the fourth scan (if the startup of new operation selected) or second scan (if continued operation). This is designed to avoid control inconsistency due to a reverse order of the sequence.

(5) Stop

The CP-3300 CPU stops its operation in the following cases:

- Power is cut off.
- A power failure occurred.
- The RUN switch on 1SW on the CPU module front is turned OFF.
- A major malfunction occurred.

Once the CPU stops its operation, it will not start up unless power is turned on again. If it stops due to a major malfunction, steady operation status will not be obtained unless the ERST switch on 1SW on the CPU module front is turned ON.

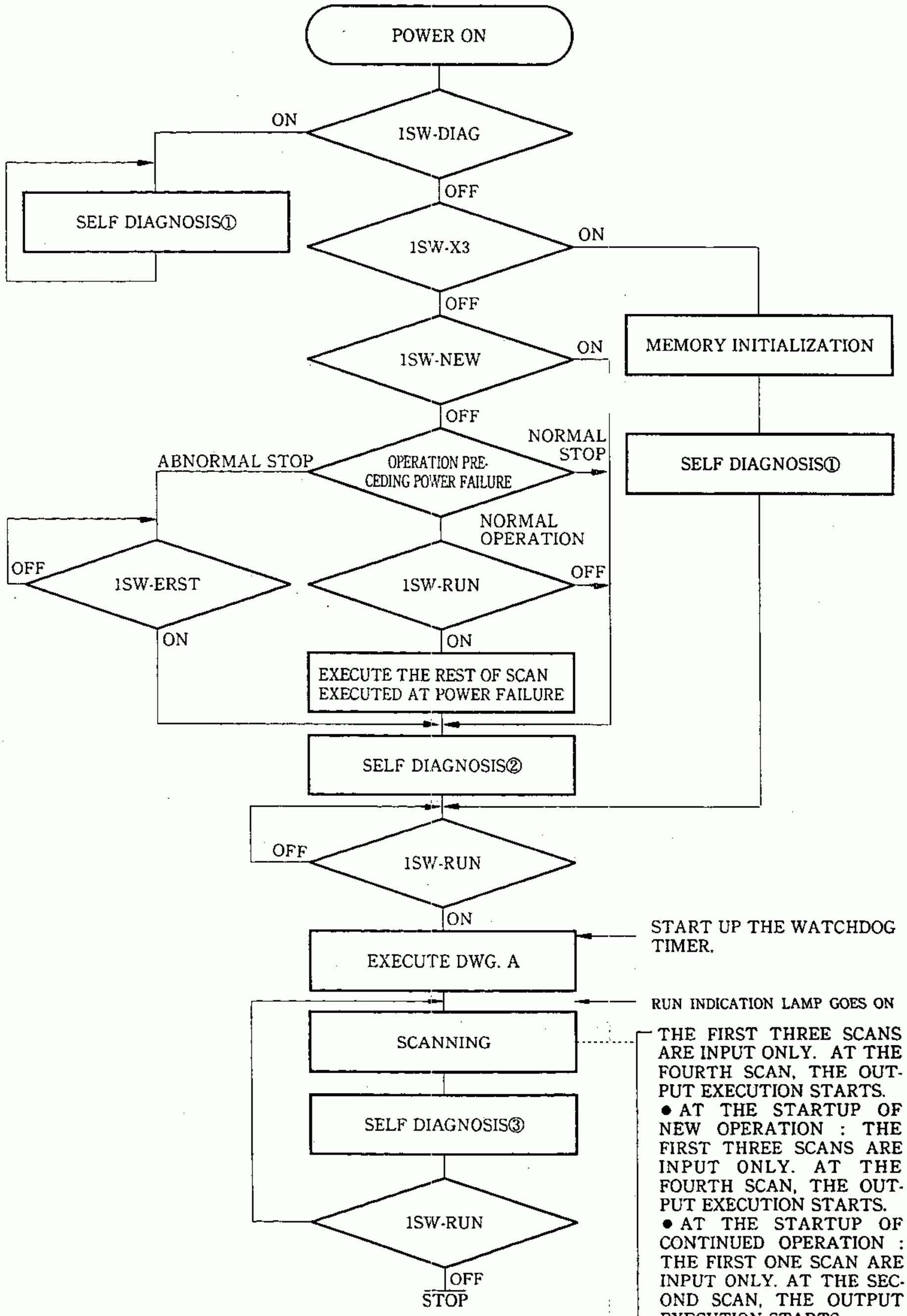


Fig. 13.1 CPU Module Operation Flow Diagram

APPENDIXES

Appendix 1 lists the data memory structure, especially the contents of system registers.

Appendix 2 illustrates the addresses where I/O variables are allocated.

Appendix 3 explains the operations of system functions and I/O parameters.

CONTENTS PAGE

CONTENTS (Cont'd)

APPENDIX 1 DATA MEMORY STRUCTURE

1.1 DATA MEMORY ALLOCATION

Table 1.1 Data memory allocation

Register number	Register area	Data reference
SW00000 to SW00767	System register, 768 words	SBnnnnnx SWnnnnn SFnnnnn nnnnn : decimal
IW0000 to IWOFFF	Process input register (4k words) • Local IW0000 to IW01FF • Remote IW0200 to IW03FF • CP-213 IW0400 to IWOBFF • Link IWOC00 to IWOFFF	IBhhhhx IWhhhh IFhhhh hhhh : hexadecimal
OW0000 to OWOFFF	Process output register (4k words) • Local IW0000 to IW01FF • Remote IW0200 to IW03FF • CP-213 IW0400 to IWOBFF • Link IWOC00 to IWOFFF	OBhhhhx OWhhhh OFhhhh hhhh : hexadecimal
MW00000 to MW16383	DWG common register, 16k words	MBnnnnnx MWnnnnn MFnnnnn nnnnn : decimal

1.2 SYSTEM REGISTER ALLOCATION

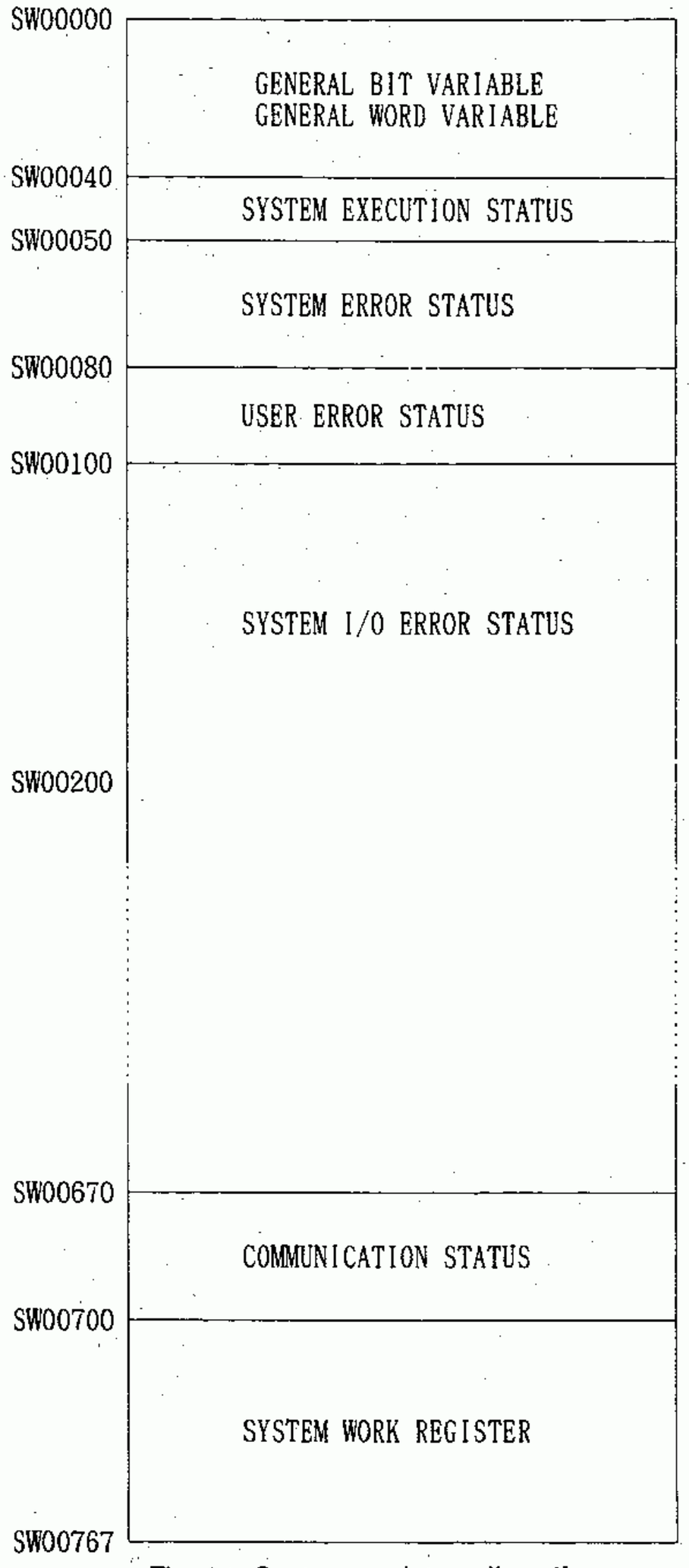


Fig. 1.1 System register allocation

Table 1.2 General bit variables (common to DWG)

No.	Name	Data address	Symbol	Remarks
0	Resume	SB000000		Startup type (hold type signal) ON : continued OFF : new start
1	First scan (high-speed)	SB000001		ON for only one scan pass after start
2				
3	First scan (low-speed)	SB000003		ON for only one scan pass after start
4	Constantly-ON relay	SB000004		
5	Dual synchronous operation	SB000005		
6	Short break detection	SB000006		Short break detected during operation
7	Voltage drop of back-up battery	SB000007		
8				
9				
10				
11				

Table 1.3 General bit variables (specialized to DWG. H)

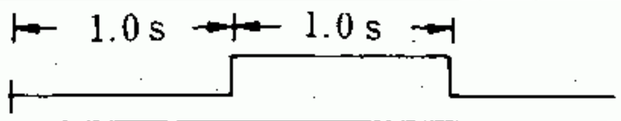
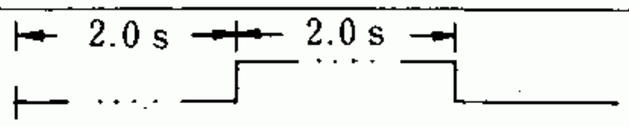
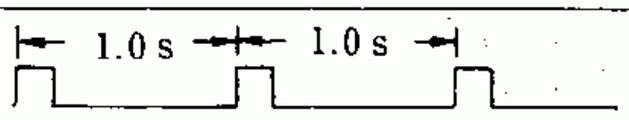
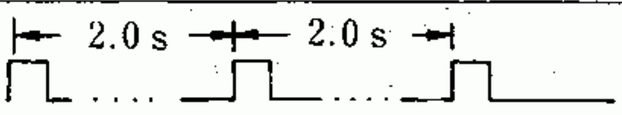
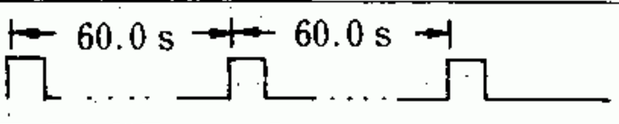
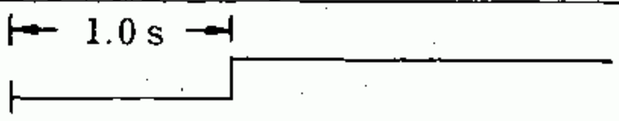
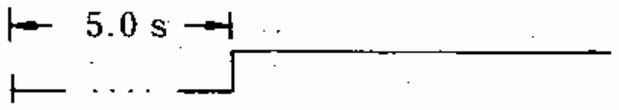
No.	Name	Data address	Symbol	Remarks
0	1 Scan Blinking relay	SB000010		
1	0.5s Blinking relay	SB000011		
2	1.0s Blinking relay	SB000012		
3	2.0s Blinking relay	SB000013		
4	0.5s Sampling relay	SB000014		
5	1.0s Sampling relay	SB000015		
6	2.0s Sampling relay	SB000016		
7	60.0s Sampling relay	SB000017		
8	Power on 1.0s-delayed relay	SB000018		
9	Power on 2.0s-delayed relay	SB000019		
10	Power on 5.0s-delayed relay	SB00001A		
11				

Table 1.4 General bit variables (specialized to DWG. L)

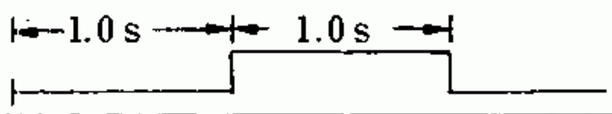
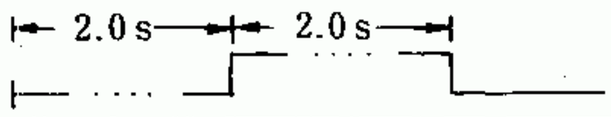
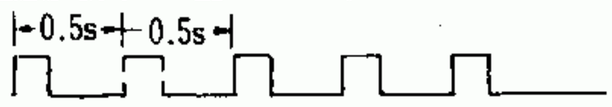
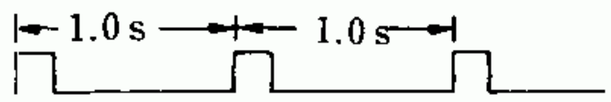
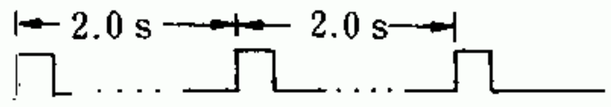
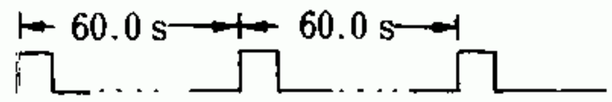
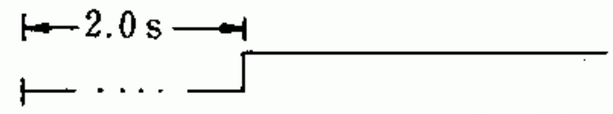
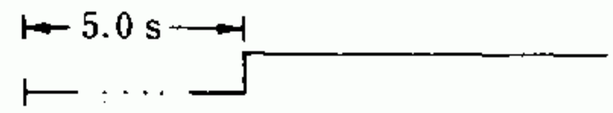
No.	Name	Data address	Symbol	Remarks
0	1 Scan Blinking relay	SB000030		
1	0.5s Blinking relay	SB000031		
2	1.0s Blinking relay	SB000032		
3	2.0s Blinking relay	SB000033		
4	0.5s Sampling relay	SB000034		
5	1.0s Sampling relay	SB000035		
6	2.0s Sampling relay	SB000036		
7	60.0s Sampling relay	SB000037		
8	Power on 1.0s-delayed relay	SB000038		
9	Power on 2.0s-delayed relay	SB000039		
10	Power on 5.0s-delayed relay	SB00003A		
11				

Table 1.5 General word variables

No.	Name	Data address	Symbol	Remarks
0	High-speed scan time preset value	SW000004		In the same unit as PP scan time preset value : 【ms】
1	(System reserved)	SW000005		
2	Low-speed scan time preset value	SW000006		In the same unit as PP scan time preset value : 【ms】
3	(System reserved)	SW000007		
4	High-speed scan time current value	SW000008		Unlike PP scan time current value, this is in units of time from the previous scan to the current scan : 【ms】
5	(System reserved)	SW000009		
6	Low-speed scan time current value	SW000010		Unlike PP scan time current value, this is in units of time from the previous scan to the current scan : 【ms】
7	Execution scan time current value	SW000011		Used in a function sharing the scan time current value (high-/low-speed) of the current scan program in execution.
8	Calendar : year	SW000012		Gregorian year 1999 : 0099 (BCD)
9	: Month and day	SW000013		December 31 : 1231 (BCD)
10	: Hour and minute	SW000014		23 hours and 59 minutes : 2359 (BCD)
11	: seconds	SW000015		59 seconds : 59 (BCD)
12	: week	SW000016		0 to 6 : Sunday, Monday through Saturday

Table 1.6 System execution status

No.	Name	Data address	Remarks
0	RTC count	SW00040	Incremented by one for each RTC interrupt
1	GND count	SW00041	Number of times ground task online self diagnosis is executed.
2	GND execution time	SW00042	Low-speed scan idle time current value Unit : 【100 μs/ 1scan】
3	GND minimum time	SW00043	Low-speed scan idle time minimum value Unit : 【100 μs/ 1scan】
4	GND maximum time	SW00044	Low-speed scan idle time maximum value Unit : 【100 μs/ 1scan】
5	Data trace execution status	SW00045	Data trace execution end status Group-1, 2 ; Bit-0, 1
6	Number of residual bytes in object memory	SW00046 SW00047	Number of residual bytes in object memory (272k bytes), upper and lower
7	Number of used bytes in object memory	SW00048 SW00049	Number of bytes used in object memory (272k bytes), upper and lower

Table 1.7 System error status

No.	Name	Data address	Remarks
0	Error type	SW00050	Error type
1	Error code	SW00051	Used for system error analysis.
2	Error IP	SW00052	
3	Error CS	SW00053	
4	Error task	SW00054	0C10H : DWG.A 0C30H : DWG.I 0C90H : DWG.L 0C50H : DWG.H Other : System
5	Program level	SW00055	0000H : System program 0001H : DWG program 0002H ~: Function program
6	DWG #Step	SW00056	Program step number of the function reference DWG at the time of an error in the function. Always zero at the time of an error in the DWG.
7	DWG No.	SW00057	Basic drawing : C800H Detailed drawing : xx00H (xxH : detailed drawing number) Expanded drawing : xxyyH (yyH : expanded drawing number) * Add 64H for a batch detailed drawing.
8	Error data	SW00058 to SW00069	Used for system error analysis.

Table 1.8 Online self diagnosis command status

No.	Name	Data address	Remarks
1	Diagnostic command*	SW00070	Diagnosis executed with the bit ON. (Initial value = 003FH)
2	Diagnosis execution counter	SW00071	Counts the diagnosis.
3	Diagnostic result	SW00072	The bit corresponding to the diagnostic command turns ON at occurrence of an error.
4	(System reserved)	SW00073	
5	2131F module status	SW00074 SW00075 SW00076 SW00077	Module status line 0 Module status line 1 Module status line 2 Module status line 3

* Diagnostic command bit allocation

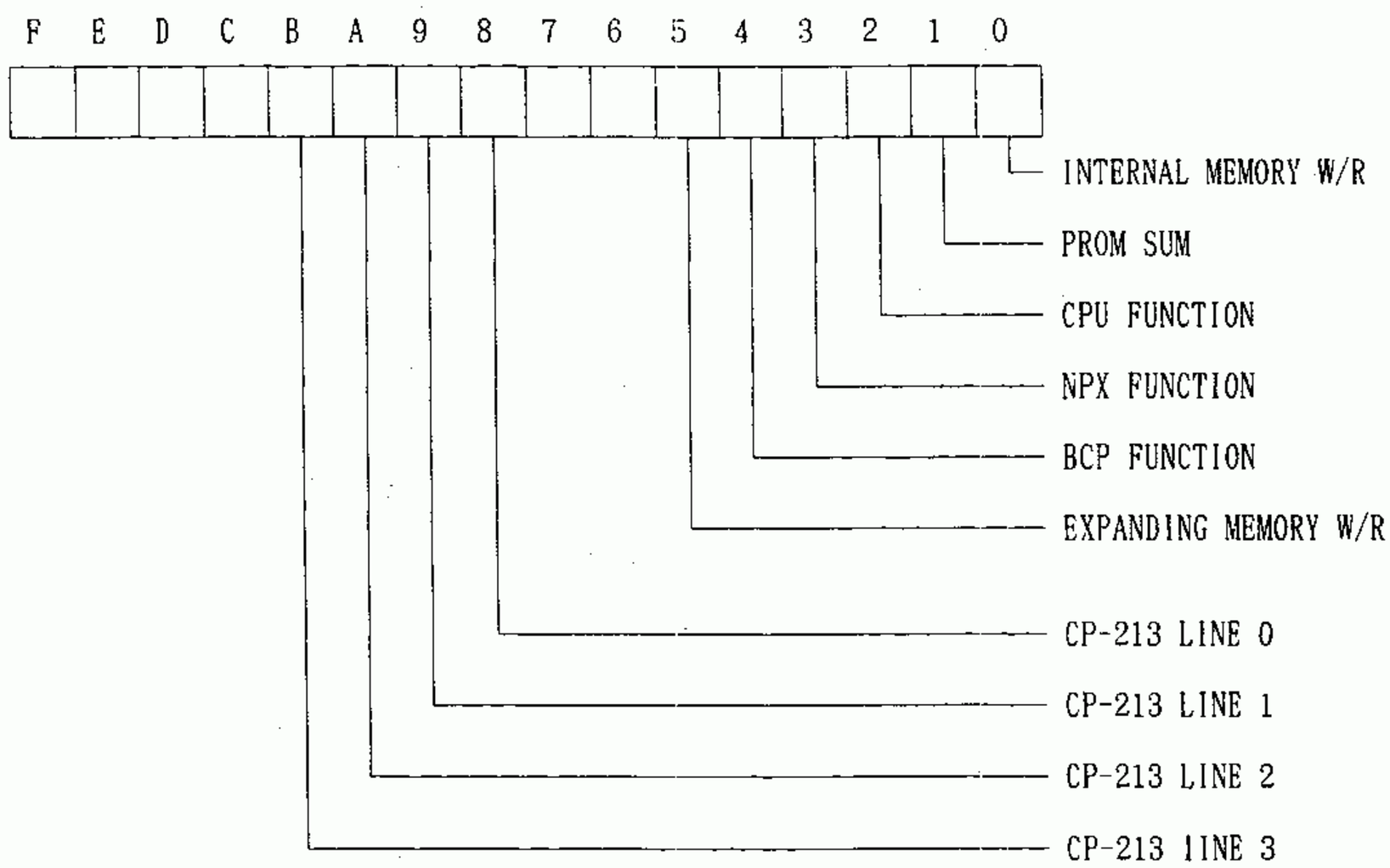


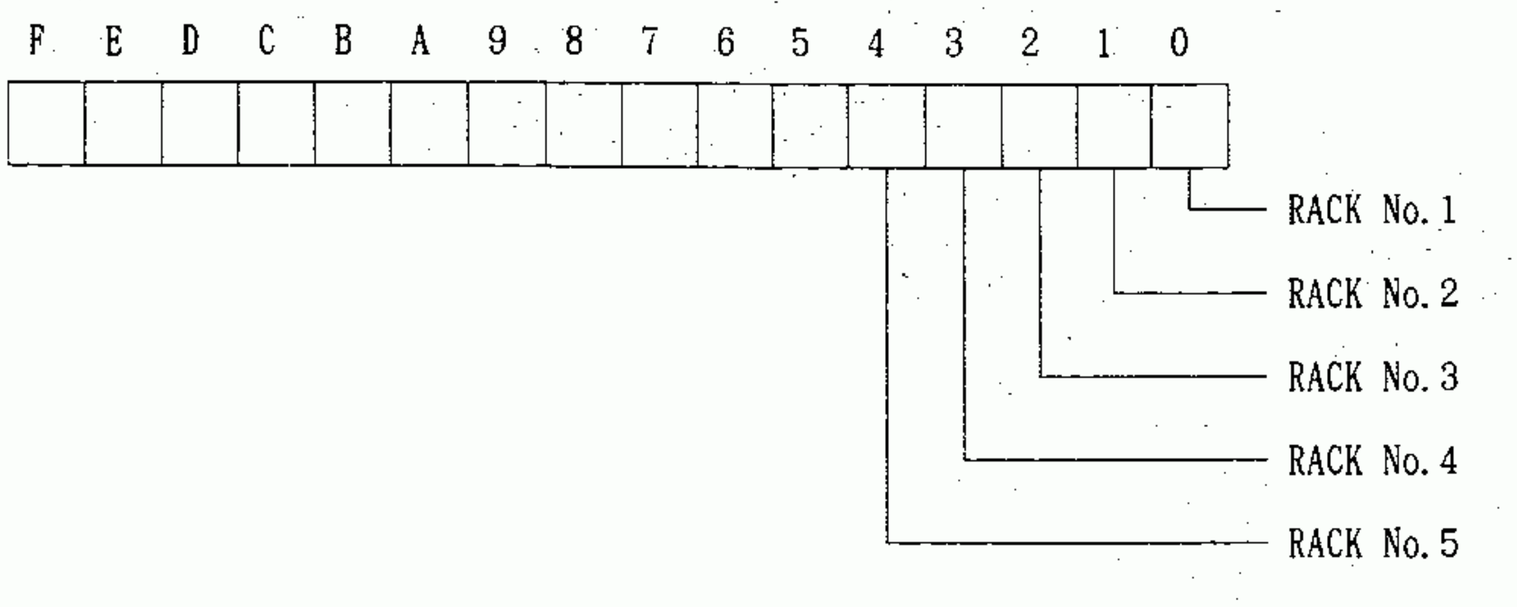
Table 1.9 User operation error status

No.	Name	Data address	Remarks
0	DWG. A error count error code	SW00080 SW00081	Error code Integer operation error 0001H : Under Flow 0002H : Over Flow 0003H : Divide Error 010XH : Error handling drawing error Real number operation error 001XH : Integer store 002XH : Real number store, division by zero 003XH : Real number operation 004XH : Internal system function 005XH : Internal system function
1	DWG. I error count error code	SW00082 SW00083	
2	DWG. H error count error code	SW00084 SW00085	
3	(Reserved)	SW00086 SW00087	
4	DWG. L error count error code	SW00088 SW00089	

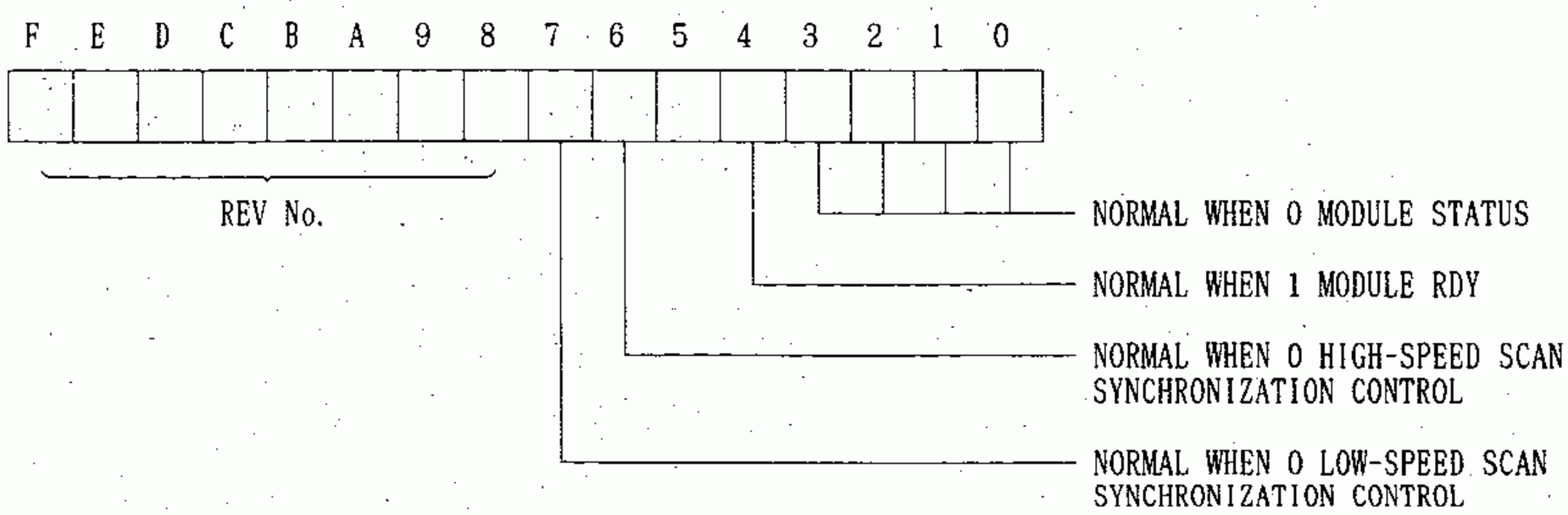
Table 1.10 System I/O status

No.	Name	Data address	Symbol	Remarks
0	Error count	SW00100		Number of occurrences of I/O errors
1	I/O error address	SW00101		Latest I/O error address
2	Local I/O rack status	SW00102		Local rack error status*1
3	Remote I/O module status	SW00103		Remote I/O driver (RIOD) module status*2
4	(Reserved)	SW00104		
5	COMM module status	SW00105		COMM module status*3
6	IOP module status	SW00106		IOP modele status*3
7	(Reserved)	SW00107		
8	213IF-0 module status	SW00108		Module status line 0
		SW00109		RESET/ICB execution status
		SW00110		Initial data/trace execution status
		SW00111		Slave initial data receiving
9	213IF-1 module status	SW00112		Module status line 1
		SW00113		RESET/ICB execution status
		SW00114		Initial data/trace execution status
		SW00115		Slave initial data receiving
10	213IF-2 module status	SW00116		Module status line 2
		SW00117		RESET/ICB execution status
		SW00118		Initial data/trace execution status
		SW00119		Slave initial data receiving
11	213IF-3 module status	SW00120		Module status line 3
		SW00121		RESET/ICB execution status
		SW00122		Initial data/trace execution status
		SW00123		Slave initial data receiving
12	(Reserved)	SW00124 to SW00129		
13	Input error status	SW00130 to SW00385		Local, remote, 213IF, link input word status (16 words/register)
14	(Reserved)	SW00386 to SW00399		
15	Output error status	SW00400 to SW00655		Local, remote, 213IF, link output word status (16 words/register)

* 1 : Local I/O rack error status bit allocation



* 2 : RIOD module status bit allocation



* 3 : COMM/IOP module status bit allocation

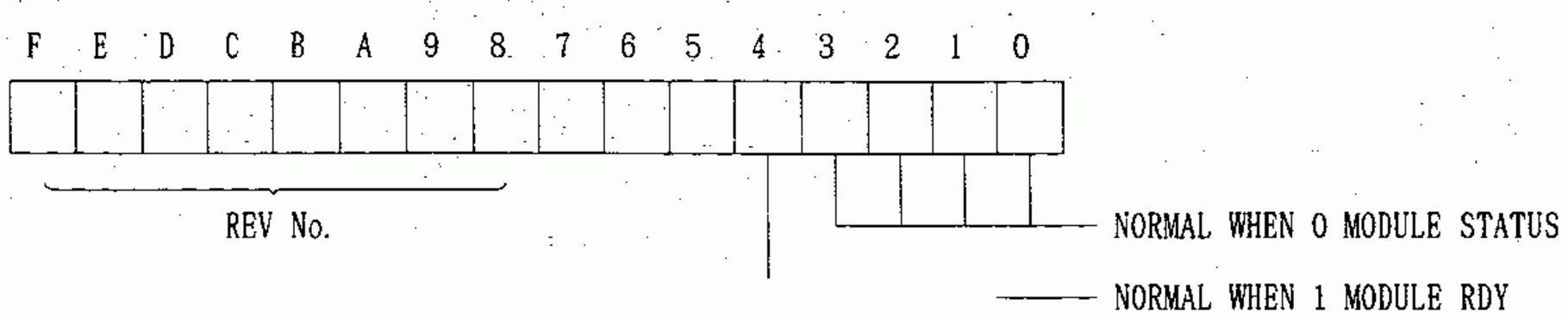


Table 1.11 Communication status

No.	Name	Data address	Remarks
0	MEMOBUS slave status PORT #1	SW00670	Internal command
		SW00671	MEMOBUS command/port number
		SW00672	Number of words
		SW00673	Register number
1	MEMOBUS slave status PORT #2	SW00674	Internal command
		SW00675	MEMOBUS command/port number
		SW00676	Number of words
		SW00677	Register number
2	MEMOBUS slave status PORT #3	SW00678	Internal command
		SW00679	MEMOBUS command/port number
		SW00680	Number of words
		SW00681	Register number
3	MEMOBUS slave status PORT #4	SW00682	Internal command
		SW00683	MEMOBUS command/port number
		SW00684	Number of words
		SW00685	Register number
4	(Reserved)	SW00686 to SW00689	
5	ASCII module interface (1)	SW00690	Send request (CPU → RIOD) RIOD #1
		SW00691	Send request (CPU → RIOD) RIOD #2
6	ASCII module interface (2)	SW00692	Receive acknowledge (CPU → RIOD) RIOD #1
		SW00693	Receive acknowledge (CPU → RIOD) RIOD #2
7	ASCII module interface (3)	SW00694	Receive request (RIOD → CPU) RIOD #1
		SW00695	Receive request (RIOD → CPU) RIOD #2
8	ASCII module interface (4)	SW00696	Send acknowledge (RIOD → CPU) RIOD #1
		SW00697	Send acknowledge (RIOD → CPU) RIOD #2
9	RIOD module interface (1)	SW00698	Request/acknowledge (CPU → RIOD)
10	RIOD module interface (2)	SW00699	Request/acknowledge (RIOD → CPU)

APPENDIX 2 ALLOCATING I/O VARIABLES

2.1 ALLOCATING PROCESS I/O

	I/O module	I/O MAP allocation
IW, 0W0000	Local I/O data, 512wds	I/O address (1) 0000 to 00FF (2) 0100 to 01FF
IW, 0W0200	Remote I/O data, 512wds	(1) 0200 to 02FF (2) 0300 to 03FF
IW, 0W0400	2131F line 0 I/O data, 496wds	(1) 0400 to 04FF (2) 0500 to 05FF
IW, 0W05F0	Control register 16wds	
IW, 0W0600	2131F line 1 I/O data, 496wds	(1) 0600 to 06FF (2) 0700 to 07FF
IW, 0W07F0	Control register 16wds	
IW, 0W0800	2131F line 2 I/O data, 496wds	(1) 0800 to 08FF (2) 0900 to 09FF
IW, 0W09F0	Control register 16wds	
IW, 0W0A00	2131F line 3 I/O data, 496wds	(1) 0A00 to 0AFF (2) 0B00 to 0BFF
IW, 0W0BF0	Control register 16wds	
IW, 0W0C00 IW, 0W0FF0	Link I/O data, 1008wds	(1) 0C00 to 0CFF (2) 0D00 to 0DFF (3) 0E00 to 0EFF (4) 0F00 to 0FFF
IW, 0W0FFF	Control register 16wds	

Fig. 2.1 Allocating I/O registers

2.2 213IF CONTROL REGISTERS

Table 2.1 Allocating 213IF control registers

No.	Item	Address	Remarks
0		0XF0	
1		F1	
2		F2	
3		F3	
4		F4	
5		F5	
6		F6	
7		F7	
8		F8	
9		F9	
10		FA	
11		FB	
12		FC	
13		FD	
14		FE	
15	Transmission status register	FF	Scan synchronization command with IN-Map specification

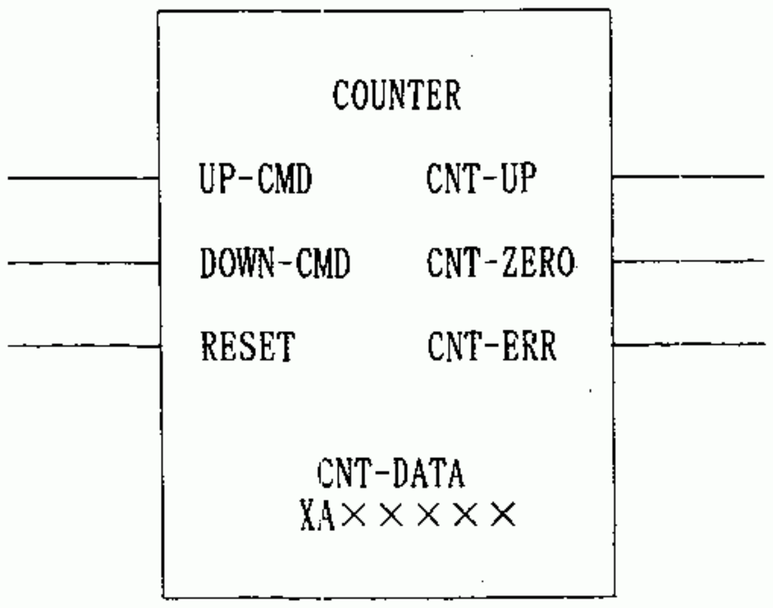
2.3 LINK CONTROL REGISTERS

Table 2.2 Allocating LINK control registers

No.	Item	Address	Remarks
0		0XF0	
1		F1	
2		F2	
3		F3	
4		F4	
5		F5	
6		F6	
7		F7	
8		F8	
9		F9	
10		FA	
11		FB	
12		FC	
13		FD	
14		FE	
15	Transmission status register	FF	Scan synchronization command with IN-Map specification

APPENDIX 3 SYSTEM STANDARD FUNCTIONS

3.1 COUNTER FUNCTIONS

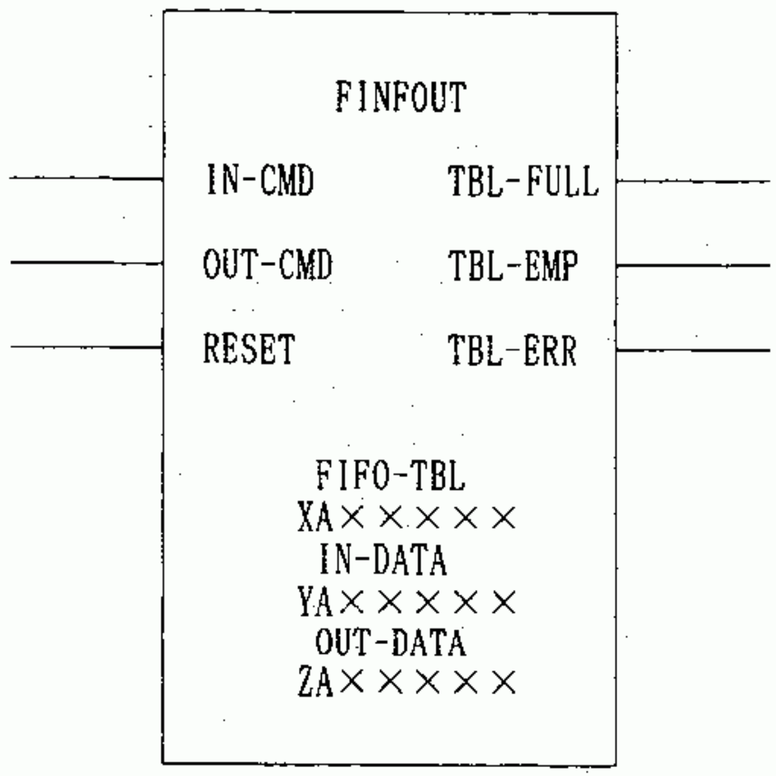
Function name	COUNTER
Function	Increases or decreases the current value when a counter up/down command (UP-CMD, DOWN-CMD) changes from OFF to ON. When a counter reset command (RESET) goes "ON", the counter current value is set to zero. Moreover, the counter current value is compared with a preset value and the result is output. Note : When a counter error occurs (current value > preset value), the current value will not be increased or decreased.
Function I/F	

	No.	Name	Type	Content
Input	1	UP-CMD	B	Up-count command (OFF → ON)
	2	DOWN-CMD	B	Down-count command (OFF → ON)
	3	RESET	B	Counter reset command
	4	CNT-DATA	A	Counter processing data area top address (3 wds : preset value, current value, work flag)
	5			
Output	1	CNT-UP	B	ON with counter current value = preset value
	2	CNT-ZERO	B	ON with counter current value = 0
	3	CNT-ERR	B	ON with counter current value > preset value
	4			
	5			

3.2 FIRST IN, FIRST OUT FUNCTION

Function name	FINFOUT
Function	A first-in, first-out type block data transfer function. The FIFO data table consists of a 4-wd header and a data buffer. The 3 wds (data size, input size, output size) in the header must be set before this function is referenced. When a data input command (IN-CMD) is "ON", a specified number of data items from a specified input data area are stored in sequence to the FIFO table. When a data output command (OUT-CMD) is "ON", a specified number of data items from the top of the FIFO table are transferred to a specified output data area. When a reset command (RESET) goes "ON", the number of data items stored is set to zero.

Function I/F

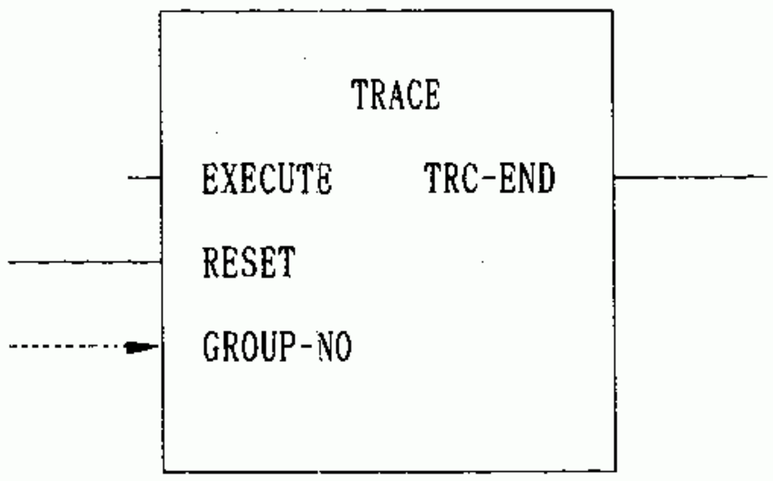


	No.	Name	Type	Content	Remarks
Input	1	IN-CMD	B	Data input command	FIFO table structure 0 : Data size 1 : Input size 2 : Output size 3 : Number of data items stored 4 : Data ⋮
	2	OUT-CMD	B	Data output command	
	3	RESET	B	Reset command	
	4	FIFO-TBL	A	Table top address	
	5	IN-DATA	A	Input data top address	
	6	OUT-DATA	A	Output data top address	
	7				
Output	1	TBL-FULL	B	FIFO table full	
	2	TBL-EMP	B	FIFO table empty	
	3	TBL-ERR	B	FIFO table error	
	4				

3.3 TRACE FUNCTION

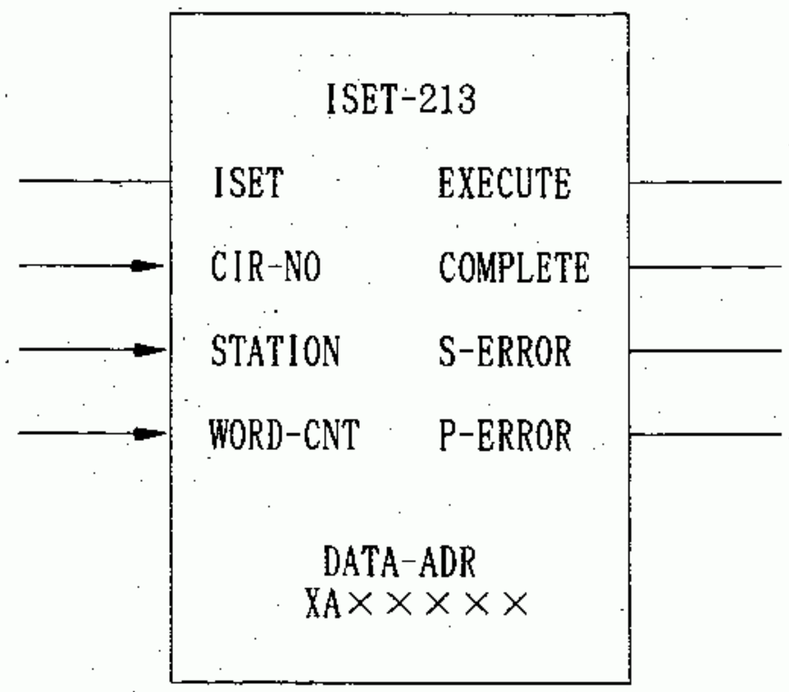
Function name	TRACE
Function	Controls trace execution on trace data specified by a trace group No. (The trace content is set by trace definition.) Tracing is executed when a trace execute command (EXECUTE) is "ON". When a trace reset command (RESET) goes "ON", the data in the trace buffer is erased and the trace counter reset. When the trace execution count reaches a preset value (set in the trace definition), trace end (TRC-END) goes "ON".

Function I/F



	No.	Name	Type	Content
Input	1	EXECUTE	B	Trace execution command
	2	RESET	B	Trace reset command
	3	GROUP-NO	W	Trace group No. specification
	4			
	5			
	6			
Output	1	TRC-END	B	Trace end
	2			
	3			
	4			
	5			
	6			

3.4 CP-213 INITIAL DATA SETUP FUNCTION

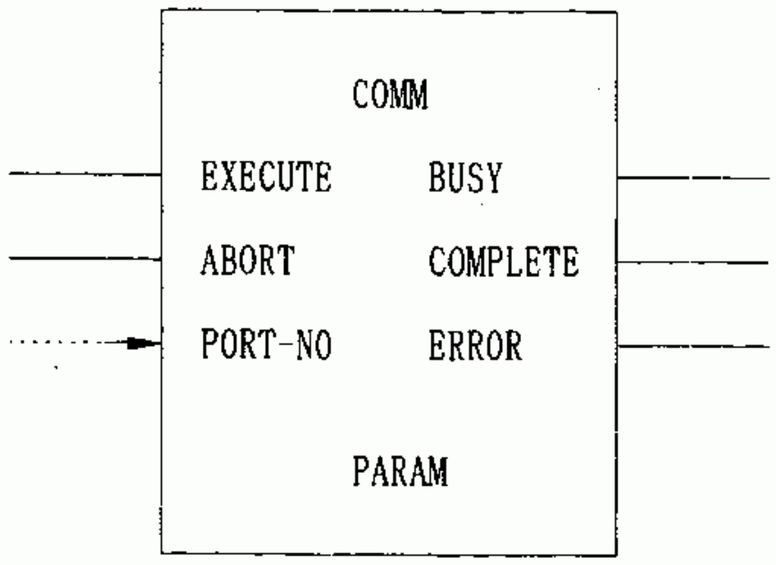
Function name	ISET-213
Function	Initial data in a specified variable area is set up in a specified remote station on a specified CP-213 line. This function can be referenced from DWG.A and DWG.L. When initialization is executed in DWG.A, setup completes immediately (about 1 ms). When initialization is executed in DWG.L, several scans are required for completing the setup. For reference in DWG.L, a setup command (ISET) must be held until a setup completion/error (COMPLETE/ERROR) goes "ON". For reference at more than one location in DWG.L, make sure the CP-213 line numbers and stations do not overlap.
Function I/F	

	No.	Name	Type	Content	Remarks
Input	1	ISET	B	Initial data setup command	
	2	CIR-NO	W	CP-213 line number	0 to 7
	3	STATION	W	CP-213 station	1 to 63
	4	WORD-CNT	W	Number of preset words	1 to 127
	5	DATA-ADR	A	Preset data top address	All variables can be specified
	6				
Output	1	EXECUTE	B	Initial data being set up	Valid when referenced from DWG.L
	2	COMPLETE	B	Initial data setup completed	
	3	S-ERROR	B	Initial data setting error	
	4	P-ERROR	B	Function parameter error	Function input
	5				

3.5 MEMOBUS MESSAGE TRANSMISSION MASTER FUNCTION

Function name	COMM	Use restriction	
Function	When an execution command (EXECUTE) goes "ON", the system sends a MEMOBUS command from a port specified by the port number (PORT-NO) to a slave and receives a MEMOBUS response from the destination. The execution command (EXECUTE) must be held until COMPLETE or ERROR goes "ON". The execution can be done in low-speed or high-speed scan.		

Function I/F



I/O

PARAM

No.	I/O	Content
00	OUT	Processing result
01	OUT	COMM STATUS
02	IN	Destination DEV #
03	IN	FUNCTION CODE
04	IN	Data address
05	IN	Data size
06	IN	(System reserved)
07	IN	Coil OFF
08	IN	Input relay OFF
09	IN	Input REG. OFF
10	IN	Hold REG. OFF
11	SYS	For system

I/O

I/O	TYPE	Name	Content
Input	BIT	EXECUTE	Send execution command
	BIT	ABORT	Send forced stop command Force send stop by a user program
	INT	PORT-NO	Port number (1 to 4)
	.....		
	.....		
Output	BIT	BUSY	Processing in progress
	BIT	COMPLETE	Processing completed
	BIT	ERROR	Error
	.....		
	.....		

3.5.1 Parameters

The parameters are explained below.

(1) Processing result (PARAM00)
The Processing result is output to the upper byte. The lower byte is reserved for system analysis.

00XX ····· Processing is progress (BUSY)
10XX ····· Processing completed (COMPLETE)
8XXX ····· Error occurred (ERROR)

(Error classification)

81XX ····· Function code error
An attempt was made to send an undefined function code (00H, 07H, 0CH, 0DH, 0EH, 11H, ···) or such a code was received.

82XX ····· Address setup error
The data address, coil offset, input relay offset, input register offset, or holding register offset set up is out of range.

83XX ····· Data size error
The send or receive data size is out of range.

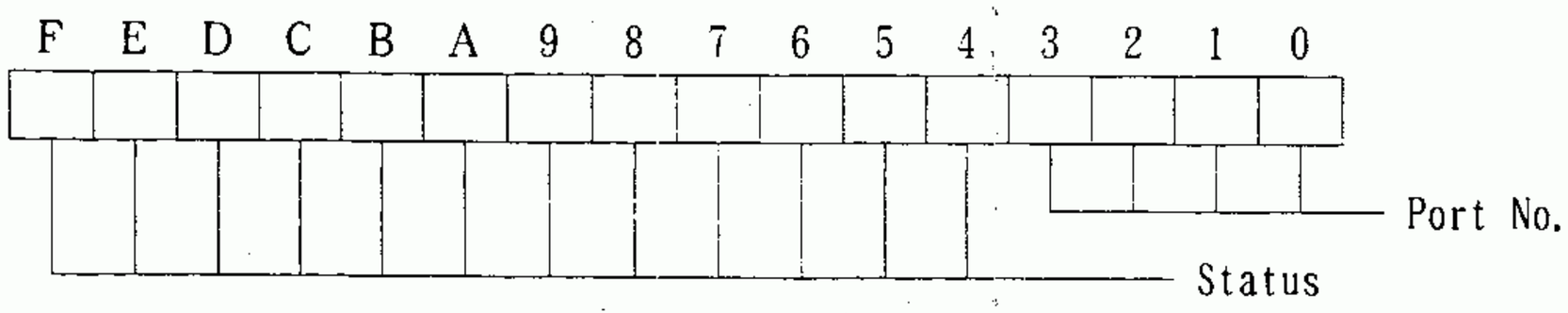
84XX ····· Port number setup error
The port number set up is 5 and over.

88XX ····· COMM error
An error response is returned from COMM. See COMM STATUS.

(2) COMM STATUS (PARAM01)

The COMM module status is output.

(a) Bit allocation



(b) Meaning

BIT#	MEANING
F	BCC error
E	Send/receive time-out
D	USART error (overrun, parity, framing error ; only at receive time)
C	Receive buffer overflow
B	erroneous number of send data
A	NO CONNECT ERROR
9	Port number error
8	COMM MODULE error
7	Not used
6	Not used
5	Contention (interference by connection with PP)
4	Token missing (for future use)
3 to 0	Port number

(3) Destination DEV # (PARAM02)

The device number of the destination as a slave is set up.

1 to 127 : A message is sent to a specified slave.
(Valid for all function codes)
0 : A message is sent to all slaves.
(Valid only for function codes 05, 06, 0F, 10H)

(4) FUNCTION CODE (PARAM03)

A MEMOBUS function code to be sent is set up.

00H	: Not used
01H	: Read coil status
02H	: Read input relay status
03H	: Read holding register content
04H	: Read input register content
05H	: Change single-coil status
06H	: Write single-holding register
07H	: Not used
08H	: Loopback test
09H	: Read holding register content (extended)
0AH	: Read input register content (extended)
0BH	: Write holding register (extended)
0CH to 0EH	: Not used
0FH	: Change multiple coil status
10H	: Write multiple holding registers
11H -	: Not used

Notes:

1. The function codes 09H, 0AH, and 0BH will be utilized in the future. An error response will be returned at present.
2. Only MW (MB) is available as a send/receive register in the master operation mode. In the slave operation mode, the coil, holding register, input relay and input register are assigned respectively to MB, MW, IB, and IW.

(5) Data address (PARAM04)

The top address of requested data to be read or written is set up.

A bit address is set up for a coil and relay and a word address for a register.

The setup range differs depending on the function code.

<Function code>	<Data address setup range>
00H : Not used	
01H : Read coil status	····· 0 to 65535 (0 to FFFFH) : Bit address
02H : Read input relay status	····· 0 to 65535 (0 to FFFFH) : Bit address
03H : Read holding register content	····· 0 to 16383 (0 to 3FFFH) : Word address
04H : Read input register content	····· 0 to 16383 (0 to 3FFFH) : Word address
05H : Change single-coil status	····· 0 to 65535 (0 to FFFFH) : Bit address
06H : Write single-holding register	····· 0 to 16383 (0 to 3FFFH) : Word address
07H : Not used	····· Invalid
08H : Loopback test	····· Invalid
09H : Read holding register content (extended)	····· 0 to 16383 (0 to 3FFFH) : Word address
0AH : Read input register content (extended)	····· 0 to 16383 (0 to 3FFFH) : Word address
0BH : Write holding register (extended)	····· 0 to 16383 (0 to 3FFFH) : Word address
0CH to 0EH : Not used	····· Invalid
0FH : Change multiple ciol status	····· 0 to 65535 (0 to FFFFH) : Bit address
10H : Write multiple holding registers	····· 0 to 16383 (0 to 3FFFH) : Word address
11H - : Not used	

(6) Data size (PARAM05)

The read or write requested data size (number of bits or words) is set up.

The setup range differs depending on the function code.

<Function code>	<Data size setup range >
00H : Not used	
01H : Read coil status	····· 1 to 2000(1 to 07D0H) : Number of bits
02H : Read input relay status	····· 1 to 2000(1 to 07D0H) : Number of bits
03H : Read holding register content	····· 1 to 125 (1 to 007DH) : Number of words
04H : Read input register content	····· 1 to 125 (1 to 007DH) : Number of words
05H : Change single-coil status	····· Invalid
06H : Write single-holding register	····· Invalid
07H : Not used	····· Invalid
08H : Loopback test	····· Invalid
09H : Read holding register content (extended)	····· 1 to 252 (1 to 00FCH) : Number of words
0AH : Read input register content (extended)	····· 1 to 252 (1 to 00FCH) : Number of words
0BH : Write holding register (extended)	····· 1 to 251 (1 to 00FBH) : Number of words
0CH to 0EH : Not used	····· Invalid
0FH : Change multiple coil status	····· 1 to 800 (1 to 0320H) : Number of bits
10H : Write multiple holding registers	····· 1 to 100 (1 to 0064H) : Number of words
11H - : Not used	

(7) Coil OFF (PARAM07)

The coil offset word address is set up.

This parameter is valid only for function codes 01H, 05H, 0FH.

(8) Input relay OFF (PARAM08)

The input relay offset word address is ste up.

This parameter is valid only for function code 02H.

(9) Input REG. OFF (PARAM09)

The input register offset word address is set up.

This parameter is valid only for function codes 04H, 0AH.

(10) Hold REG. OFF (PARAM10)

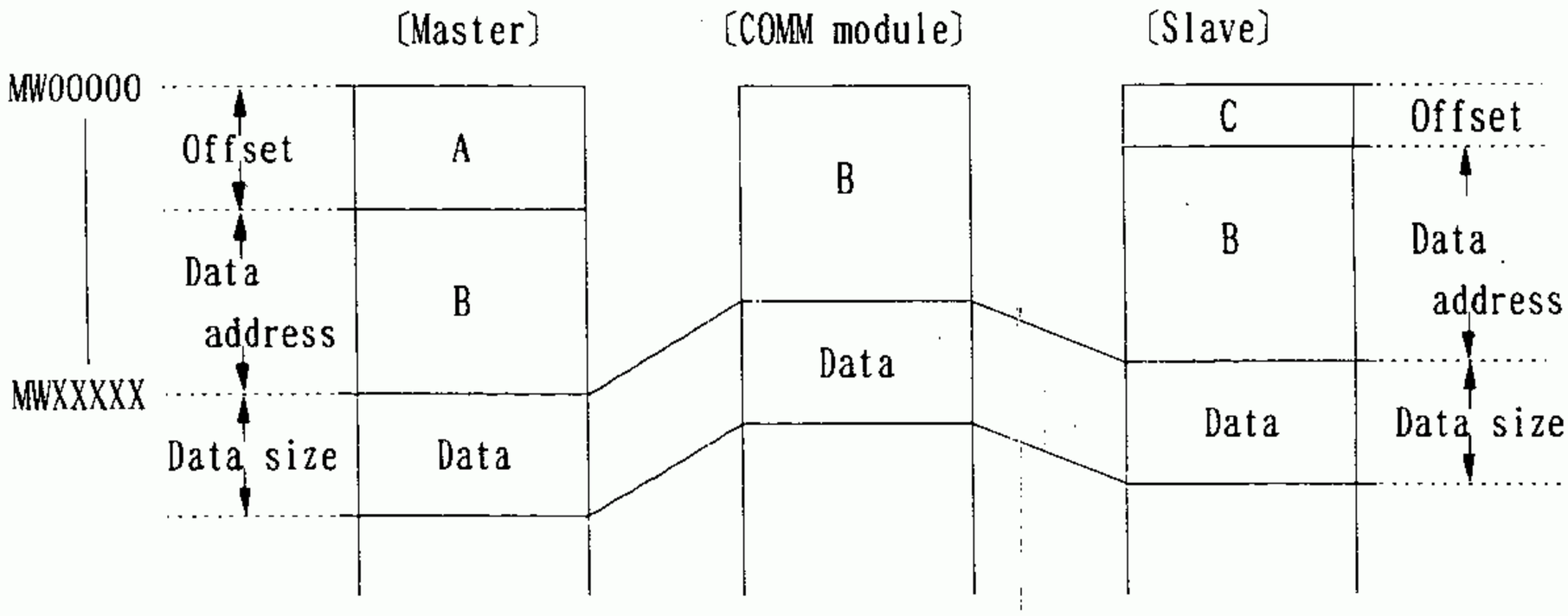
The holding register offset word address is set up.

This parameter is valid only for function code 03H, 06H, 09H, 0BH, 10H.

(11) For system (PARAM11)

The current port number in transmission is held. Be sure to set this to 0000H by a user program at the first scan at the time of power-on. After this, do not change the value by the user program since it is used by the system.

(12) Relationship between data address, size, and offset



A = sender offset address

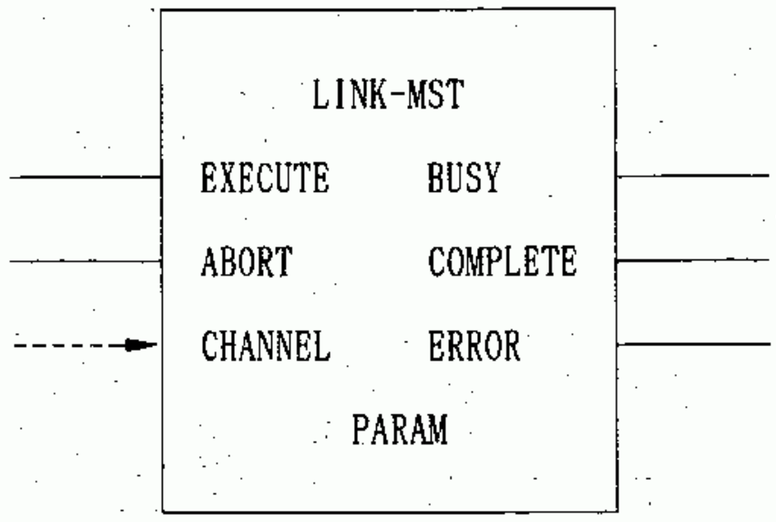
B = sender data address

C = receiver offset address

3.6 LINK MEMOBUS COMMUNICATION MASTER FUNCTION

Function name	LINK-MST	Use restriction	
Function	Sends a MEMOBUS command to a slave and receives a MEMOBUS response from the destination. Reads/writes a coil, input relay, holding register, input register, etc. This function can be executed at low-speed or high-speed scan.		

Function I/F



I/O

PARAM

No.	I/O	Content
00	OUT	Processing result
01	OUT	LINK STATUS
02	IN	Destination ST#
03	IN	FUNCTION CODE
04	IN	Data address
05	IN	Data size
06	IN	Destination CP#
07	IN	Coil OFF
08	IN	Input relay OFF
09	IN	Input REG. OFF
10	IN	Holding REG. OFF
11	SYS	For system

I/O

I/O	TYPE	Name	Content
Input	BIT	EXECUTE	Send execution command
	BIT	ABORT	Send forced stop command Stop sending by force by a user program
	INT	CHANNEL	Send channel number
	.....		
	.....		
Output	BIT	BUSY	Processing in progress
	BIT	COMPLETE	Processing completed
	BIT	ERROR	Error occurred
	.....		
	.....		

3.6.1 Parameters

(1) Processing result (PARAM00)

The processing result is output to the upper byte. The lower byte is reserved for system analysis.

00XX ····· Processing in progress (BUSY)

10XX ····· Processing completed (COMPLETE)

8XXX ····· Error occurred (ERROR)

〔Error classification〕

81XX ····· Function code error

An attempt was made to send an undefined function code (00H, 07H, 0CH, 0DH, 0EH, 11H, …) or such a code was received.

82XX ····· Address setup error

The data address, coil offset, input relay offset, input register offset, or holding register offset set up is out of range.

The data address set up is out of range.

83XX ····· Data size error

The send or receive data size is out of range.

84XX ····· Channel number setup error

The channel number set up is 0 or 17 or more.

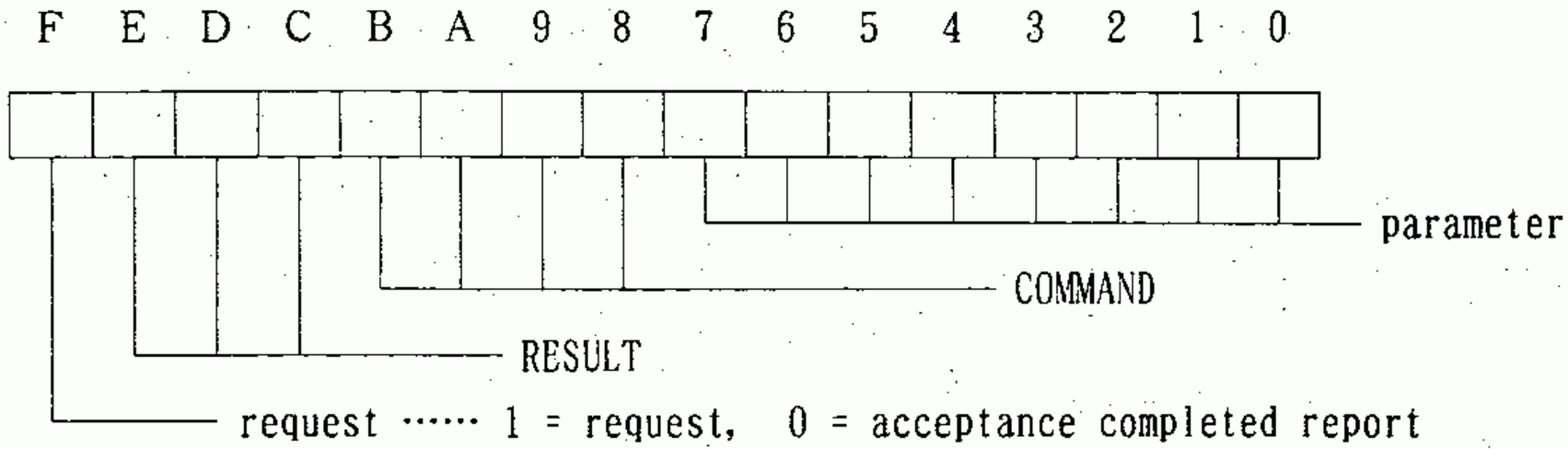
88XX ····· LINK error

An error response is returned from LINK. See LINK STATUS.

(2) LINK STATUS (PARAM01)

The LINK module status is output.

(a) Bit allocation



(b) COMMAND

Code	Abbreviation	Meaning
1	U_SEND	General message sending
2	U_REC	General message receiving
3	ABORT	Forced stop
8	M_SEND	MEMOBUS command sending··· Completed by response receiving
9	M_REC	MEMOBUS command receiving··· Followed by response sending
C	MR_SEND	MEMOBUS response sending

(c) RESULT

Code	Abbreviation	Meaning
1	SEND_OK	: Normal send completed
2	REC_OK	: Normal receive completed
3	ABORT_NG	: Forced stop completed
4	FMT_ND	: Parameter format error
5	SEQ_NG	: Command sequence error. M REC, MR SEND sequence error
	or INTI_NG	: A token not yet received. Not connected to a FABUS transmission system.
6	RESET_NG	: A send/receive command issued in the reset status (before issuing of a RUN command)
	or O_RING_NG	: Out of ring. A token could not be received within the token monitoring time.
7	REC_NG	: Data receive error (the error detected in the lower-level program)

(d) Parameter

COM	Meaning
1.8	Station address of the destination to which to send 1 to 32 = Individual address 129 to 160 = Group address 255 = Global address
2.9	Station address of the destination from which to receive, or station address of the other party from which data were received. 1 to 32 = Individual address 129 to 160 = Group address 255 = Global address
Other	Invalid

(3) Destination ST # (PARAM02)

1 to 32 : Sends to a specified station.

129 to 160 : Sends to stations of a specified group address. (Group transmission)

00FFH : Sends to all stations. (Broadcasting)

(4) FUNCTION CODE (PARAM03)

The MEMOBUS function codes to be sent are set up.

00H : Not used
01H : Read coil status
02H : Read input relay status
03H : Read holding register content
04H : Read input register content
05H : Change single-coil status
06H : Write single-holding register
07H : Not used
08H : Loopback test
09H : Read holding register content (extended)
0AH : Read input register content (extended)
0BH : Write holding register (extended)
0CH to 0EH : Not used
0FH : Change multiple coil status
10H : Write multiple holding registers
11H - : Not used

Note: MW (MB) is the only send/receive register during master operation.

In the slave operation, MB, MW, IB, and IW are used respectively as a coil, holding register, input relay, and input register.

(5) Data address (PARAM04)

The top address of requested data to be read or written is set up.

A bit address is set up for a coil and relay and a word address for a register.

The setup range differs depending on the function code.

<Function code>	<Data address setup range>
00H : Not used	
01H : Read coil status	····· 0 to 65535 (0 to FFFFH) : Bit address
02H : Read input relay status	····· 0 to 65535 (0 to FFFFH) : Bit address
03H : Read holding register content	····· 0 to 16383 (0 to 3FFFH) : Word address
04H : Read input register content	····· 0 to 16383 (0 to 3FFFH) : Word address
05H : Change single-coil status	····· 0 to 65535 (0 to FFFFH) : Bit address
06H : Write single-holding register	····· 0 to 16383 (0 to 3FFFH) : Word address
07H : Not used	····· Invalid
08H : Loopback test	····· Invalid
09H : Read holding register content (extended)	····· 0 to 16383 (0 to 3FFFH) : Word address
0AH : Read input register content (extended)	····· 0 to 16383 (0 to 3FFFH) : Word address
0BH : Write holding register (extended)	····· 0 to 16383 (0 to 3FFFH) : Word address
0CH to 0EH : Not used	····· Invalid
0FH : Change multiple coil status	····· 0 to 65535 (0 to FFFFH) : Bit address
10H : Write multiple holding registers	····· 0 to 16383 (0 to 3FFFH) : Word address
11H - : Not used	

(6) Data size (PARAM05)

The read or write requested data size (number of bits or words) is set up.

The setup range differs depending on the function code.

〈Function code〉	〈Data size setup range 〉
00H : Not used	
01H : Read coil status	····· 1 to 2000(1 to 07D0H) : Number of bits
02H : Read input relay status	····· 1 to 2000(1 to 07D0H) : Number of bits
03H : Read holding register content	····· 1 to 125 (1 to 007DH) : Number of words
04H : Read input register content	····· 1 to 125 (1 to 007DH) : Number of words
05H : Change single-coil status	····· Invalid
06H : Write single-holding register	····· Invalid
07H : Not used	····· Invalid
08H : Loopback test	····· Invalid
09H : Read hold register content (extended)	····· 1 to 252 (1 to 00FCH) : Number of words
0AH : Read input register content (extended)	····· 1 to 252 (1 to 00FCH) : Number of words
0BH : Write holding register (extended)	····· 1 to 251 (1 to 00FBH) : Number of words
0CH to 0EH : Not used	····· Invalid
0FH : Change multiple coil status	····· 1 to 800 (1 to 0320H) : Number of bits
10H : Write multiple holding registers	····· 1 to 100 (1 to 0064H) : Number of words
11H - : Not used	

(7) Destination CP # (PARAM06)

The destination CP number is set up.

If the send destination party is CP-3500, set number 1 to 4. In other cases (CP-5500, CP-3300, etc.), set to 0.

(8) Coil OFF (PARAM07)

The coil offset word address is set up.

This parameter is valid only for function codes 01H, 05H, and 0FH.

(9) Input relay OFF (PARAM08)

The input relay offset word address is set up.

This parameter is valid only for the function code 02H.

(10) Input REG. OFF (PARAM09)

The input register offset word address is set up.

This parameter is valid only for function codes 04H and 0AH.

(11) Holding REG. OFF (PARAM10)

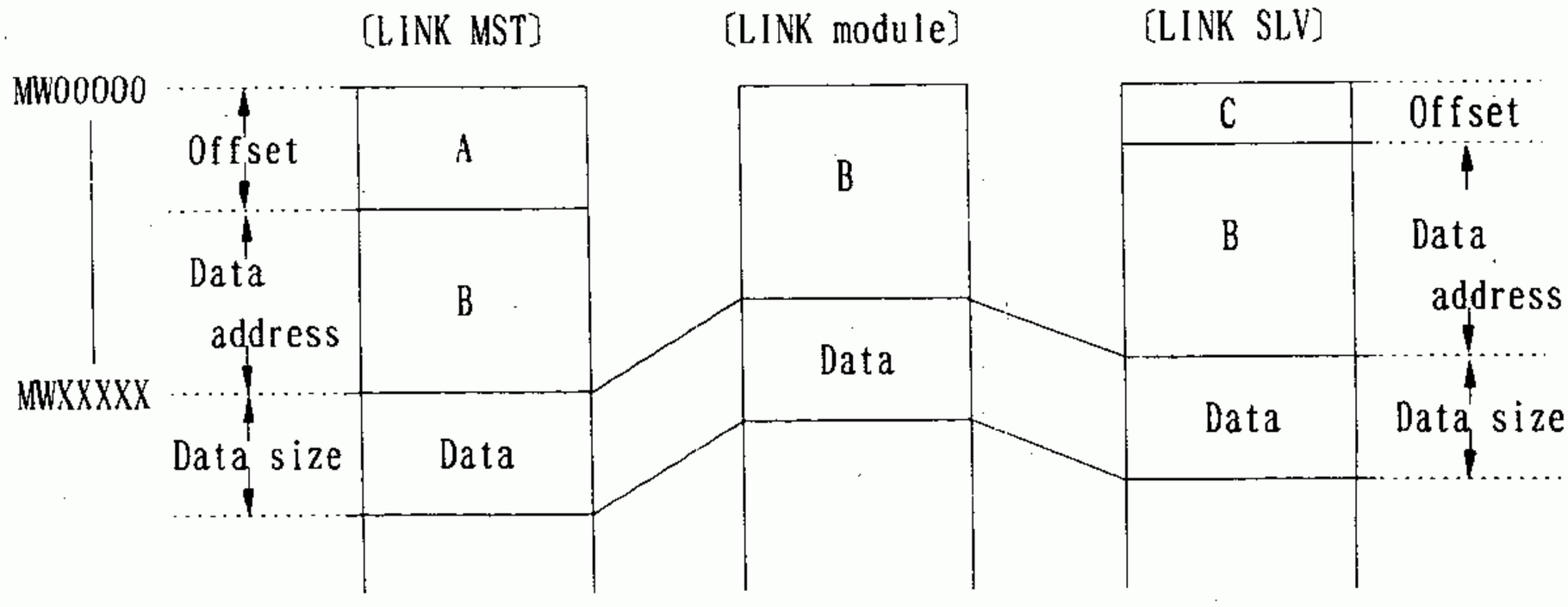
The holding register offset word address is set up.

This parameter is valid only for the function codes 03H, 06H, 09H, 0BH, and 10H.

(12) For system (PARAM11)

The current channel number used is held. Be sure to set this to 0000H by a user program at the first scan at the time of power-on. After this, do not change the value by the user program since the system uses it.

(13) Relationship between data address, size, and offset



A = sender offset address

B = sender data address

C = receiver offset address

3.6.2 Input

(1) EXECUTE (send execution command)
When the command goes "ON", a message is sent.

This must be held until COMPLETE (processing completed) or ERROR (error occurred) goes "ON".

(2) ABORT (send forced stop command)
Stops sending by force. This overrides EXECUTE (send execution command).

(3) CHANNEL (send channel number)
Specifies a send channel number. 1 to 16 is set.

3.6.3 Output

(1) BUSY (processing in progress)
Processing being executed. Hold EXECUTE at "ON".

(2) COMPLETE (processing completed)
This goes "ON" only for one scan at normal termination.

(3) ERROR (error occurred)
This goes "ON" only for one scan at occurrence of an error.

See PARAM00 (processing result) and PARAM01 (LINK STATUS) for the error cause.

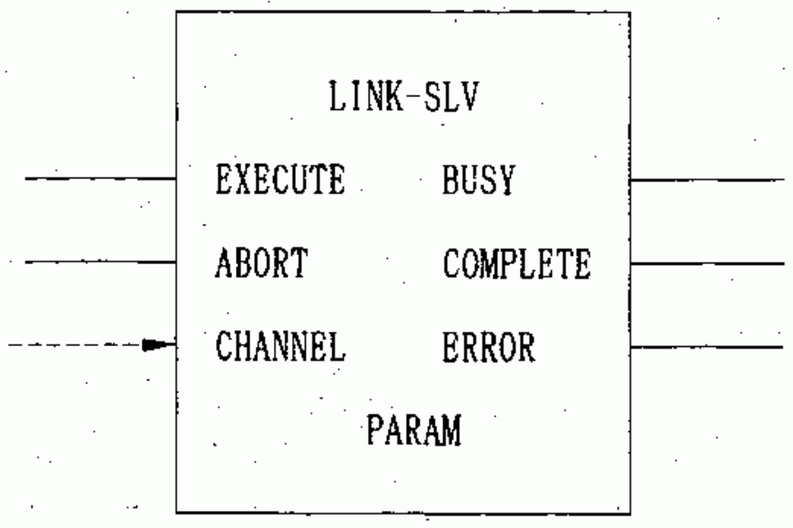
3.6.4 Sample programs

```
(Set the system register to 0 in the first scan)
   SB000003
|----| |----
[ |- 00000        ]                                        => DW00011 ]

(Start up
 every second)   (Completed)      (Error)     (One-second
                                              start up delay)   (Command)
   SB000035        DB000211       DB000212      SB000038        DB000201
|----| |-----+--------| |-----------| |-------------| |-------------( )----|
             |
(Hold command)|
   DB000201  |
|----| |-----+

                      (System function)
                   +--------------------+
 (Command)         |      LINK-MST      |                      (Running)
   DB000201        |                    |                       DB000210
|----| |-----------| EXECUTE      BUSY  |--------------------------( )----|
 (Abort)           |                    |                     (Completed)
   DB000208        |                    |                       DB000211
|----| |-----------| ABORT    COMPLETE  |--------------------------( )----|
                   |                    |                        (Error)
(Channel number)   |                    |                       DB000212
|- 00001 ......--> | CHANNEL      ERROR |--------------------------( )----|
                   | (Parameter address)|
                   |       PARAM        |
                   |      DA00000       |
                   +--------------------+

   DB000211
|----| |----
[  INC        DW00024 ]    (Pass counter)
   DB000212
|----| |----
IFON
INC      DW00025      (Error counter)
|- DW00000            (Save processing result)             =>  DW00026
|- DW00001            (Save LINK status)                   =>  DW00027
IEND
DEND
```

3.7 LINK MEMOBUS COMMUNICATION SLAVE FUNCTION

Function name	LINK-SLV	Use restriction	Allowed only at low-speed scan
Function	When an execute command (EXECUTE) goes "ON", a memobus command is received from a channel specified by a channel number (CHANNEL) and a memobus response sent to the other party. Hold the execute command (EXECUTE) until COMPLETE or ERROR "ON" is set.		

Function I/F



I/O

PARAM

No.	I/O	Content
00	OUT	Processing result
01	OUT	LINK STATUS
02	OUT	Destination ST#
03	OUT	FUNCTION CODE
04	OUT	Data address
05	OUT	Data size
06	OUT	Destination CP#
07	IN	Coil OFF
08	IN	Input relay OFF
09	IN	Input REG. OFF
10	IN	Hold REG. OFF
11	SYS	Write range LO
12	IN	Write range HI
13	SYS	Reserved for system

I/O

I/O	TYPE	Name	Content
Input	BIT	EXECUTE	Receive execute command
	BIT	ABORT	Receive abort command. Stop receiving by force by a user program.
	INT	CHANNEL	Receive channel number
	.....		
	.....		
Output	BIT	BUSY	Processing in progress
	BIT	COMPLETE	Processing completed
	BIT	ERROR	Error occurred
	.....		
	.....		

3.7.1 Parameters

(1) Processing result (PARAM00)

The processing result is output to the upper byte. The lower byte is reserved for system analysis.

00XX ····· Processing in progress (BUSY)

10XX ····· Processing completed (COMPLETE)

8XXX ····· Error occurred (ERROR)

〔Error classification〕

81XX ····· Function code error

An attempt was made to send an undefined function code (00H, 07H, 0CH, 0DH, 0EH, 11H, …) or such a code was received.

82XX ····· Address setup error

The data address, coil offset, input relay offset, input register offset, or holding register offset set up is out of range.

83XX ····· Data size error

The send or receive data size is out of range.

84XX ····· Channel number setup error

The channel number set up is 0 or 17 or more.

88XX ····· LINK error

A response is returned from LINK. See LINK STATUS.

(2) LINK STATUS (PARAM01)

The LINK module status is output. See Par. 3.6.1 (2).

(3) Destination ST# (PARAM02)

The send source station number is output.

(4) FUNCTION CODE (PARAM03)

The MEMOBUS function code received is output.

00H : Not used
01H : Read coil status
02H : Read input relay status
03H : Read holding register content
04H : Read input register content
05H : Change single-coil status
06H : Write single-holding register
07H : Not used
08H : Loopback test
09H : Read holding register content (extended)
0AH : Read input register content (extended)
0BH : Write holding register (extended)
0CH to 0EH : Not used
0FH : Change multiple coil status
10H : Write multiple holding registers
11H - : Not used

Notes: In the slave operation, MB, MW, IB, and IW are used respectively as a coil, holding register, input relay, and input register.

(5) Data address (PARAM04)

The data address requested by the sender is output.

(6) Data size (PARAM05)

The data size requested by the sender is output.

(7) Destination CP # (PARAM06)

The send source CP number is output.

(8) Coil OFF (PARAM07)

The coil offset word address is set up.

This parameter is valid only for the function codes 01H, 05H, and 0FH.

(9) Input relay OFF (PARAM08)

The input relay offset word address is set up.

This parameter is valid only for the function code 02H.

(10) Input REG. OFF (PARAM09)

The input register offset word address is set up.

This parameter is valid only for function codes 04H, and 0AH.

(11) Holding REG. OFF (PARAM10)

The holding register offset word address is set up.

This parameter is valid only for the function codes 03H, 06H, 09H, 0BH, and 10H.

(12) Write range LO (PARAM11), write range HI (PARAM12)

The write enabled range for a write request is set up. Any request out of this range leads to an error.

These parameters are valid only for the funciton codes 0BH, 0FH, and 10H.

$0 \leqq$ write range LO $\leqq$ write range HI $\leqq 16383$

(13) For system (PARAM13)

The current channel number used is held. Be sure to set this to 0000H by a user program at the first scan at the time of power-on. After this, do not change the value by the user program since the system uses it.

3.7.2 Input

(1) EXECUTE (receive execute command)
When the command goes "ON", message receiving is performed.

This must be held until COMPLETE (processing completed) or ERROR (error occurred) goes "ON".

(2) ABORT (receive forced stop command)
Stops receiving by force. This overrides EXECUTE (receive execute command).

(3) CHANNEL (receive channel number)
Specifies a receive channel number. Set 1 to 16.

3.7.3 Output

(1) BUSY (processing in progress)
Processing being executed. Hold EXECUTE at "ON".

(2) COMPLETE (processing completed)
This goes "ON" only for one scan at normal termination.

(3) ERROR (error occurred)
This goes "ON" only for one scan at occurrence of an error.

See PARAM00 (processing result) and PARAM01 (LINK STATUS) for the error cause.

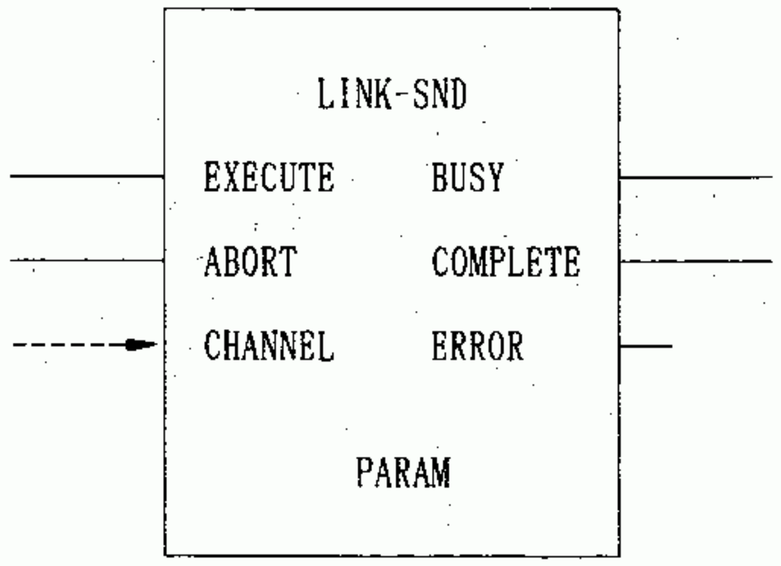
3.7.4 Sample programs

```
(Set the system register to 0 in the first scan)
    SB000003
 ├──┤ ├──
 [ ├─ 00000      ]                                          [     => DW00013 ]

                              (System function)

(Command : always ON)                LINK-SLV.                    (Being executed)
   SB000004                                                          DB000210
 ├──┤ ├──                    EXECUTE          BUSY                  ──( )──
   (Abort)
   DB000208                                                        (Completed)
 ├──┤ ├──                    ABORT            COMPLETE              DB000211
                                                                    ──( )──
                                                                    (Error)
(Channel number)                                                    DB000212
 ├─ 00002 ·····→            CHANNEL          ERROR                 ──( )──

                            (Parameter address)
                                 PARAM
                                 DA00000

   DB000211
 ├──┤ ├──
 [  INC       DW00024 ]    (Pass counter)
   DB000212
 ├──┤ ├──
IFON
INC     DW00025     (Error counter)
 ├─ DW00000         (Processing result saving)                  =>  DW00026
 ├─ DW00001         (LINK status saving)                        =>  DW00027
IEND
DEND
```

3.8 LINK DATA SEND FUNCTION

Function name	LINK-SND	Use restriction	
Function	Sends data to the other party's station. Does not receive a response from the other party. Can be executed at low-speed or high-speed scan. The M register is the only register that can send data.		

Function I/F



I/O

PARAM

No.	I/O	Content
00	OUT	Processing result
01	OUT	LINK STATUS
02	IN	Destination ST #
03	SYS	System reserved
04	IN	Data address
05	IN	Data size
06	IN	Destination CP #
07	SYS	Reserved for the system
08		
09		
10		

I/O

I/O	TYPE	Name	Content
Input	BIT	EXECUTE	Send execute command
	BIT	ABORT	Send forced stop command. Stop sending by force by a user program.
	INT	CHANNEL	Send channel number
	.....		
	.....		
Output	BIT	BUSY	Processing in progress
	BIT	COMPLETE	Processing completed
	BIT	ERROR	Error occurred
	.....		
	.....		

3.8.1 Parameters

(1) Processing result (PARAM00)

The processing result is output to the upper byte. The lower byte is reserved for system analysis.

00XX ····· Processing in progress (BUSY)
10XX ····· Processing completed (COMPLETE)
8XXX ····· Error occurred (ERROR)

〔Error classification〕

82XX ····· Address setting error
The data address set up is out of range.

83XX ····· Data size error
The send data size is out of range.

84XX ····· Channel number setup error
The channel number set up is 0 or 17 or more.

88XX ····· LINK error
A response is returned from LINK. See LINK STATUS.

(2) LINK STATUS (PARAM01)

The LINK module status is output. See Par. 3.6.1 (2).

(3) Destination ST# (PARAM02)

1 to 32 : Sends to a specified station.
129 to 160 : Sends to stations at a specified group address. (Group transmission)
00FFH : Sends to all stations. (Broadcasting)

(4) System reserved (PARAM03)

A system reserved register. Set 0.

(5) Data address (PARAM04)

The top address (word address) of data to be sent is set.

Setup range : 0 to 16383 (0 to 3FFFH)

(6) Data size (PARAM05)

The data size (number of words) of data to be sent is set.

Setup range : 1 to 254 (0 to 00FEH)

(7) Destination CP# (PARAM06)

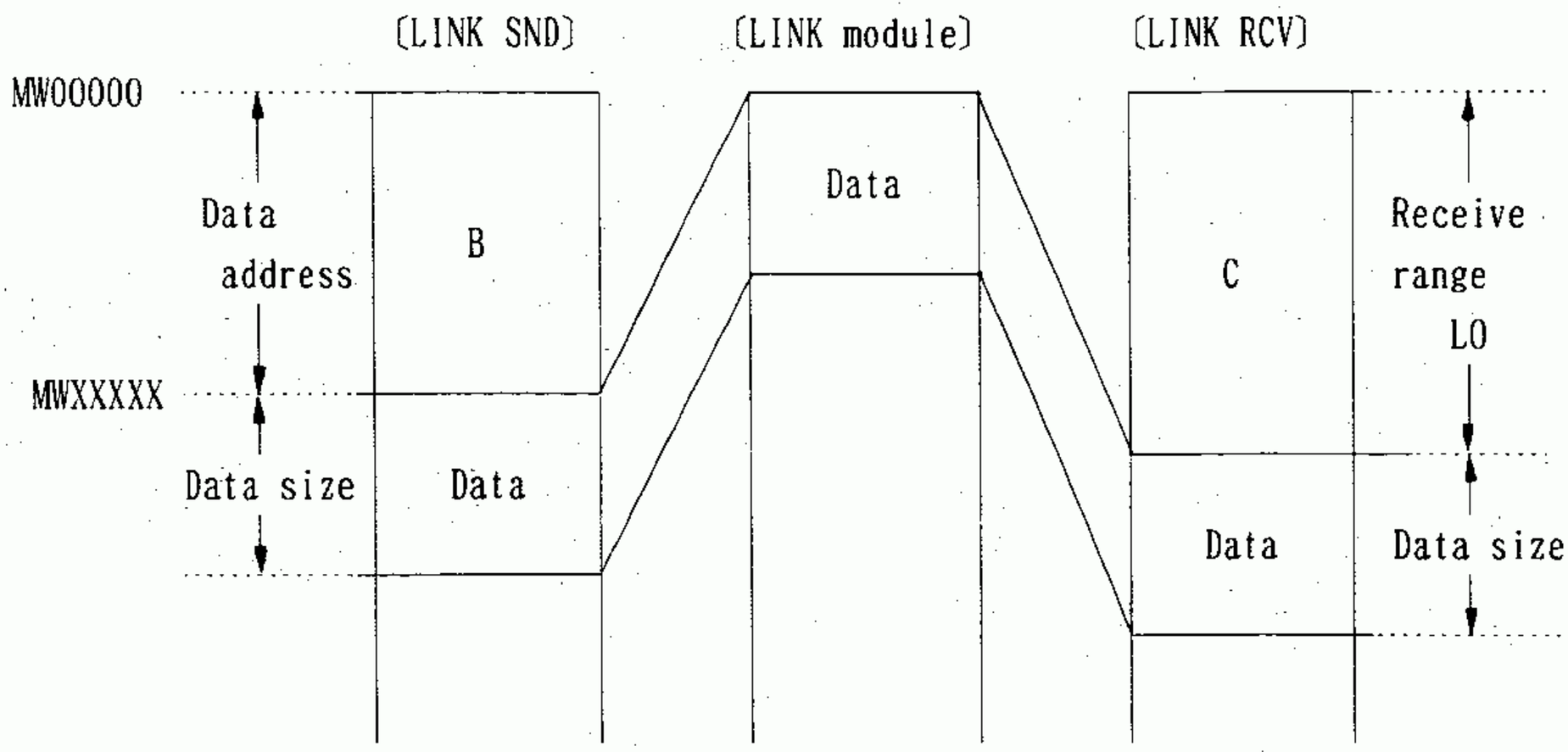
The destination number is set.

If the send destination is CP-3500, set a number 1 to 4. In other cases (for CP-5500, CP-3300, etc.), set to 0.

(8) Reserved for system (PARAM07)

The current channel number used is held. Be sure to set this to 0000H by a user program at the first scan after power-on. After this, do not change the value by the user program since the parameter is used by the system.

(9) Relationship between data address and data size.



B = Sender data address

C = Receiver receive range (LO)

3.8.2 Input

(1) EXECUTE (send execute command)
When the command goes "ON", message sending is performed.

This must be held until COMPLETE (processing completed) or ERROR (error occurred) goes "ON".

(2) ABORT (send forced stop command)
Stops sending by force. This overrides EXECUTE (send execute command).

(3) CHANNEL (send channel number)
Specifies a send channel number. Set a number from 1 to 16.

3.8.3 Output

(1) BUSY (processing in progress)
Processing being executed. Hold EXECUTE at "ON".

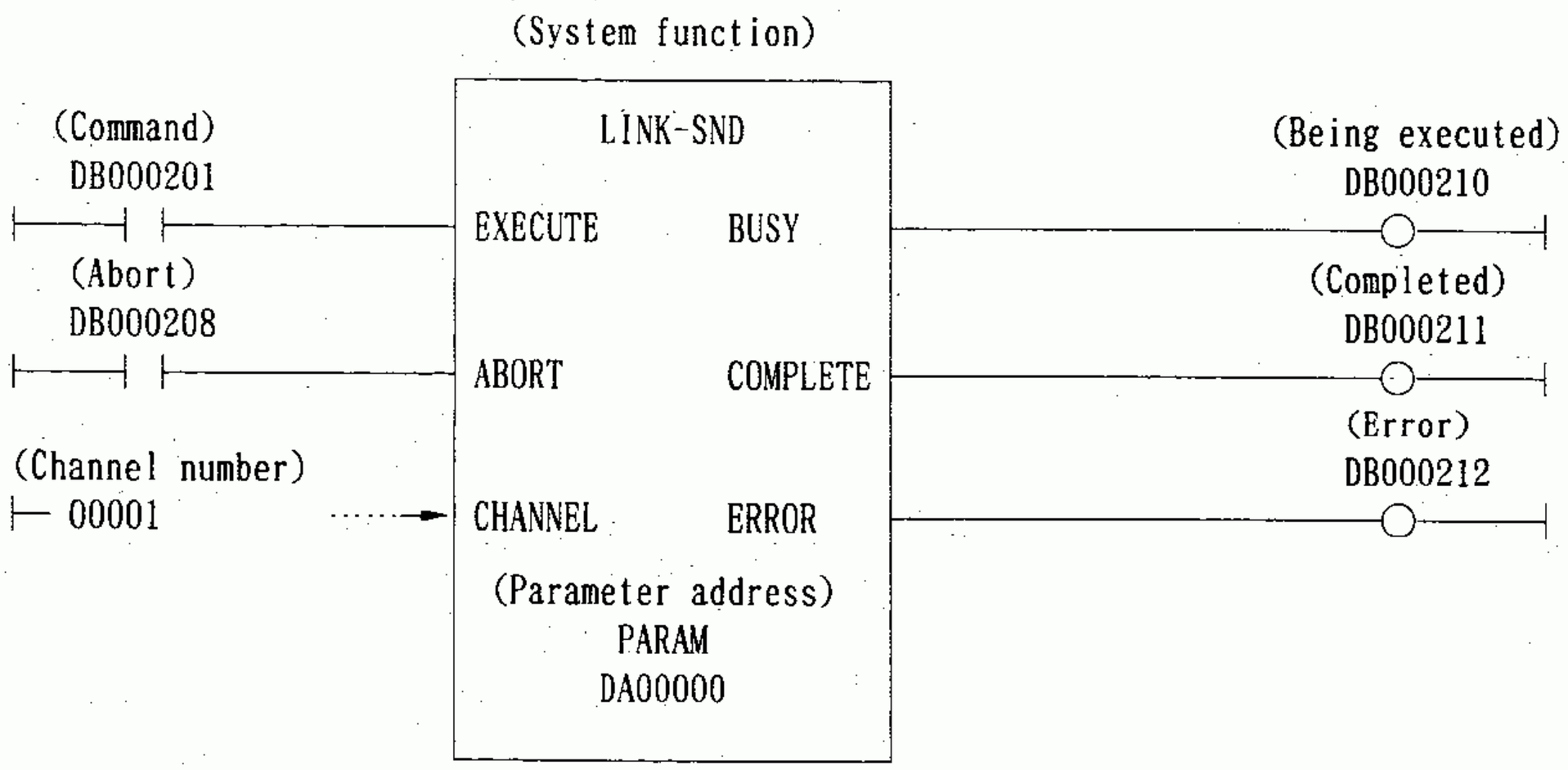
(2) COMPLETE (processing completed)
This goes "ON" only for one scan at normal termination.

(3) ERROR (error occurred)
This goes "ON" only for one scan at error occurrence.

See PARAM00 (processing result) and PARAM01 (LINK STATUS) for the error cause.

3.8.4 Sample programs

```
(Set the system register to 0 in the first scan)
    SB000003
 ---| |---
 [ |- 00000      ]                                                    => DW00013 ]

(Start up every second)  (Completed)      (Error)      (One-second delay at start up)  (Command)
    SB000035             DB000211         DB000212          SB000038                   DB000201
 ---| |-------------------| |--------------| |---------------| |-------------------------( )---
    |
    |  (Hold command)
    |     DB000201
    +---| |---
```

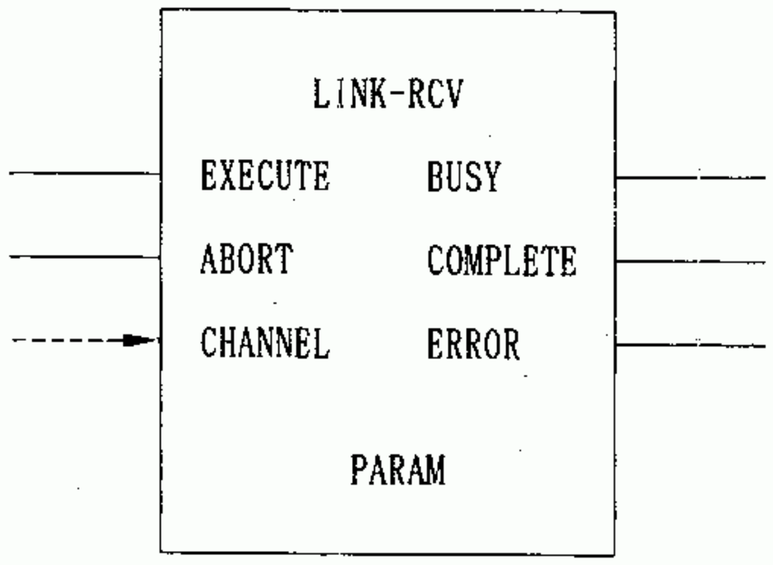



```
    DB000211
 ---| |---
 [  INC        DW00024 ]    (Pass counter)
    DB000212
 ---| |---
IFON
INC      DW00025    (Error counter)
 |- DW00000        (Processing result saving)                          => DW00026
 |- DW00001        (LINK status saving)                                => DW00027
IEND
DEND
```

3.9 LINK DATA RECEIVE FUNCTION

Function name	LINK-RCV	Use restriction	Allowed only in low-speed scan
Function	Receives data from the other party's station. Does not return a response. The M register is the only register that receive data.		

Function I/F



I/O

PARAM

No.	I/O	Content
00	OUT	Processing result
01	OUT	LINK STATUS
02	OUT	Destination ST #
03	SYS	System reserved
04	SYS	System reserved
05	OUT	Data size
06	OUT	Destination CP #
07	IN	Receive range LO
08	IN	Receive range HI
09	SYS	Reserved for the system
10		

I/O

I/O	TYPE	Name	Content
Input	BIT	EXECUTE	Receive execute command
	BIT	ABORT	Receive forced stop command. Stop receiving by force by a user program.
	INT	CHANNEL	Send channel number
	.....		
	.....		
Output	B T	BUSY	Processing in progress
	B T	COMPLETE	Processing completed
	B T	ERROR	Error occurred
	.....		

3.9.1 Parameters

(1) Processing result (PARAM00)

The processing result is output to the upper byte. The lower byte is reserved for system analysis.

00XX ..... Processing in progress (BUSY)
10XX ..... Processing completed (COMPLETE)
8XXX ..... Error occurred (ERROR)

〔Error classification〕

82XX ..... Address setting error
The data address set up is out of range.

83XX ..... Data size error
The send data size is out of range.

84XX ..... Channel number setup error
The channel number set up is 0 or 17 or more.

88XX ..... LINK error
A response is returned from LINK. See LINK STATUS.

(2) LINK STATUS (PARAM01)
The LINK module status is output. See Par. 3.6.1 (2).

(3) Destination ST# (PARAM02)
The send source station number is output.

(4) System reserved (PARAM03, PARAM04)
System reserved registers. Set 0.

(5) Data size (PARAM05)
The data size requested by the sender is output.

(6) Destination CP# (PARAM06)
The send source CP number is output.

(7) Receive range LO (PARAM07), receive range HI (PARAM08)
The write enables range for a receive request is set. A request outside this range is an error.

The receive range LO is also used as the top address at which to write receive data.

$0 \leqq$ receive range LO $\leqq$ receive range HI $\leqq 16383$

(8) Reserved for system (PARAM09)
The current channel number used is held. Be sure to set this to 0000H by a user program at the first scan after power-on. After this, do not change the value by the user program since the parameter is used by the system.

3.9.2 Input

(1) EXECUTE (receive execute command)
When the command goes "ON", message receiving is performed.

This must be held until COMPLETE (processing completed) or ERROR (error occurred) goes "ON".

(2) ABORT (receive forced stop command)
Stops receiving by force. This overrides EXECUTE (receive execute command).

(3) CHANNEL (receive channel number)
Specifies a receive channel number. Set a number from 1 to 16.

3.9.3 Output

(1) BUSY (processing in progress)
Processing being executed. Hold EXECUTE at "ON".

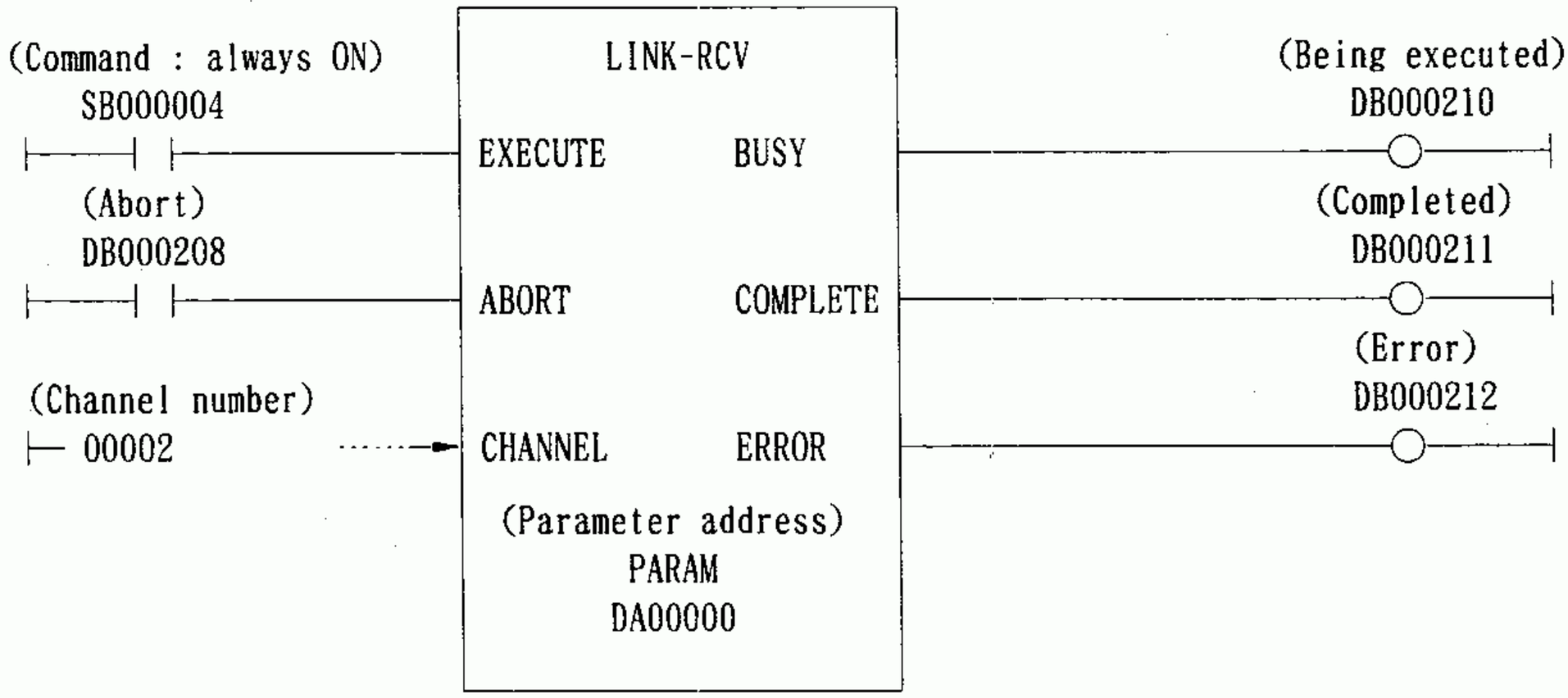
(2) COMPLETE (processing completed)
This goes "ON" only for one scan at normal termination.

(3) ERROR (error occurred)
This goes "ON" only for one scan at error occurrence.

See PARAM00 (processing result) and PARAM01 (LINK STATUS) for the error cause.

3.9.4 Sample programs

```
(Set the system register to 0 in the first scan)
    SB000003
|------| |------
[ |- 00000      ]                                          [     => DW00009 ]
```

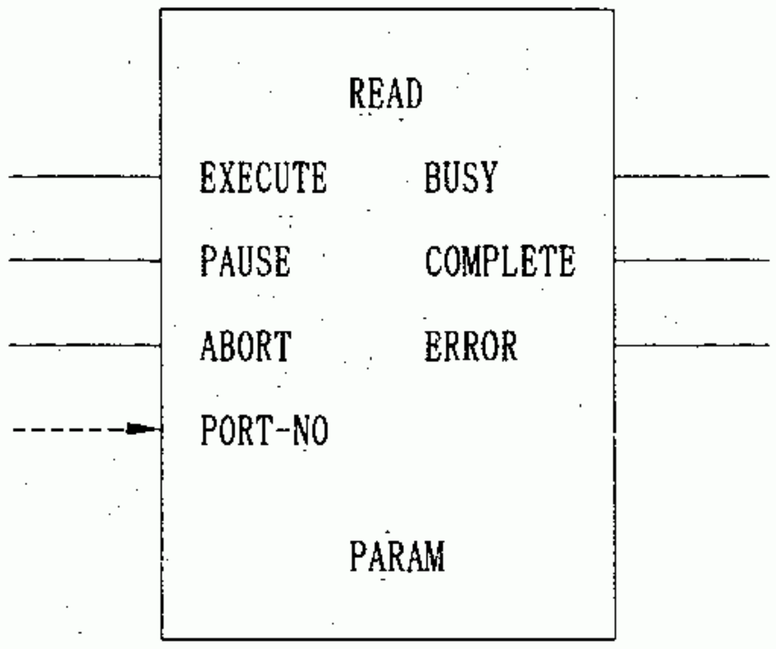



```
    DB000211
|------| |------
[  INC       DW00024 ]    (Pass counter)
    DB000212
|------| |------
IFON
INC      DW00025    (Error counter)
 |- DW00000         (Processing result saving)                    =>  DW00026
 |- DW00001         (LINK status saving)                          =>  DW00027
IEND
DEND
```

3.10 ASCII INPUT FUNCTION

Function name	READ	Use restriction	
Function	Enters data from ASCII equipment (keyboard, bar code reader, etc.) and stores the data into M register according to the format specified in a message. Enters maximum data of 57 words in each scan for several scans. Low-speed or high-speed scan can be used.		

Function I/F



I/O

PARAM

No.	I/O	Content
00	OUT	Processing result
01	OUT	ASCII STATUS
02	IN	Line number
03	IN	Message number
04	IN	Data address
05	OUT	Number of required registers
06	OUT	Number of previously input data
07	SYS	Reserved for system
08	SYS	Reserved for system
09		
10		

I/O

I/O	TYPE	Name	Content
Input	BIT	EXECUTE	Input execute command
	BIT	PAUSE	Input stop command
	BIT	ABORT	Input forced stop command
	INT	PORT-NO	Input port number
	.....		
	.....		
Output	BIT	BUSY	Processing in progress
	BIT	COMPLETE	Processing completed
	BIT	ERROR	Error occurred
	.....		
	.....		

3.10.1 Parameters

(1) Processing result (PARAM00)

The processing result is output to the upper byte. The lower byte is reserved for system analysis.

00XX ····· Processing in progress (BUSY)
10XX ····· Processing completed (COMPLETE)
8XXX ····· Error occurred (ERROR)

〔Error classification〕

81XX ····· Message number error
An attempt was made to send an unused message number.

82XX ····· Address setup error
The data address set up is out of range.

84XX ····· Port number setup error
The port number 17 or higher number is set up.

88XX ····· ASCII error
A specified RIOD is not mounted on the main unit. Or an error response is returned from ASCII module. See ASCII STATUS.

(2) ASCII STATUS (PARAM01)

The ASCII module status is output.

〔BIT#〕		〔STATUS〕
15	·····	Reserved for future use.
14	·····	Reserved for future use.
13	·····	A transmission error occurred between ASCII equipment and ASCII slave station.
12	·····	An overrun error in the buffer of ASCII slave station
11	·····	Invalid format data in the execution message
10	·····	RIOD not mounted on the main unit
9	·····	A specified message missing in the ASCII slave station
8	·····	An ASCII slave station with a specified port missing on the remote side
7	·····	Port number 17 or higher number
6	·····	RIOD with a relevant ASCII slave station not mounted on the main unit.
5	·····	An ASCII message being programmed
4	·····	} Port number
3	·····	
2	·····	
1	·····	
0	·····	

(3) Line number (PARAM02)

Specifies the line number of RIOD to which an ASCII module is connected.

Preset value : 1, 2

(4) Message number (PARAM03)

Specifies a message number programmed in the ASCII module.

Preset value : 1 to 1024 (1 to 0400H)

(5) Data address (PARAM04)

The top word address of data to be output is set.

Preset value : 0 to 16383 (0 to 3FFFH)

(6) Number of required registers (PARAM05)

The number of registers (number of words) programmed in the ASCII module is output.

Output value : 1 to 9999 (1 to 270FH)

(7) Number of previously output data (PARAM06)

Outputs number of data already input from the start of output to the present.

(8) Reserved for system (PARAM07, PARAM08)

System work registers. Be sure to set them to 0000H by a user program in the first scan at power-on. After this, do not change the value by the user program since it is used by the system.

3.10.2 Input

(1) EXECUTE (input execute command)
When the command goes "ON", message input is performed.

This must be held until COMPLETE (processing completed) or ERROR (error occurred) goes "ON".

(2) PAUSE (input stop command)
Stops the input temporarily.

(3) ABORT (input forced stop command)
Stops the input by force. This overrides EXECUTE (input execute command).

(4) PORT-NO (input port number)
Specifies an input port number. Set a number from 1 to 16.

3.10.3 Output

(1) BUSY (processing in progress)
Processing being executed. Hold EXECUTE at "ON".

(2) COMPLETE (processing completed)
This goes "ON" only for one scan at normal termination.

(3) ERROR (error occurred)
This goes "ON" only for one scan at error occurrence.

See PARAM01 (processing result) and PARAM02 (ASCII STATUS) for the error cause.

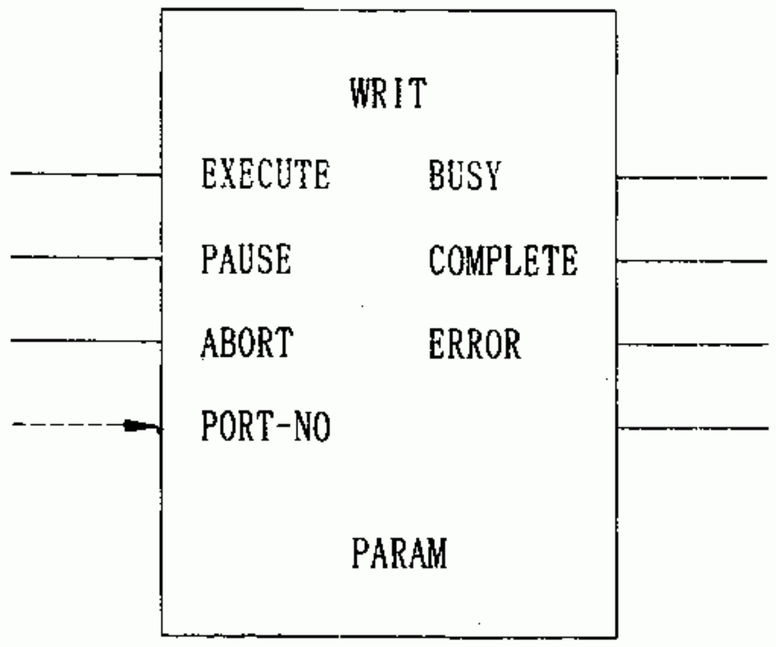
3.10.4 Sample programs

```
(Set the system register to 0 in the first scan)
   SB000003
|----| |----
[ |- 00000      ]                                          [      =>  DW00007 ]
                                                           [      =>  DW00008 ]

 (Always ON)   (Completed)    (Error)    (One-second delay    (Command)
                                           at start up)
  SB000004      DB000211      DB000212       SB000038          DB000201
|----| |----+------| |---------| |-----------------| |-----------( )----|
            |
(Hold command)
  DB000201  |
|----| |----+

                            (System function)
 (Command)              +------------------------+              (Busy)
  DB000201              |          READ          |             DB000210
|----| |----------------| EXECUTE          BUSY  |---------------( )----|
  (Pause)               |                        |            (Completed)
  DB000208              |                        |             DB000211
|----| |----------------| PAUSE        COMPLETE  |---------------( )----|
  (Abort)               |                        |              (Error)
  DB000209              |                        |             DB000212
|----| |----------------| ABORT           ERROR  |---------------( )----|
(Port number)           |                        |
|- 00001 ..........---->| PORT-NO                |
                        |  (Parameter address)   |
                        |         PARAM          |
                        |        DA00000         |
                        +------------------------+

   DB000211
|----| |----
[  INC        DW00024 ]     (Pass counter)

   DB000212
|----| |----
IFON
INC      DW00025      (Error counter)
 |- DW00000           (Processing result saving)                  =>  DW00026
 |- DW00001           (ASCII status saving)                       =>  DW00027
IEND
DEND
```

3.11 ASCII OUTPUT FUNCTION

Function name	WRIT	Use restriction	
Function	Outputs data in the M register to ASCII equipment (CRT, printer) according to the format specified in a message. Outputs maximum data of 57 words in each scan for several scans. Low-speed or high-speed scan can be used.		

Function I/F



I/O

PARAM

No.	I/O	Content
00	OUT	Processing result
01	OUT	ASCII STATUS
02	IN	Line number
03	IN	Message number
04	IN	Data address
05	OUT	Number of required registers
06	OUT	Number of previously output data
07	SYS	Reserved for system
08	SYS	Reserved for system
09		
10		

I/O

I/O	TYPE	Name	Content
Input	BIT	EXECUTE	Output execute command
	BIT	PAUSE	Output stop command
	BIT	ABORT	Output forced stop command
	INT	PORT-NO	Output port number
	.....		
	.....		
Output	BIT	BUSY	Processing in progress
	BIT	COMPLETE	Processing completed
	BIT	ERROR	Error occurred
	.....		
	.....		

3.11.1 Parameters

(1) Processing result (PARAM00)

The processing result is output to the upper byte. The lower byte is reserved for system analysis.

00XX ····· Processing in progress (BUSY)
10XX ····· Processing completed (COMPLETE)
8XXX ····· Error occurred (ERROR)

〔Error classification〕

81XX ····· Message number error
An attempt was made to send an unused message number.

82XX ····· Address setup error
The data address set up is out of range.

84XX ····· Port number setup error
The port number 17 or higher number is set up.

88XX ····· ASCII error
A specified RIOD is not mounted on the main unit. Or an error response is returned from ASCII module. See ASCII STATUS.

(2) ASCII STATUS (PARAM01)

The ASCII module status is output.

[BIT#]		[STATUS]
15	·····	Reserved for future use.
14	·····	Reserved for future use.
13	·····	A transmission error occurred between ASCII equipment and ASCII slave station.
12	·····	An overrun error in the buffer of ASCII slave station
11	·····	Invalid format data in the execution message
10	·····	RIOD not mounted on the main unit
9	·····	A specified message missing in the ASCII slave station
8	·····	An ASCII slave station with a specified port missing on the remote side
7	·····	Port number 17 or higher number
6	·····	RIOD with a relevant ASCII slave station not mounted on the main unit.
5	·····	An ASCII message being programmed
4	·····	Port number
3	·····	
2	·····	
1	·····	
0	·····	

(3) Line number (PARAM02)
Specifies the line number of RIOD to which an ASCII module is connected.

Preset value : 1, 2

(4) Message number (PARAM03)
Specifies a message number programmed in the ASCII module.

Preset value : 1 to 1024 (1 to 0400H)

(5) Data address (PARAM04)
Sets the top word address of data to be input.

Preset value : 0 to 16383 (0 to 3FFFH)

(6) Number of required registers (PARAM05)
Outputs the number of registers (number of words) programmed in the ASCII module.

Output value : 1 to 9999 (1 to 270FH)

(7) Number of previously input data (PARAM06)
Outputs the number of data already input from the start of input to the present.

(8) Reserved for system (PARAM07, PARAM08)
System work registers. Be sure to set them to 0000H by a user program in the first scan at power-on. After this, do not change the value by the user program since it is used by the system.

3.11.2 Input

(1) EXECUTE (output execute command)
When the command goes "ON", message output is performed.

This must be held until COMPLETE (processing completed) or ERROR (error occurred) goes "ON".

(2) PAUSE (output stop command)
Stops the output temporarily.

(3) ABORT (output forced stop command)
Stops the output by force. This overrides EXECUTE (output execute command).

(4) PORT-NO (output port number)
Specifies an output port number. Set a number from 1 to 16.

3.11.3 Output

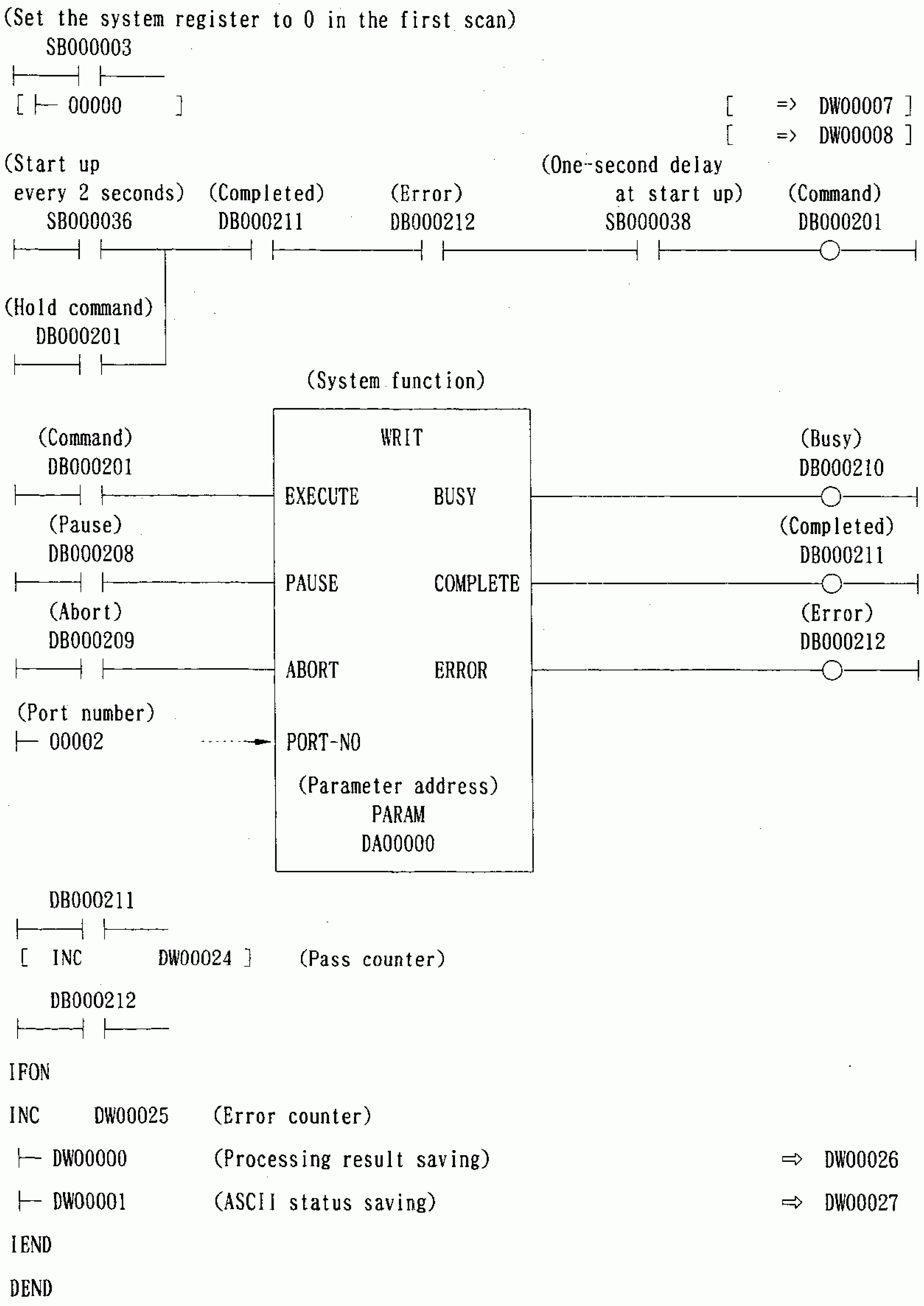
(1) BUSY (processing in progress)
Processing being executed. Hold EXECUTE at "ON".

(2) COMPLETE (processing completed)
This goes "ON" only for one scan at normal termination.

(3) ERROR (error occurred)
This goes "ON" only for one scan at error occurrence.

See PARAM01 (processing result) and PARAM02 (ASCII STATUS) for the error cause.

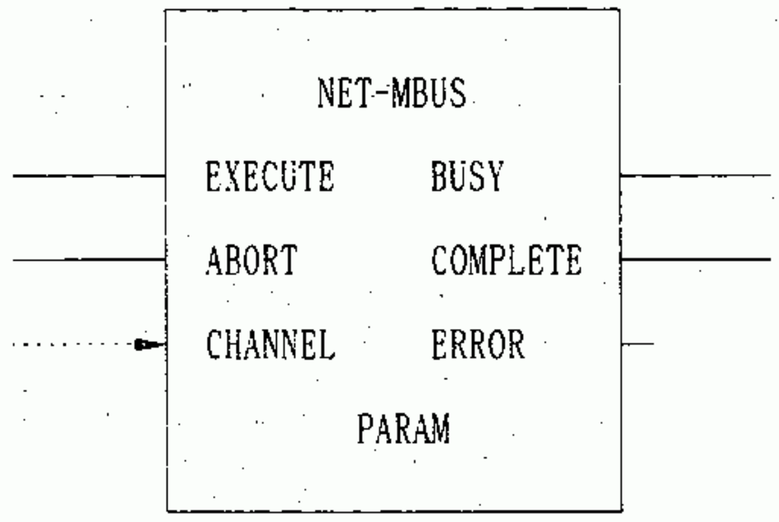
3.11.4 Sample programs



3.12 YENET MEMOBUS MESSAGE TRANSMISSION MASTER FUNCTION

Function name	NET-MBUS	Use restriction	
Function	Sends a MEMOBUS command to a slave and receives a MEMOBUS response from the destination. Reads/writes a coil, input relay, holding register, input register, etc. Complete station command transmission cannot be available. This function can be executed at low-speed or high-speed scan.		

Function I/F



I/O

PARAM

No.	I/O	Content
00	OUT	Processing result
01	OUT	YENET STATUS
02	IN	Destination NET#
03	IN	Destination NODE#
04	IN	FUNCTION CODE
05	IN	Data address
06	IN	Data size
07	IN	Offset
08	IN	Number of retries
09	IN	Timer
10	SYS	Reserved for the system

I/O

I/O	TYPE	Name	Content
Input	BIT	EXECUTE	Send execution command
	BIT	ABORT	Send forced stop command Stop sending by force by a user program
	INT	CHANNEL	Send channel number
	.....		
	.....		
Output	BIT	BUSY	Processing in progress
	BIT	COMPLETE	Processing completed
	BIT	ERROR	Error occurred
	.....		
	.....		

3.12.1 Parameters

(1) Processing result (PARAM00)

The processing result is output to the upper byte. The lower byte is reserved for system analysis.

00XX ····· Processing in progress (BUSY)

10XX ····· Processing completed (COMPLETE)

8XXX ····· Error occurred (ERROR)

〔Error classification〕

81XX ····· Function code error

An attempt was made to send an undefined function code or such a code was received.

82XX ····· Address setup error

The data address or offset set up is out of range.

83XX ····· Data size error

The send or receive data size is out of range.

84XX ····· Channel number setup error.

The channel number set up is 17 or more.

88XX ····· YENET error

An error response is returned from YENET. See YENET STATUS.

(2) YENET STATUS (PARAM01)

The YENET module status is output.

〔BIT#〕	〔STATUS〕
15	Busy
14	Wait
13	Complete
12	Error response receiving
11	Receive time overflow
10	Send time overflow
9	0 Normal / 1 Upper loopback
8	0 Lower loopback / 1 Loop error
7	Address error
6	Function error
5	Reference error
4	Parameter error
3	Buffer size error
2	Number of retries (0 to 7)
1	
0	

(3) Destination NET# (PARAM02)

Destination station node network address is set up.

00 :Self network

1 to 126 (007EH) :Specified network

(4) Destination NODE# (PARAM03)

Destination station node address is set up. When the destination station is NSB, NSB node address is set up. When the destination station is NSU, device address connected to NSU plus 1000 is set up.

NSB : 1 to 126 (0001H to 007EH)

USU : 1001 to 1247 (03E9H to 04DFH)

In case the destination address is NSU, relationship between device address and NSU node address is as follows:

Device address = { 2× (NSU node address− 1) } + Port No.

Port No.: 1 or 2 (NSU RS-232C Port)

(5) FUNCTION CODE (PARAM04)

The function codes to be sent are set up.

<Function code> (hex)		<CP-3300 used register> (Master)	(Slave)
1	: Read coil status	MB	MB
2	: Read input relay status	MB	IB
3	: Read holding register content	MW	MW
4	: Read input register content	MW	IW
F	: Write coil	MB	MB
10	: Write holding register	MW	MW

(6) Data address (PARAM05)

The top address of requested data to be read or written is set up. A bit address is set up for a coil and relay and a word address for a register. The setup range differs depending on the function code.

<Function code>	<Data size address range >
1 : Read coil status	0 to 65535 (0 to FFFFH) : Bit address
2 : Read input relay status	0 to 65535 (0 to FFFFH) : Bit address
3 : Read holding register content	0 to 16383 (0 to 3FFFH) : Word address
4 : Read input register content	0 to 16383 (0 to 3FFFH) : Word address
F : Write coil	0 to 65535 (0 to FFFFH) : Bit address
10 : Write holding register	0 to 16383 (0 to 3FFFH) : Word address

(7) Data size (PARAM06)

The read or write requested data size (number of bits or words) is set up. The setup range differs depending on the function code.

<Function code>	<Data size setup range >
1 : Read coil	0 to 2000 (1 to 07D0H) : Number of bits
2 : Read input relay status	0 to 2000 (1 to 07D0H) : Number of bits
3 : Read holding register content	0 to 125 (1 to 007DH) : Number of words
4 : Read input register content	0 to 125 (1 to 007DH) : Number of words
F : Write coil	0 to 800 (1 to 0320H) : Number of bits
10 : Write holding register	0 to 100 (1 to 0064H) : Number of words

(8) Offset (PARAM07)

Send / receive data offset word address is set up.

Set value : 0 to 16383 (0 to 3FFFH)

(9) Number of retries (PARAM08)

The number of retries at communication error occurrence is set up.

Set value : 1 to 7

0 - Retry is not performed

Communication error becomes time-out. Time-out is classified into two cases.

- Response from the destination station cannot be received.
- Sending becomes impossible since a token is lost.

Communications can be performed until "set value+1". In case sending is impossible or loop error occurs , retry is not performed.

(10) Timer (PARAM09)

Time until receiving the response from the destination station (receive waiting time) is set up.

Set value : 1 to 9999 (Unit : 100ms)

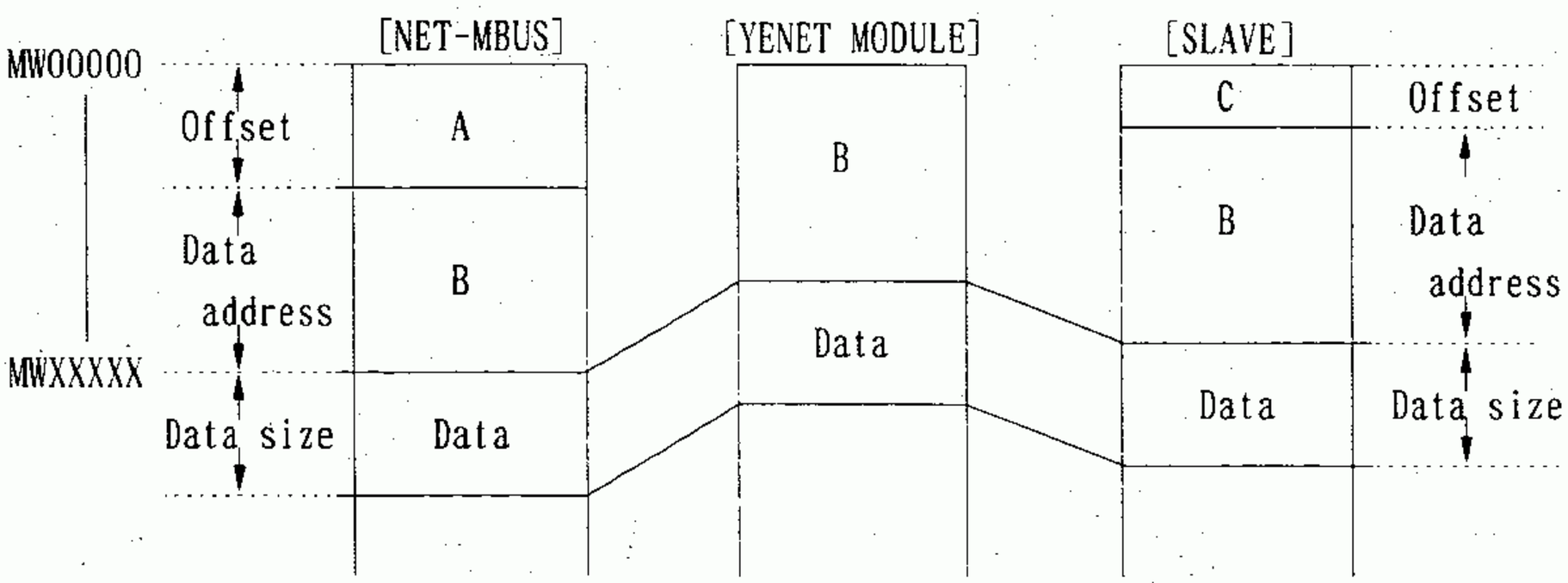
0 - Timer set is not performed. (undecided time)

Receive waiting time is started up after data is sent to LAN.

(11) Reserved for the system (PARAM10)

The current channel number used is held. Be sure to set this to 0000H by a user program at the first scan at the time of power-ON. After this , do not change the value by the user program since the system uses it.

(12) Relationship between data address, size and offset



A = sender offset address

B = sender data address

C = receiver offset address

3.12.2 Input

(1) EXECUTE (send execution command)
When the command goes "ON", a message is sent.

This must be held until COMPLETE (processing completed) or ERROR (error occurred) goes"ON".

(2) ABORT (send forced stop command)
Stops sending by force. This overrides EXECUTE (send execution command).

(3) CHANNEL(send channel number)
Specifies a send channel number. 1 to 16 is set.

3.12.3 Output

(1) BUSY (processing in progress)
Processing being executed. Hold EXECUTE at "ON".

(2) CONPLETE (processing completed)
This goes "ON" only for one scan at normal termination.

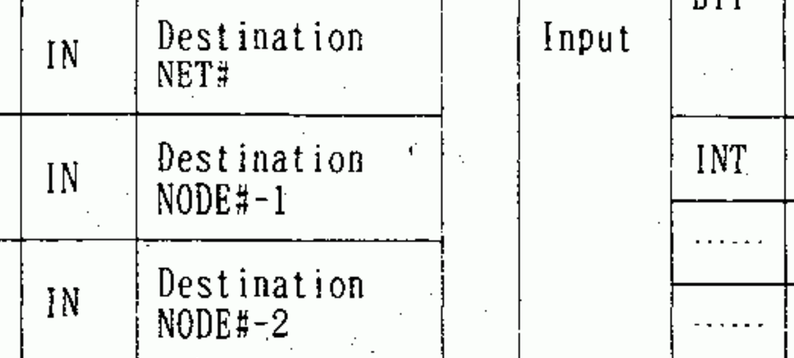
(3) ERROR (error occurred)
This goes "ON" only for one scan at occurrence of an error.

See PARAM01 (processing result) and PARAM02 (LINK STATUS) for the error cause.

3.13 YENET SELECTIVE BROADCASTING FUNCTION

Function name	NET-PEER	Use restriction	
Function	Sends the same data to the multiple (max. 8) parties' stations. Does not receive a response from the other party. Can be executed at low-speed or high-speed scan. The M register is the only register that can send data.		

Function I/F



I/O

PARAM

No.	I/O	Content
00	OUT	Processing result
01	OUT	YENET STATUS
02	IN	Destination NET#
03	IN	Destination NODE#-1
04	IN	Destination NODE#-2
05	IN	Destination NODE#-3
06	IN	Destination NODE#-4
07	IN	Destination NODE#-5
08	IN	Destination NODE#-6
09	IN	Destination NODE#-7
10	IN	Destination NODE#-8
11	IN	Data address
12	IN	Data size
13	IN	Offset
14	SYS	Reserved for the system

I/O

I/O	TYPE	Name	Content
Input	BIT	EXECUTE	Send execute command
	BIT	ABORT	Send forced stop command. Stop sending by force by a user program.
	INT	CHANNEL	Send channel number
	.....		
	.....		
Output	BIT	BUSY	Processing in progress
	BIT	COMPLETE	Processing completed
	BIT	ERROR	Error occurred
	.....		
	.....		

3.13.1 Parameters

(1) Processing result (PARAM00)

The processing result is output to the upper byte. The lower byte is reserved for system analysis.

00XX ····· Processing in progress (BUSY)
10XX ····· Processing completed (COMPLETE)
8XXX ····· Error occurred (ERROR)

〔Error classification〕

82XX ····· Address setting error
The data address and offset set up are out of range.
83XX ····· Data size error
The send data size is out of range.
84XX ····· Channel number setup error.
The channel number set up is 17 or more.
88XX ····· YENET error
A response is returned from YENET. See YENET STATUS.

(2) YENET STATUS (PARAM01)

The YENET module status is output.

〔BIT#〕		〔STATUS〕
15	———	Busy
14	———	Wait
13	———	Complete
12	———	Error response receiving
11	———	Receive time overflow
10	———	Send time overflow
9	———	0 0 } Normal 1 0 } Upper loopback 0 1 } Lower loopback 1 1 } Loop error
8	———	
7	———	Address error
6	———	Function error
5	———	Reference error
4	———	Parameter error
3	———	Buffer size error
2	———	Number of retries (0 to 7)
1	———	
0	———	

(3) Destination NET# (PARAM02)

Distination station node network address is set up.

00	:Self network
1 to 126 (007EH)	:Specified network

(4) Destination NODE#-n (PARAM03 to 10)

Destination station node address is set up.

00	:Destination does not exist
1 to 126 (007EH)	:Specified station

(5) Data address (PARAM11)

Data top word address required to be written is set up.

Set value : 0 to 16383 (0 to 3FFFH)

(6) Data size (PARAM12)

Data size (number of words) required to be written is set up.

Set value : 1 to 100 (1 to 0064H)

(7) Offset (PARAM13)

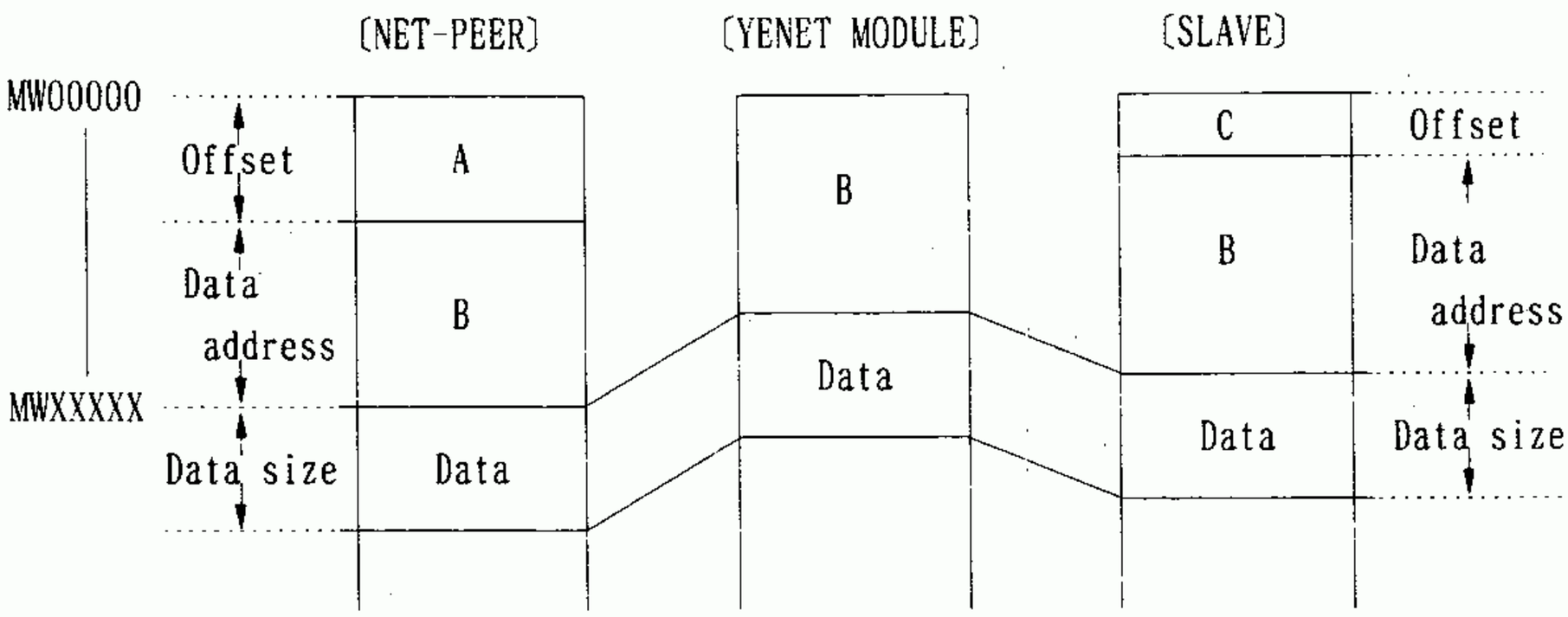
Send data offset word address is set up.

Set value : 0 to 16383 (0 to 3FFFH)

(8) Reserved for the system (PARAM14)

The current channel number used is held. Be sure to set this to 0000H by a user program at the first scan at the time of power-ON. After this , do not change the value by the user program since the system uses it.

(9) Relationship between data address, size and offset



A = sender offset address

B = sender data address

C = receiver offset address

3.13.2 Input

(1) EXECUTE (send execution command)

When the command goes "ON", a message is sent.

This must be held until COMPLETE (processing complated) or ERROR (error occurred) goes"ON".

(2) ABORT (send forced stop command)

Stops sending by force. This overrides EXECUTE (send execution command).

(3) CHANNEL(send channel number)

Specifies a send channel number. 1 to 16 is set.

3.13.3 Output

(1) BUSY (processing in progress)

Processing being executed. Hold EXECUTE at "ON".

(2) COMPLETE (processing completed)

This goes "ON" only for one scan at normal termination.

(3) ERROR (error occurred)

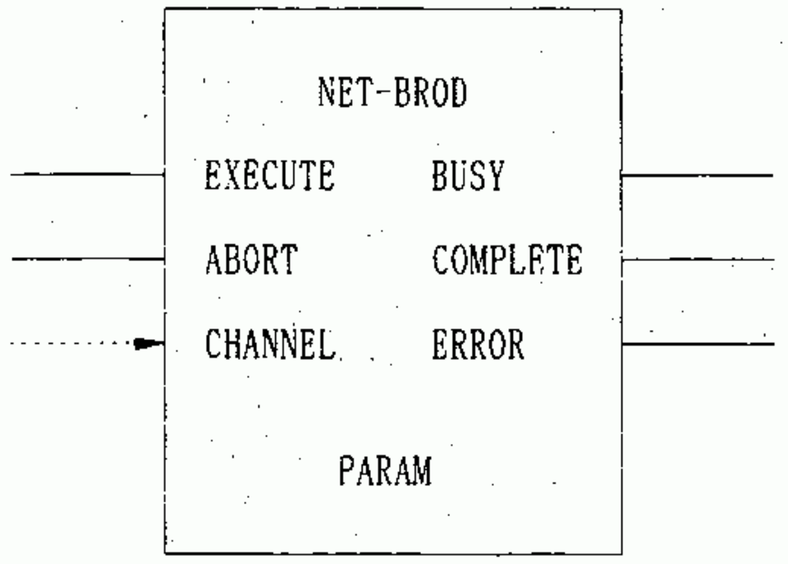
This goes "ON" only for one scan at occurrence of an error.

See PARAM01 (processing result) and PARAM02 (YENET STATUS) for the error cause.

3.14 YENET COMPLETE STATION BROADCASTING FUNCTION

Function name	NET-BROD	Use restriction	
Function	Executes the write function by broadcasting. Sends data to complete network stations except the device connected to NSU. Does not receive a response from the other party. Can be executed at low-speed or high-speed scan. The M register is the only register that can send data.		

Function I/F



I/O

PARAM

No.	I/O	Content
00	OUT	Processing result
01	OUT	YENET STATUS
02	IN	Destination NET#
03	SYS	System reserved
04	IN	FUNCTION CODE
05	IN	Data address
06	IN	Data size
07	IN	Offset
08	SYS	Reserved for the system
09		
10		

I/O

I/O	TYPE	Name	Content
Input	BIT	EXECUTE	Send execute command
	BIT	ABORT	Send forced stop command. Stop sending by force by a user program.
	INT	CHANNEL	Send channel number
	.....		
	.....		
Output	BIT	BUSY	Processing in progress
	BIT	COMPLETE	Processing completed
	BIT	ERROR	Error occurred
	.....		
	.....		

3.14.1 Parameters

(1) Processing result (PARAM00)

The processing result is output to the upper byte. The lower byte is reserved for system analysis.

00XX ····· Processing in progress (BUSY)
10XX ····· Processing completed (COMPLETE)
8XXX ····· Error occurred (ERROR)

〔Error classification〕

81XX ····· Function code error
An attempt was made to send an undefined function code or such a code was received.

82XX ····· Address setting error
The data address or offset setup is out of range.

83XX ····· Data size errorr
The send or receive data size is out of range.

84XX ····· Channel number setup error.
The channel number setup is 17 or more.

88XX ····· YENET error
An error response is returned from YENET. See YENET STATUS.

(2) YENET STATUS (PARAM01)

The YENET module status is output.

〔BIT#〕		〔STATUS〕
15	———	Busy
14	———	Wait
13	———	Complete
12	———	Error response receiving
11	———	Receive time overflow
10	———	Send time overflow
9	———	0 / 0 } Normal; 1 / 0 } Upper loopback; 0 / 1 } Lower loopback; 1 / 1 } Loop error
8	———	
7	———	Address error
6	———	Function error
5	———	Reference error
4	———	Parameter error
3	———	Buffer size error
2		
1	———	Number of retries (0 to 7)
0		

(3) Destination NET# (PARAM02)

Destination station node network address is set up.

00		:Self network
1 to 126	(007EH)	:Specified network
255	(007EH)	:Broadcasting to complete station

(4) System reserved (PARAM03)

Register for system reserving. Set 0.

(5) FUNCTION CODE (PARAM04)

A function code to be sent is set up.

<Function code> (hex)	<CP-3300 used register> (Master)	(Slave)
F : Write coil	MB	MB
10 : Write holding register	MW	MW

(6) Data address (PARAM05)

The top address of requested data to be written is set up. A bit address is set up for a coil and relay and a word address for a register. The setup range differs depending on the function code.

<Function code> (hex)	<Data address setup range>
F : Write coil	0 to 65535 (0 to FFFFH) : Bit address
10 : Write holding register	0 to 16383 (0 to 3FFFH) : Word address

(7) Data size (PARAM06)

The write requested data size (number of bits or words) is set up. The setup range differs depending on the function code.

<Function code> (hex)	<Data size setup range>
F : Write coil	1 to 800 (1 to 0320H) : Number of bits
10 : Write holding register	1 to 100 (1 to 0064H) : Number of words

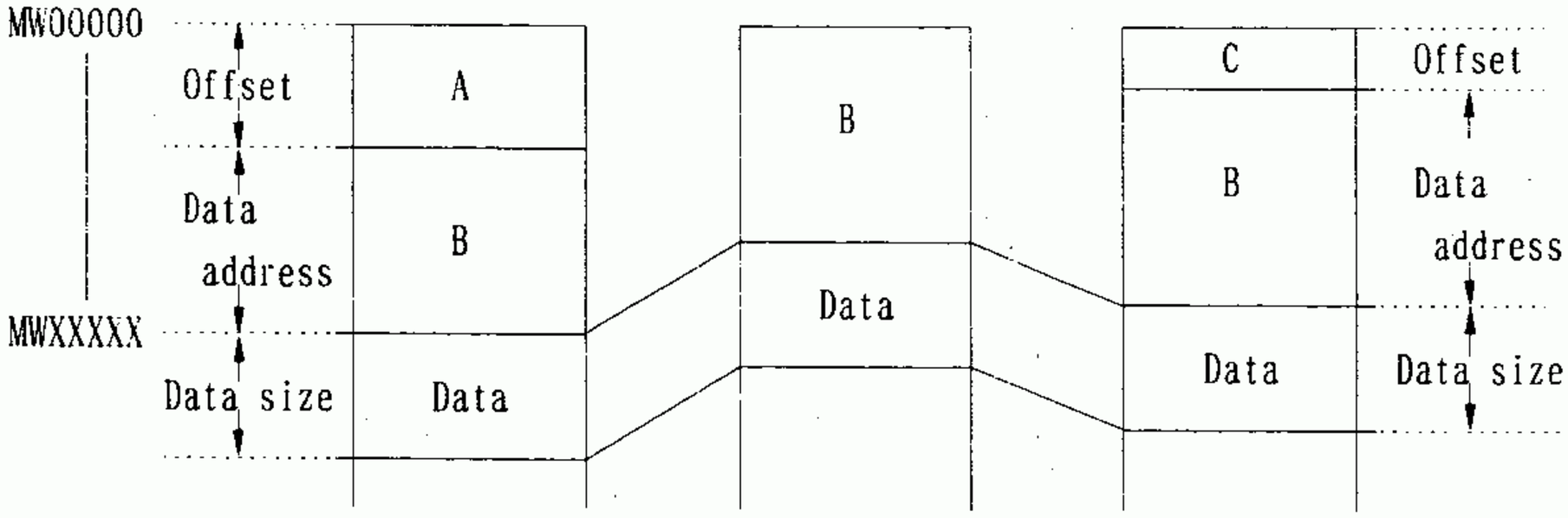
(8) Offset (PARAM07)

The send data offset word address is set up.

Set value : 0 to 16383 (0 to 3FFFH)

(9) Reserved for the system (PARAM08)
The current channel number used is held. Be sure to set this to 0000H by a user program at the first scan at the time of power-ON. After this , do not change the value by the user program since the system uses it.

(10) Relationship between data address, size and offset



A = sender offset address B = sender data address C = receiver offset address

3.14.2 Input

(1) EXECUTE (send execution command)
When the command goes "ON", a message is sent.
This must be held until COMPLETE (processing completed) or ERROR (error occurred) goes"ON".

(2) ABORT (send forced stop command)
Stops sending by force. This overrides EXECUTE (send execution command).

(3) CHANNEL(send channel number)
Specifies a send channel number. 1 to 16 is set.

3.14.3 Output

(1) BUSY (processing in progress)
Processing being executed. Hold EXECUTE at "ON".

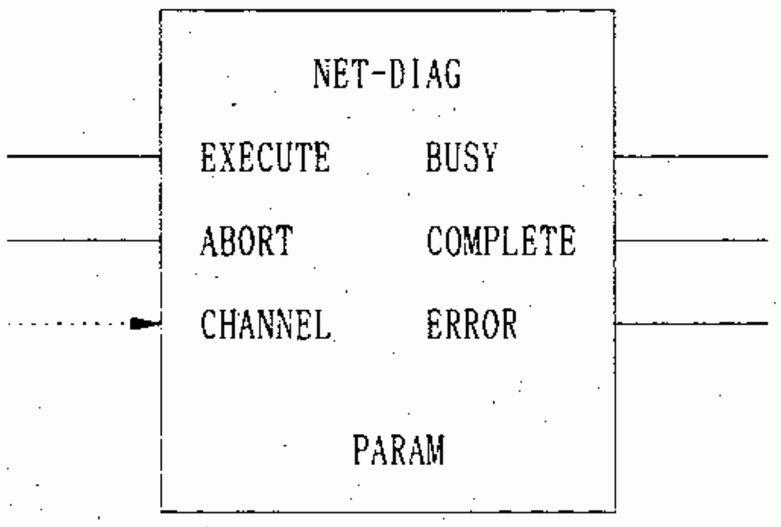
(2) COMPLETE (processing completed)
This goes "ON" only for one scan at normal termination.

(3) ERROR (error occurred)
This goes "ON" only for one scan at occurrence of an error.
See PARAM01 (processing result) and PARAM02 (YENET STATUS) for the error cause.

3.15 YENET NODE DIAGNOSIS FUNCTION

Function name	NET-DIAG	Use restriction	
Function	Makes a node diagnosis of the multiple (max. 8) parties' stations. Can be executed at low-speed or high-speed scan.		

Function I/F



I/O

PARAM

No.	I/O	Content
00	OUT	Processing result
01	OUT	YENET STATUS
02	IN	Destination NET #
03	IN	Destination NODE #-1
04	IN	Destination NODE #-2
05	IN	Destination NODE #-3
06	IN	Destination NODE #-4
07	IN	Destination NODE #-5
08	IN	Destination NODE #-6
09	IN	Destination NODE #-7
10	IN	Destination NODE #-8
11	IN	Timer
12	OUT	Diagnosis result
13	SYS	Reserved for the system

I/O

I/O	TYPE	Name	Content
Input	BIT	EXECUTE	Send execute command
	BIT	ABORT	Send forced stop command. Stop sending by force by a user program.
	INT	CHANNEL	Send channel number
	.....		
	.....		
Output	BIT	BUSY	Processing in progress
	BIT	COMPLETE	Processing completed
	BIT	ERROR	Error occurred
	.....		
	.....		

3.15.1 Parameters

(1) Processing result (PARAM00)

The Processing result is output to the upper byte. The lower byte is reserved for system analysis.

00XX ····· Processing in progress (BUSY)
10XX ····· Processing completed (COMPLETE)
8XXX ····· Error occurred (ERROR)

〔Error classification〕

84XX ····· Channel number setting error
Channel number is set to 17 or more.
88XX ····· YENET error
A response is returned from YENET. See YENET STATUS.

(2) YENET STATUS (PARAM01)

The YENET module status is output.

〔BIT#〕		〔STATUS〕
15	——	Busy
14	——	Wait
13	——	Complete
12	——	Error response receiving
11	——	Receive time overflow
10	——	Send time overflow
9	——	0 0 } Normal; 1 0 } Upper loopback; 0 1 } Lower loopback; 1 1 } Loop error
8		
7	——	Address error
6	——	Function error
5	——	Reference error
4	——	Parameter error
3	——	Buffer size error
2	——	Number of retries (0 to 7)
1		
0		

(3) Destination NET# (PARAM02)

Destination station node network address is set up.

00 :Self network

1 to 126 (007EH) :Specified network

(4) Destination NODE#-n (PARAM03 to 10)

Destination station node address is set up. When the destination station is NSB , NSB node address is set up. When the destination station is NSU , device address connected to NSU plus 1000 is set up.

NSB : 1 to 126 (0001H to 007EH)

NSU : 1001 to 1247 (03E9H to 04DFH)

0 : Destination does not exist

In case the destination address is NSU , relationship between device address and NSU node address is as follows:

Device address = { 2× (NSU node address− 1) } + Port No.

Port No.: 1 or 2 (NSU RS-232C Port)

(5) Timer (PARAM11)

Time until receiving the response from the destination station (receive waiting time) is set up.

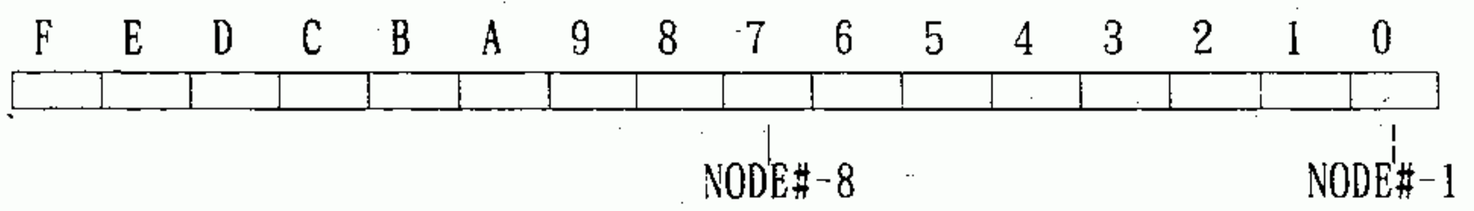
Set value : 1 to 9999 (Unit : 100ms)

0 - Timer set is not performed.(undecided time)

Receive waiting time is started up after data is sent to LAN.

(6) Diagnosis result (PARAM12)

Diagnosis result of specified station is output in bits. NODE#-1 to NODE#-8 corresponds to BIT#0 to BIT#7 respectively.



1 … No response / Without diagnosis specification, 0…Normal response

(7) Reserved for the system (PARAM13)

The current channel number used is held. Be sure to set this to 0000H by a user program at the first scan at the time of power-ON. After this , do not change the value by the user program since the system uses it.

3.15.2 Input

(1) EXECUTE (send execution command)
When the command goes "ON", a message is sent.
This must be held until COMPLETE (processing complated) or ERROR (error occurred) goes"ON".

(2) ABORT (send forced stop command)
Stops sending by force. This overrides EXECUTE (send execution command).

(3) CHANNEL(send channel number)
Specifies a send channel number. 1 to 16 is set.

3.15.3 Output

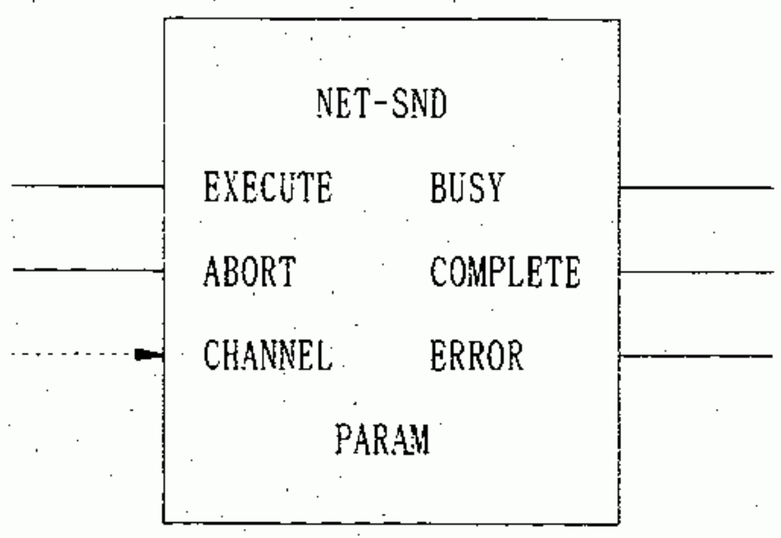
(1) BUSY (processing in progress)
Processing being executed. Hold EXECUTE at "ON".

(2) COMPLETE (processing completed)
This goes "ON" only for one scan at normal termination.
(3) ERROR (error occurred)
This goes "ON" only for one scan at occurrence of an error.
See PARAM01 (processing result) and PARAM02 (YENET STATUS) for the error cause.

3.16 YENET DATA SEND FUNCTION

Function name	NET-SND	Use restriction	
Function	Sends the data to the destination station. Does not receive a response from the other party. Complete station broadcasting is available. Communicates with a controller furnished with a communication facility of SUMINET-3200 UDP(datagram). Normally used with a pair of NET-RCV functions. Can be executed at low-speed or high-speed scan. The M register is the only register that can send data.		

Function I/F



I/O

PARAM

No.	I/O	Content
00	OUT	Processing result
01	OUT	YENET STATUS
02	IN	Destination NET#
03	IN	Destination NODE#
04	IN	Data address
05	IN	Data size
06	SYS	Reserved for the system
07		
08		
09		
10		

I/O

I/O	TYPE	Name	Content
Input	BIT	EXECUTE	Send execute command
	BIT	ABORT	Send forced stop command Stop sending by force by a user program
	INT	CHANNEL	Send channel number
	.....		
	.....		
Output	BIT	BUSY	Processing in progress
	BIT	COMPLETE	Processing completed
	BIT	ERROR	Error occurred
	.....		
	.....		

3.16.1 Parameters

(1) Processing result (PARAM00)

The processing result is output to the upper byte. The lower byte is reserved for system analysis.

00XX ····· Processing in progress (BUSY)
10XX ····· Processing completed (COMPLETE)
8XXX ····· Error occurred (ERROR)

〔Error classification〕

82XX ····· Address setting error
Data address setting is out of range.
83XX ····· Data size error
Send data size is out of range.
84XX ····· Channel number setting error.
Channel number is set to 17 or more.
88XX ····· YENET error
A response is returned from YENET. See YENET STATUS.

(2) YENET STATUS (PARAM01)

The YENET module status is output.

〔BIT#〕		〔STATUS〕
15	———	Busy
14	———	Wait
13	———	Complete
12	———	Error response receiving
11	———	Receive time overflow
10	———	Send time overflow
9	———	0 0 } Normal; 1 0 } Upper loopback; 0 0 } Lower loopback; 1 1 } Loop error
8	———	
7	———	Address error
6	———	Function error
5	———	Reference error
4	———	Parameter error
3	———	Buffer size error
2		Number of retries (0 to 7)
1	———	
0		

(3) Destination NET# (PARAM02)

Destination station node network address is set up.

00		:Self network
1 to 126	(007EH)	:Specified network
255	(007EH)	:Broadcasting to Complete station

(4) Destination NODE# (PARAM03)

Destination station node address is set up.

1 to 126	(0001H to 007EH)	: Specified station
255	(00FFH)	: Broadcasting to Complete station

(5) Data address (PARAM04)

Send data top word address is set up.

Set value : 0 to 16383 (0 to 3FFFH)

(6) Data size (PARAM05)

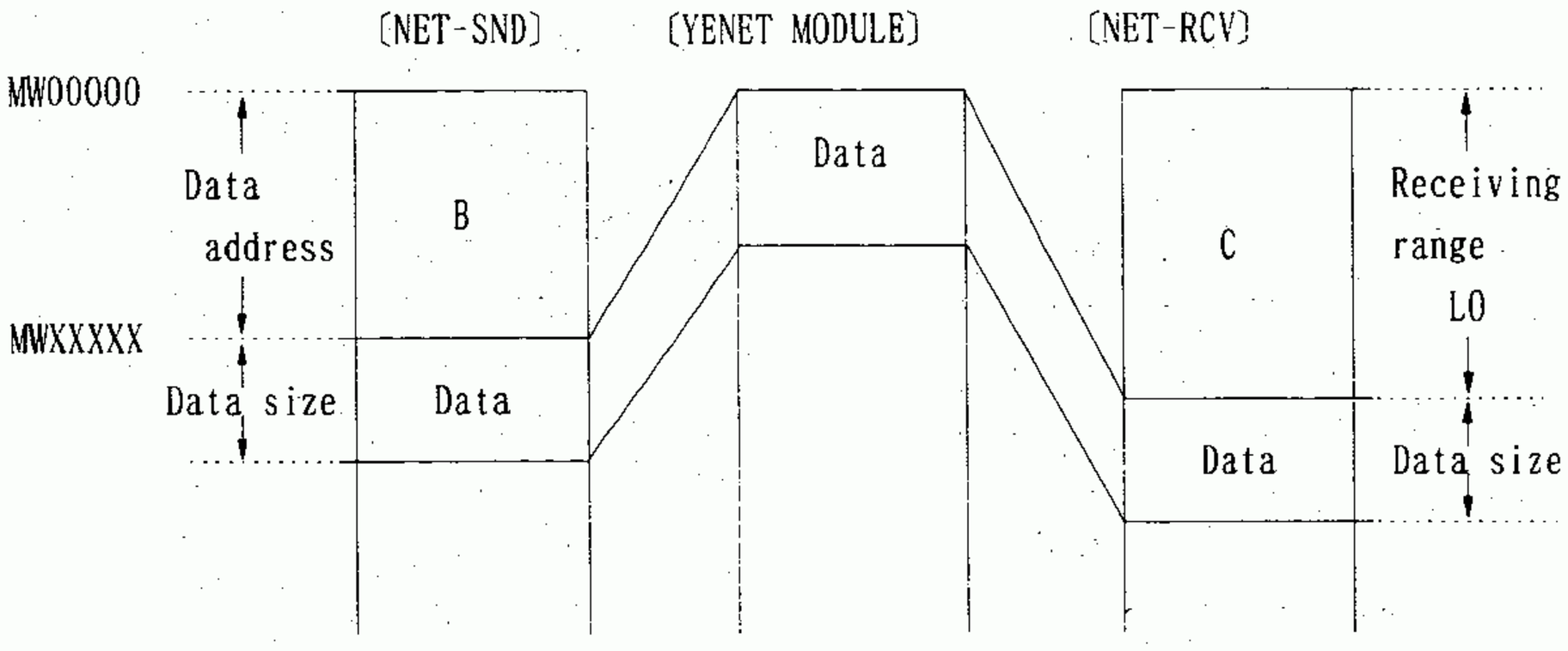
Send data size (number of words) is set up.

Set value : 1 to 250 (0 to 00FAH)

(7) Reserved for the system (PARAM06)

The current channel number used is held. Be sure to set this to 0000H by a user program at the first scan at the time of power-ON. After this , do not change the value by the user program since the system uses it.

(8) Relationship between data address and size.



B = Sender data address

C = Receiving range (LO)

3.16.2 Input

(1) EXECUTE (send execution command)
When the command goes "ON", a message is sent.
This must be held until COMPLETE (processing completed) or ERROR (error occurred) goes"ON".

(2) ABORT (send forced stop command)
Stops sending by force. This overrides EXECUTE (send execution command).

(3) CHANNEL(send channel number)
Specifies a send channel number. 1 to 16 is set.

3.16.3 Output

(1) BUSY (processing in progress)
Processing being executed. Hold EXECUTE at "ON".

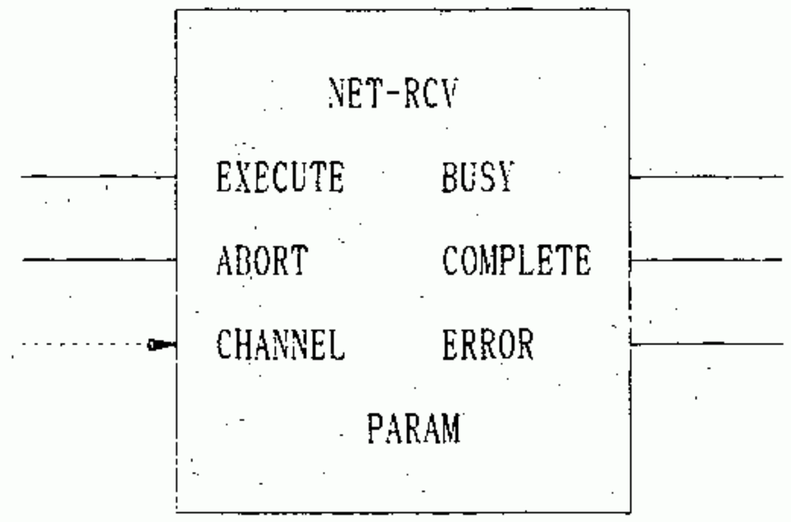
(2) COMPLETE (processing completed)
This goes "ON" only for one scan at normal termination.

(3) ERROR (error occurred)
This goes "ON" only for one scan at occurrence of an error.
See PARAM01 (processing result) and PARAM02 (YENET STATUS) for the error cause.

3.17 YENET DATA RECEIVE FUNCTION

Function name	NET-RCV	Use restriction	
Function	Receives the data from the destination station. Normally used with a pair of NET-RCV functions, while can receive the data from the other controller. However sender controller requires the communication facility of SUMINET-3200 UDP (datagram). Can be executed at low-speed or high-speed scan. The M register is the only register that can receive data.		

Function I/F



I/O

PARAM

No.	I/O	Content
00	OUT	Processing result
01	OUT	YENET STATUS
02	IN	Destination NET#
03	IN	Destination NODE#
04	SYS	System reserved
05	OUT	Data size
06	IN	Receiving range LO
07	IN	Receiving range HI
08	IN	Timer
09	SYS	Reserved for the system
10		

I/O

I/O	TYPE	Name	Content
Input	BIT	EXECUTE	Send execute command
	BIT	ABORT	Send forced stop command Stop sending by force by a user program
	INT	CHANNEL	Receive channel number
	.....		
	.....		
Output	BIT	BUSY	Processing in progress
	BIT	COMPLETE	Processing completed
	BIT	ERROR	Error occurred
	.....		
	.....		

3.17.1 Parameters

(1) Processing result (PARAM00)

The processing result is output to the upper byte. The lower byte is reserved for system analysis.

00XX ····· Processing in progress (BUSY)
10XX ····· Processing completed (COMPLETE)
8XXX ····· Error occurred (ERROR)

〔Error classification〕

82XX ····· Address setting error
Receive range setting is out of range.
83XX ····· Data size error
Receive data size is out of range
84XX ····· Channel number setting error.
Channel number is set to 7 or more.
88XX ····· YENET error
A response is returned from YENET. See YENET STATUS.

(2) YENET STATUS (PARAM01)

The YENET module status is output.

〔BIT#〕	〔STATUS〕
15	Busy
14	Wait
13	Complete
12	Error response receiving
11	Receive time overflow
10	Send time overflow
9, 8	0,0 Normal / 1,0 Upper loopback / 0,0 Lower loopback / 1,1 Loop error
7	Address error
6	Function error
5	Reference error
4	Parameter error
3	Buffer size error
2, 1, 0	Number of retries (0 to 7)

(3) Destination NET# (PARAM02)

The send source, station node network address is set up.

00 :Self network

1 to 126 (007EH) :Specified network

(4) Destination NODE# (PARAM03)

The send source, station node address is set up.

1 to 126 (007EH) : Specified station

(5) System reserved (PARAM04)

Register for system reserving. Set 0.

(6) Data size (PARAM05)

The data size requested by the sender is output.

(7) Receiving range LO(PARAM06), Receiving range HI(PARAM07)

Write-in allowable range for receiving request is set up. If request is out of this range, it becomes an error. Receiving range LO is also used as receive data write-in top word address.

0 ≦ Receiving range LO ≦ Receiving range HI ≦ 16383

(8) Timer (PARAM08)

Time until receiving the response from the destination station (receive waiting time) is set up.

Set value : 1 to 9999 (Unit : 100ms)

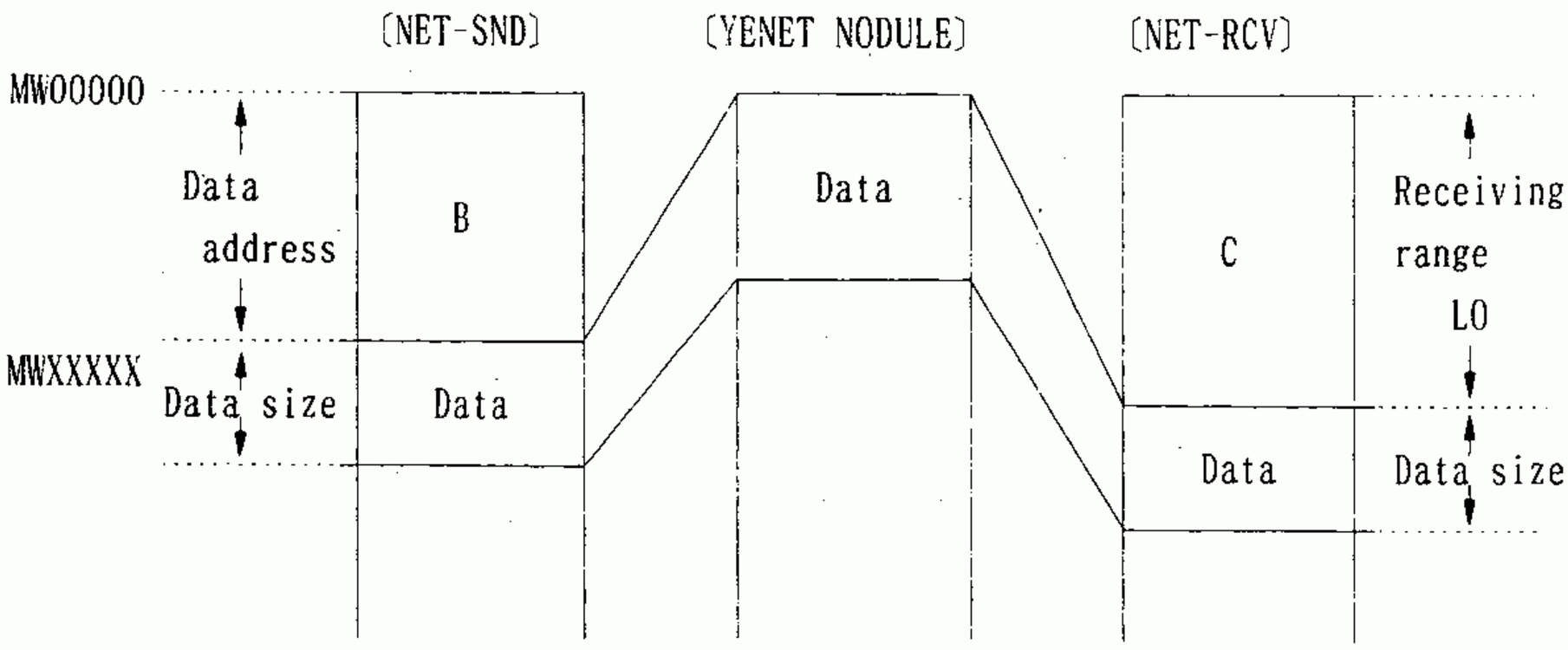
0 - Timer set is not performed.(undecided time)

Receive waiting time is started up after data is sent to LAN.

(9) Reserved for the system (PARAM09)

The current channel number used is held. Be sure to set this to 0000H by a user program at the first scan at the time of power-ON. After this , do not change the value by the user program since the system uses it.

(10) Relationship between data address and size.



B = Sender data address

C = Receiving range (LO)

3.17.2 Input

(1) EXECUTE (send execution command)

When the command goes "ON", a message is received.

This must be held until COMPLETE (processing completed) or ERROR (error occurred) goes"ON".

(2) ABORT (send forced stop command)

Stops sending by force. This overrides EXECUTE (receive execution command).

(3) CHANNEL (send channel number)

Specifies a receive channel number. 1 to 16 is set.

3.17.3 Output

(1) BUSY (processing in progress)

Processing being executed. Hold EXECUTE at "ON".

(2) COMPLETE (processing completed)

This goes "ON" only for one scan at normal termination.

(3) ERROR (error occurred)

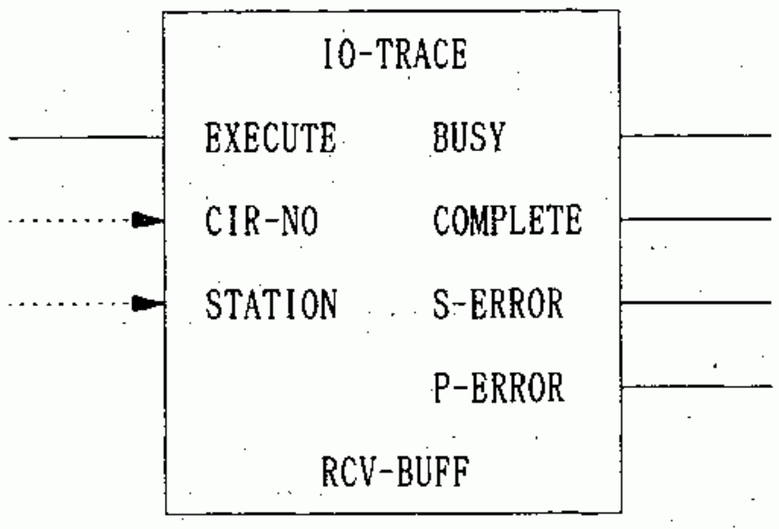
This goes "ON" only for one scan at occurrence of an error.

See PARAM01 (processing result) and PARAM02 (YENET STATUS) for the error cause.

3.18 I/O TRACE FUNCTION

Function name	IO-TRACE	Use restriction	
Function	When EXECUTE is ON, CP-213 slave trace data (specified by CIR-NO and STATION) are read out. The read out trace data are stored to the registers (specified by RCV-BUFF) in order of read-out.		

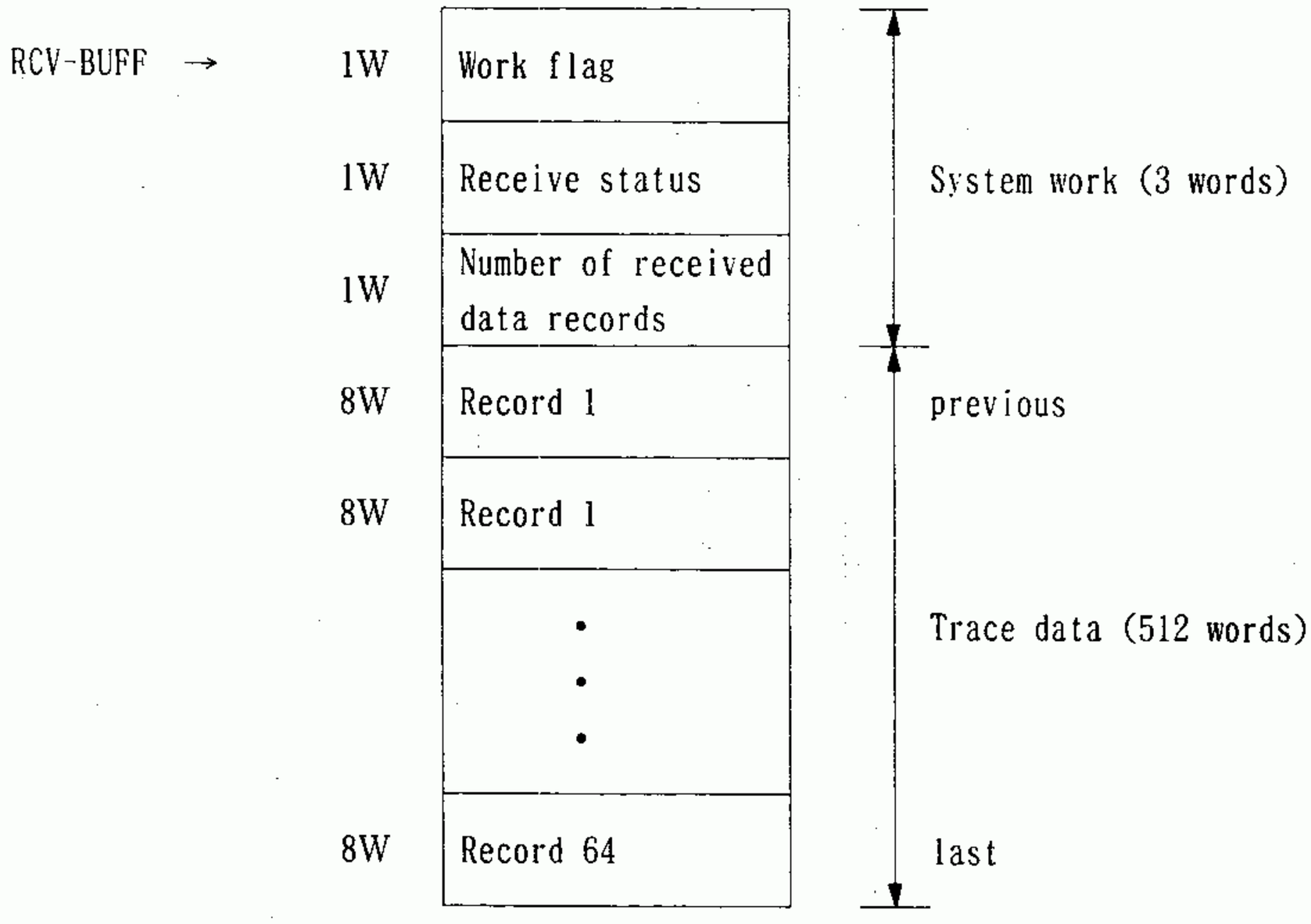
Function I/F



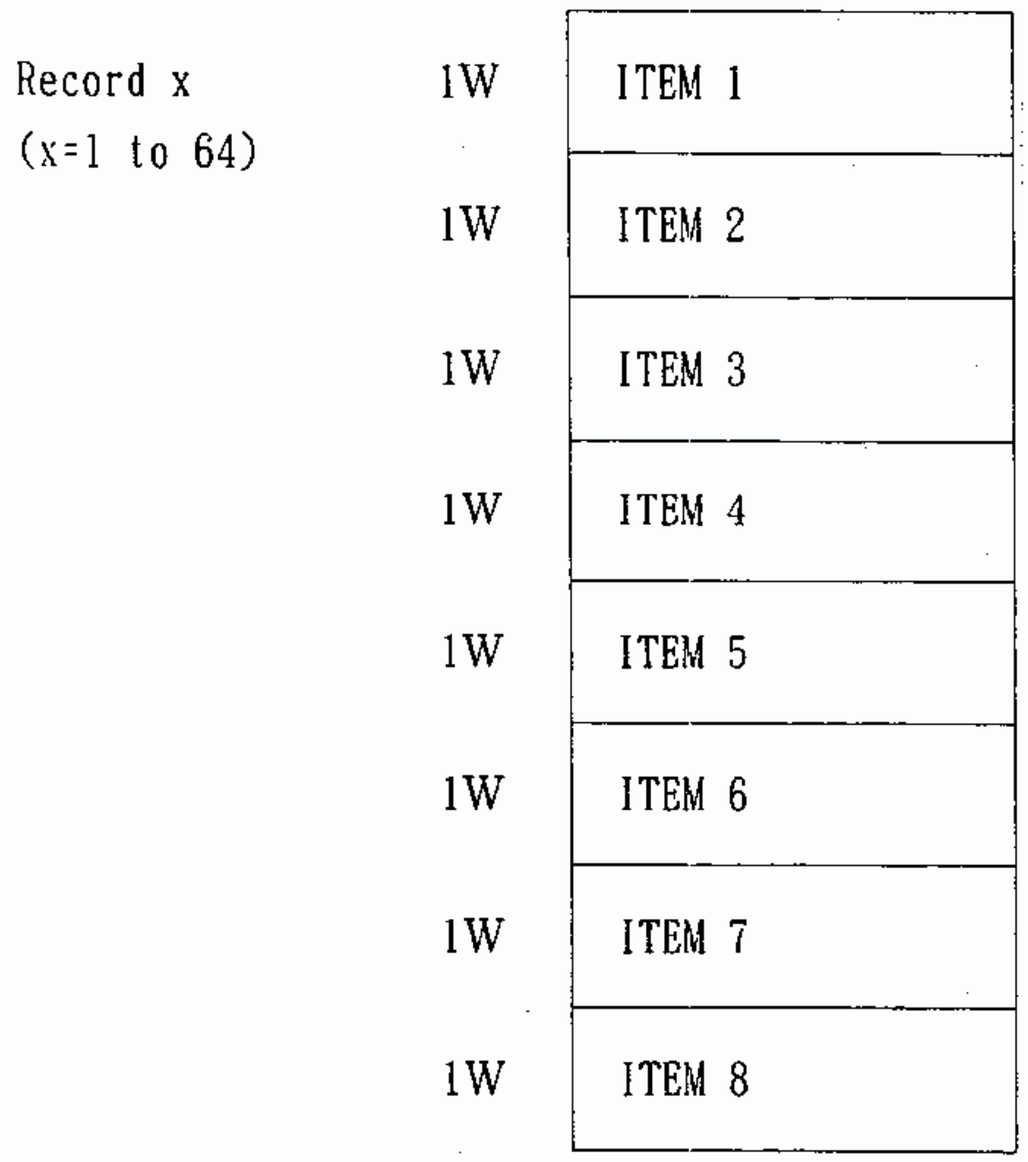
	No.	Name	Type	Content	Remarks
Input	1	EXECUTE	BIT	Execution command	Holds during operation
	2	CIR-NO	INT	CP-213 line number	0 to 3
	3	STATION	INT	Slave station number	0 to 31
	4	RCV-BUFF	ADRS	Top address of receiving buffer	
Output	1	BUSY	BIT	Processing in progress	
	2	COMPLATE	BIT	Processing completed	
	3	S-ERROR	BIT	Transmission error	
	4	P-ERROR	BIT	Parameter error	

3.18.1 Receiving buffer

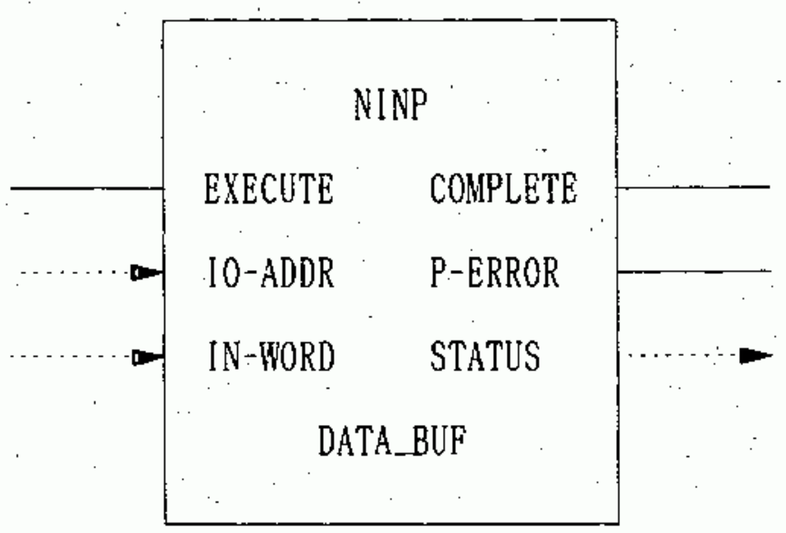
(1) Receiving buffer structure



(2) Record structure

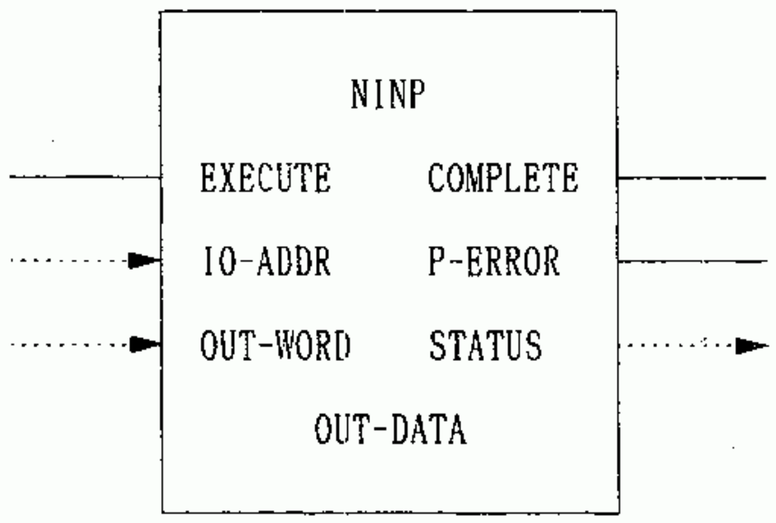


3.19 DIRECT INPUT FUNCTION

Function name	NINP	Use restriction	
Function	This function executes data input of prural words collectively from a local input module, in user program independently of system I/O. • When EXECUTE is ON, the data (specified by IN-WORD) are input continuously from local input module (specified by IO-ADDR). • Input data are stored to the specified area DATA-BUF. • COMPLETE goes ON at input completion. If the input is failure, P-ERROR goes ON and the input is not completed. The data input status is shown in STATUS. • Holds EXECUTE until COMPLETE or P-ERROR is ON.		
Function I/F			

	No.	Name	Type	Content	Remarks
Input	1	EXECUTE	BIT	Input command	Holds until completion.
	2	IO-ADDR	W (integer)	Input module specification	Setting method is same as IN command.
	3	IN-WORD	W (integer)	Number of input data items	1 to 8
	4	DATA-BUF	ADRS	Input data area to be stored	
Output	1	COMPLETE	BIT	Processing completed	
	2	P-ERROR	BIT	Function input error	
	3	STATUS	W (integer)	Shows each input status corresponding to bits. (1: normal, 0:error) Bit 0 : Input status of 1st word Bit 1 : Input statas of 2nd word ⋮ Bit 7 : Input status of 8th word	

3.20 DIRECT OUTPUT FUNCTION

Function name	NINP	Use restriction	
Function	This function executes data output of prural words collectively to a local output module, in user program independently of system I/O. • When EXECUTE is ON, the data (specified by OUT-WORD) are output continuously from local output module (specified by IO-ADDR). • Output data are stored to the specified area OUT-DATA beforehand. • COMPLETE goes ON at output completion. If the output is failure, P-ERROR goes ON and the output is not completed. The data output status is shown is STATUS. • Holds EXECUTE until COMPLETE or P-ERROR is ON.		
Function I/F			

	No.	Name	Type	Content	Remarks
Input	1	EXECUTE	BIT	Output command	Holds until completion.
	2	IO-ADDR	W (integer)	Output module specification	Setting method is same as OUT command.
	3	OUT-WORD	W (integer)	Number of output data items	1 to 8
	4	DATA_BUF	ADRS	Output data area to be stored	
Output	1	COMPLETE	BIT	Processing completed	
	2	P-ERROR	BIT	Function output error	
	3	STATUS	W (integer)	Shows each output status corresponding to bits. (1: normal, 0:error) Bit 0 : Output status of 1st word Bit 1 : Output status of 2nd word ⋮ Bit 7 : Output status of 8th word	

CONTROL PACK CP-3300
DESIGNER'S MANUAL

TOKYO OFFICE
New Pier Takeshiba South Tower, 1-16-1, Kaigan, Minatoku, Tokyo 105-6891 Japan
Phone 81-3-5402-4511 Fax 81-3-5402-4580

YASKAWA ELECTRIC AMERICA, INC.
2121 Norman Drive South, Waukegan, IL 60085, U.S.A.
Phone 1-847-887-7000 Fax 1-847-887-7370

MOTOMAN INC. HEADQUARTERS
805 Liberty Lane West Carrollton, OH 45449, U.S.A.
Phone 1-937-847-6200 Fax 1-937-847-6277

YASKAWA ELÉTRICO DO BRASIL COMÉRCIO LTDA.
Avenida Fagundes Filho, 620 Bairro Saude-Sao Pãulo-SP, Brazil CEP: 04304-000
Phone 55-11-5071-2552 Fax 55-11-5581-8795

YASKAWA ELECTRIC EUROPE GmbH
Am Kronberger Hang 2, 65824 Schwalbach, Germany
Phone 49-6196-569-300 Fax 49-6196-888-301

Motoman Robotics Europe AB
Box 504 S38525 Torsås, Sweden
Phone 46-486-48800 Fax 46-486-41410

Motoman Robotec GmbH
Kammerfeldstraβe 1, 85391 Allershausen, Germany
Phone 49-8166-900 Fax 49-8166-9039

YASKAWA ELECTRIC UK LTD.
1 Hunt Hill Orchardton Woods Cumbernauld, G68 9LF, United Kingdom
Phone 44-1236-735000 Fax 44-1236-458182

YASKAWA ELECTRIC KOREA CORPORATION
Kfpa Bldg #1201, 35-4 Youido-dong, Yeongdungpo-Ku, Seoul 150-010, Korea
Phone 82-2-784-7844 Fax 82-2-784-8495

YASKAWA ELECTRIC (SINGAPORE) PTE. LTD.
151 Lorong Chuan, #04-01, New Tech Park Singapore 556741, Singapore
Phone 65-282-3003 Fax 65-289-3003

YASKAWA ELECTRIC (SHANGHAI) CO., LTD.
4F No.18 Aona Road, Waigaoqiao Free Trade Zone, Pudong New Area, Shanghai 200131, China
Phone 86-21-5866-3470 Fax 86-21-5866-3869

YATEC ENGINEERING CORPORATION
Shen Hsiang Tang Sung Chiang Building 10F 146 Sung Chiang Road, Taipei, Taiwan
Phone 886-2-2563-0010 Fax 886-2-2567-4677

YASKAWA ELECTRIC (HK) COMPANY LIMITED
Rm. 2909-10, Hong Kong Plaza, 186-191 Connaught Road West, Hong Kong
Phone 852-2803-2385 Fax 852-2547-5773

BEIJING OFFICE
Room No. 301 Office Building of Beijing International Club, 21
Jianguomenwai Avenue, Beijing 100020, China
Phone 86-10-6532-1850 Fax 86-10-6532-1851

TAIPEI OFFICE
Shen Hsiang Tang Sung Chiang Building 10F 146 Sung Chiang Road, Taipei, Taiwan
Phone 886-2-2563-0010 Fax 886-2-2567-4677

SHANGHAI YASKAWA-TONGJI M & E CO., LTD.
27 Hui He Road Shanghai China 200437
Phone 86-21-6531-4242 Fax 86-21-6553-6060

BEIJING YASKAWA BEIKE AUTOMATION ENGINEERING CO., LTD.
30 Xue Yuan Road, Haidian, Beijing P.R. China Post Code: 100083
Phone 86-10-6233-2782 Fax 86-10-6232-1536

SHOUGANG MOTOMAN ROBOT CO., LTD.
7, Yongchang-North Street, Beijing Economic Technological Investment & Development Area,
Beijing 100076, P.R. China
Phone 86-10-6788-0551 Fax 86-10-6788-2878

YASKAWA

YASKAWA ELECTRIC CORPORATION

Specifications are subject to change without notice
for ongoing product modifications and improvements.

MANUAL NO. SIE-C873-20.2D

689-169,688-421