

mitsubishi

QnA SERIES

**MELSEC QnA Series
General Instruction Manual**



Mitsubishi Programmable Controller

REVISIONS

*The manual number is given on the bottom left of the back cover.

Print Date	*Manual Number	Revision
July, 1996	IB (NA) 66605-A	First edition

INTRODUCTION

Thank you for choosing the Mitsubishi MELSEC-QnA Series of General Purpose Programmable Controllers. Please read this manual carefully so that the equipment is used to its optimum. A copy of this manual should be forwarded to the end User.

CONTENTS

1. GENERAL DESCRIPTION	1-1 ~ 1-15
1.1 Large Capacity	1-2
1.2 High Speed Processing	1-4
1.3 Numerous Convenient Functions	1-7
1.4 Improved Program Productivity	1-11
2. SYSTEM CONFIGURATION	2-1 ~ 2-15
2.1 System Configuration	2-1
2.1.1 Equipment configuration in an independent system	2-1
2.1.2 Configuration of peripheral devices for QnACPU	2-2
2.2 System Equipment List	2-3
2.3 Cautions about the System Configuration	2-12
2.4 Selecting Memory Card Models	2-14
3. QnACPU	3-1 ~ 3-45
3.1 General Specifications	3-1
3.2 Performance Specifications	3-2
3.2.1 Performance specifications	3-2
3.2.2 Devices	3-4
3.3 Instruction Lists	3-9
3.3.1 Sequence instructions	3-9
3.3.2 Basic instructions	3-11
3.3.3 Application instructions	3-18
3.3.4 Data link instructions	3-31
3.3.5 PID control instructions	3-32
3.3.6 Special function module instructions	3-33
3.4 Functions	3-36
3.4.1 Program structuring	3-37
3.4.2 File management	3-39
3.4.3 Diagnosis function	3-40
3.4.4 Unrestricted I/O allocation in module units	3-42
3.4.5 Boot operation	3-43
3.4.6 Data protection function	3-44
3.4.7 Device initial values	3-45
4. GPP FUNCTION SOFTWARE PACKAGE (SW0IVD-GPPQ)	4-1 ~ 4-31
4.1 Function List	4-1
4.2 System Configuration	4-3
4.3 Functions	4-6
4.3.1 Enhanced editing functions	4-6
4.3.2 Program standardization	4-8
4.3.3 Debugging function	4-11

4.3.4	Document creation function	4 - 20
4.3.5	Utilizing a MELSEC-A series program	4 - 21
4.4	Powerful Support Tools	4 - 22
4.4.1	SW0IVD-MSPQ/MSDQ-type macro/library software package	4 - 22
4.4.2	SW0IVD-CNVQ data conversion software package	4 - 26
4.4.3	SW0IVD-CADQ-type CAD conversion software package	4 - 28
4.4.4	SW0IVD-LNKQ-type ladder sequence program linking software package	4 - 29
5.	PROGRAMMING UNIT (Q6PU)	5 - 1 ~ 5 - 4
5.1	Performance Specifications	5 - 1
5.2	Function List	5 - 2
6.	SFC	6 - 1 ~ 6 - 13
6.1	Performance Specifications Related to SFC Programs	6 - 1
6.2	SFC Features	6 - 3
6.3	List of SFC Diagram Symbols	6 - 12
7.	MELSECNET/10 NETWORK SYSTEM	7 - 1 ~ 7 - 16
7.1	Performance Specifications	7 - 1
7.1.1	Performance specifications for a PC-to-PC network	7 - 1
7.1.2	Performance Specifications for a Remote I/O Network	7 - 2
7.2	Functions	7 - 3
7.2.1	Direct access	7 - 3
7.2.2	Improved transient transmission	7 - 4
7.2.3	Transient transmission possible even during PC CPU error	7 - 6
7.2.4	Improved convenience of default parameters	7 - 7
7.2.5	Network duplication (PC-to-PC network)	7 - 8
7.2.6	Multiple master stations (remote I/O network)	7 - 9
7.2.7	Parallel master stations (remote I/O network)	7 - 10
7.2.8	Increasing the number of transmission points by installing multiple modules with the same network No. (PC-to-PC network)	7 - 11
7.2.9	Preventing loopback by the external power supply (PC-to-PC network: Optical loop system)	7 - 12
7.3	Compatibility with MELSECNET/10 for AnU	7 - 13
7.3.1	PC-to-PC network	7 - 13
7.3.2	Remote I/O network	7 - 15
8.	SERIAL COMMUNICATION MODULE (AJ71QC24)	8 - 1 ~ 8 - 16
8.1	Transmission Specifications	8 - 1
8.2	Features	8 - 3
8.2.1	Dedicated protocol function	8 - 3
8.2.2	No-protocol protocol function	8 - 10
8.2.3	Bidirectional protocol function	8 - 11
8.2.4	Choose the model that suits your intended application	8 - 12
8.2.5	Other useful functions	8 - 14
8.2.6	Incorporation into your existing system	8 - 16

1. GENERAL DESCRIPTION

Programmable controllers have become increasingly important as the central components of factory automation systems. Together with CIM developments, and user systems increasing in both size and complexity, the range of applications has steadily expanded and control programs are rapidly becoming larger. In recent years, the cost of hardware in control systems using programmable controllers has fallen, while software costs have soared.

This situation, combined with a lack of software technicians, gives rise to the issue of how to proceed with program development, debugging and maintenance. There are increasing demands for software tools with complete man-machine interfaces and foundation programmable controllers to efficiently develop and debug increasingly larger programs.

The QnA series was developed to provide programmable controllers capable of meeting these needs, with the following items as development concepts.

(1) Concepts for software development environment

- (a) Programs are created in a structured way and can be divided into their different functions, enabling development to be shared among a number of people.
- (b) Standard ladder programs can be developed, and can be applied freely and easily.
- (c) Flexible editing and fast operations are possible.
- (d) The program is easily understood by anyone, and includes a comprehensive software package for all stages of design, debugging, adjustment and maintenance. You will realize a rapid improvement in your program development environment.

(2) Concepts for programmable controllers which provide a foundation

- (a) The PC is a basic engine with considerably improved basic performance and system performance.
- (b) There is a large program and device capacity, enabling you to conduct program design easily.
- (c) There are numerous useful features to enable you to apply the PC in a wide range of areas.

1.1 Large Capacity

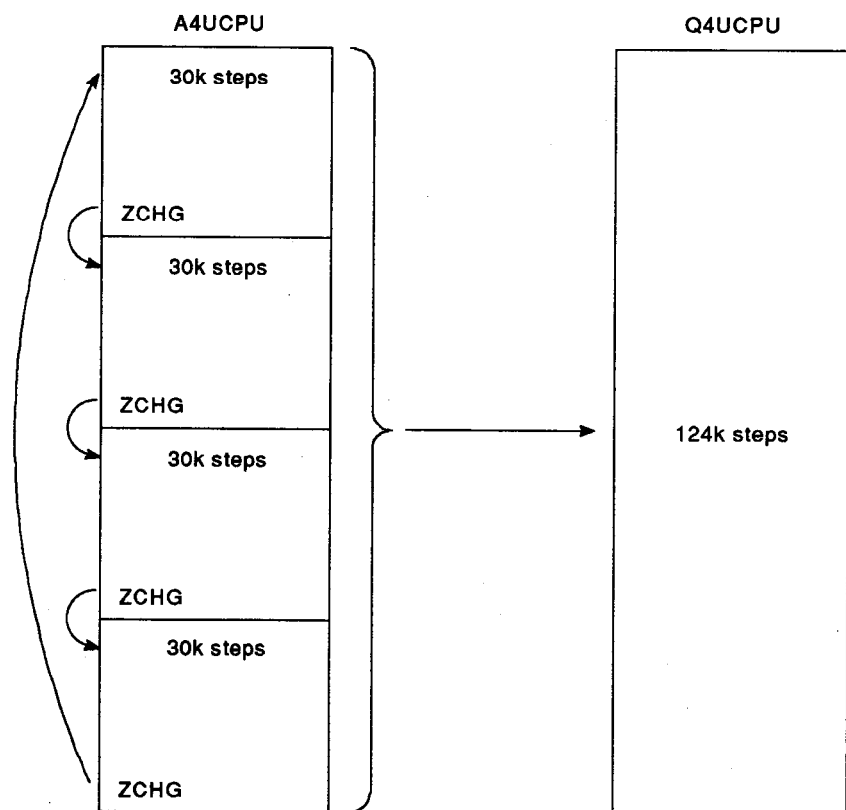
(1) Program capacity of 124k steps

The respective program capacities are: Q2ACPU: 28k steps, Q2ACPU-S1: 60k steps, Q3ACPU: 92k steps, and Q4ACPU: 124k steps of program capacity, with that memory incorporated into QnACPU.

124k step programs can be executed as one continuous program.

Moreover, by reducing the number of steps in the instructions, about 1.7 times more control content can be realized in comparison to MELSEC-A.

When executing a program exceeding 30k steps with MELSEC-A, special programs were needed, such as switching between main programs and sub-programs by CHG(ZCHG) instructions. However, using QnACPU these sorts of programs are no longer required.



(2) Device memory of about 29k words

The large device memory totals 29k words, comprising internal relays (M): 8192 points, latch relays (L): 8192 points, data registers: 12288 points, and normal timers: 2048 points.

You can change the memory allocated to each device type to suit your applications.

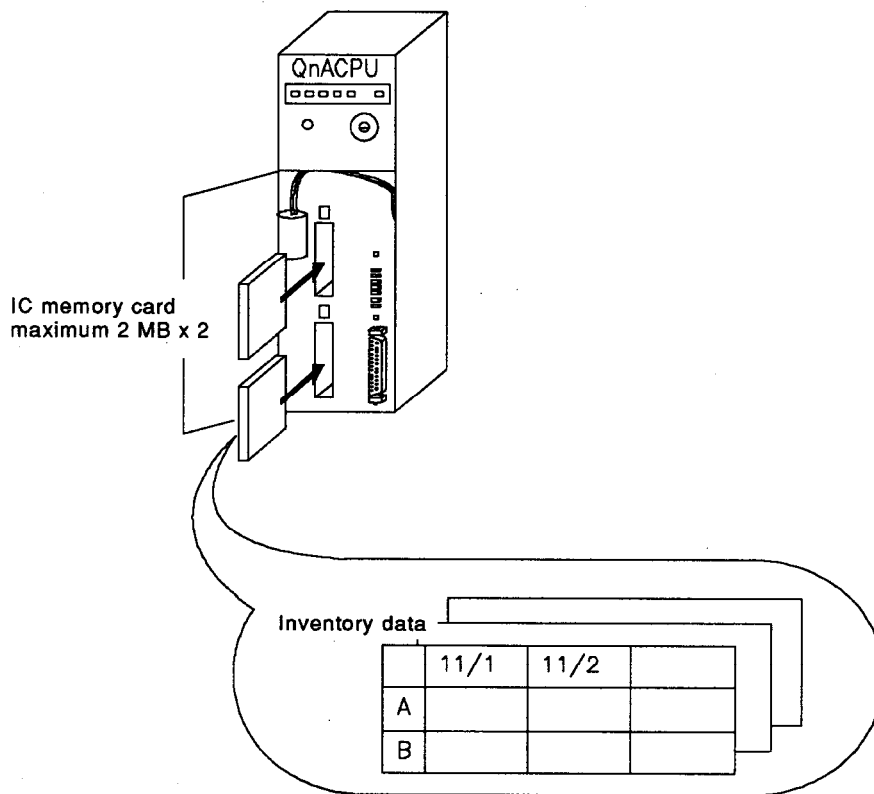
For example, if your control requires a large number of bit devices, the internal relay (M) area can be expanded from its default value of 8k to 32k.

(3) Memory card maximum of 2 MB x 2

Memory cards are used for storing programs, comments, statements and file registers. (However as programs can be stored in the CPU module itself, QnACPU can run even without memory cards).

You can transfer programs from the memory card to QnACPU at high speed.

A high-speed memory card conforming to JEIDA Ver. 4.1^{*1} is provided for this purpose.



*1 JEIDA: Japan Electronic Publishers Association

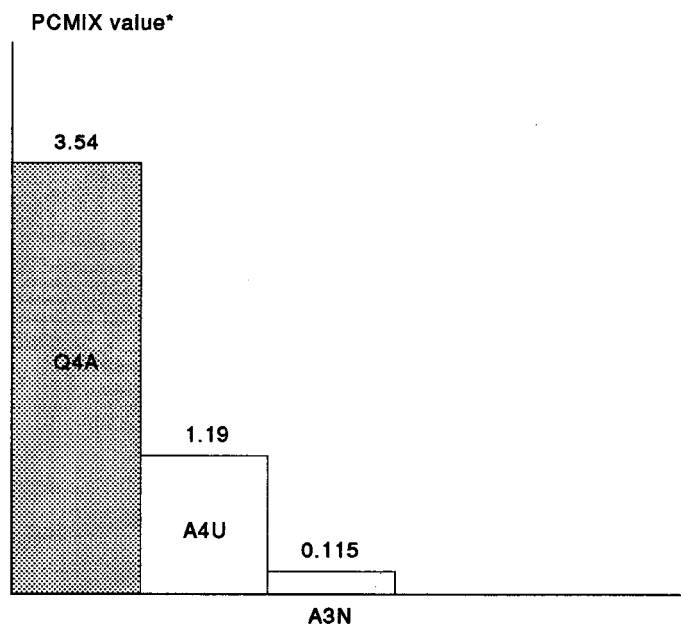
1.2 High Speed Processing

(1) Unsurpassed operation processing speed

Both basic instructions and application instructions have been sped up.

	A4UCPU		Q4ACPU
Basic instructions	0.15 μ s	→	0.075 μ s
Transfer instructions	0.90 μ s	→	0.225 μ s
PCMIX	1.20	→	3.54

The PCMIX value is the average number of instructions that can be executed in 1 μ s.



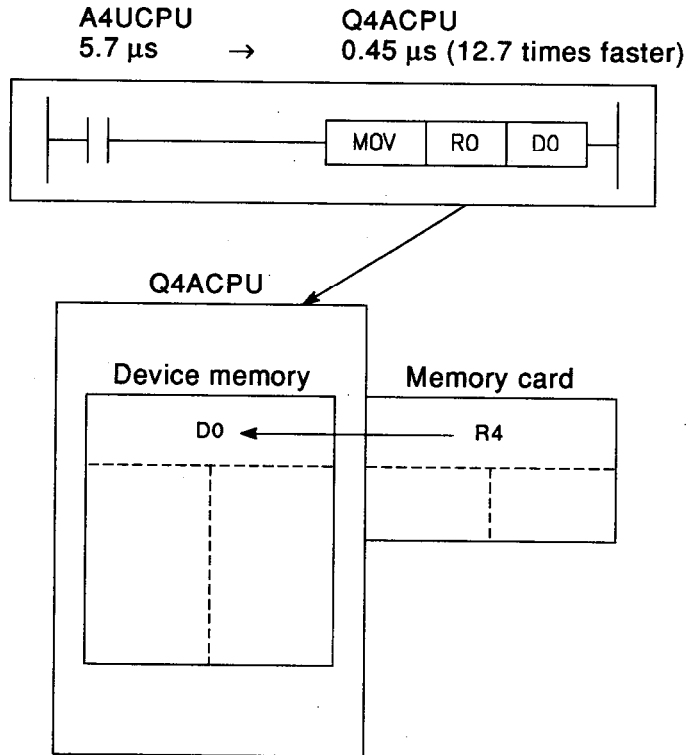
*PCMIX: Average number of instructions that can be executed in 1 μ s.

1. GENERAL DESCRIPTION

MELSEC-QnA

- (2) High speed processing of expanded data memory (File registers: R)

The processing time for file registers is the same as for QnACPU internal devices (data registers: D, link registers: W).



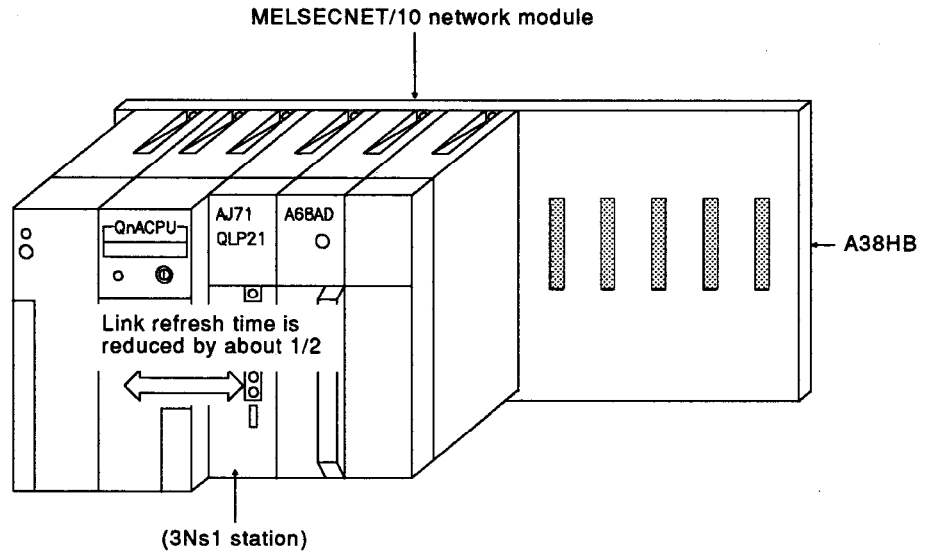
- (3) High speed accessing of buffer memories

The speed of reading/writing to QnA special function module (serial communication module) buffer memories has been increased several times over the AnUCPU.

(Note that the processing speed for existing special function modules is the same as for the AnUCPU.)

(4) The high speed access base unit (A38HB)

Data transfer between QnACPU and special function modules is conducted about twice as fast as when using the A38B. The QnACPU processing time for accessing special function modules can be reduced simply by mounting network modules or serial communication modules which handle large amounts of data on the high speed access base unit.



1.3 Numerous Convenient Functions

(1) Program execution type is selectable to suit control type

Programs can be selected from four types: initial execution, scan execution, low-speed execution, and standby.

These programs can be set to run when required, thereby reducing the scan time of the permanently executing scan program.

(a) Initial execution (initial program)

A program executed once only when the QnACPU is set to RUN. Used for initialization.

(b) Scan execution (scan program)

A permanently executing program. Equivalent to a conventional program running from step 0 to END.

Can include sub-routine programs and interrupt programs.

(c) Low-speed execution (low-speed program)

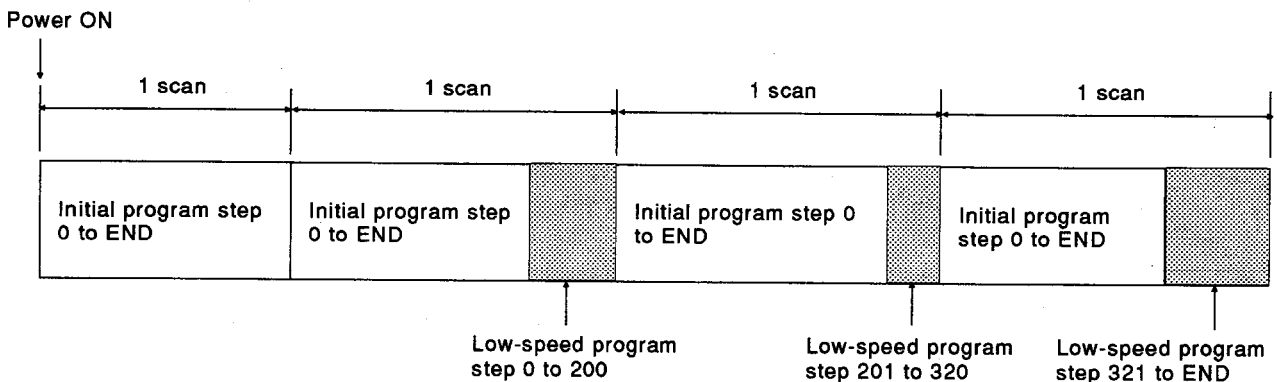
A program which is not entirely executed each scan but which is executed over several scans, using the constant scan surplus time or a set time.

Used for programs run at low frequency, such as scheduled inspection programs.

(d) Standby (standby program)

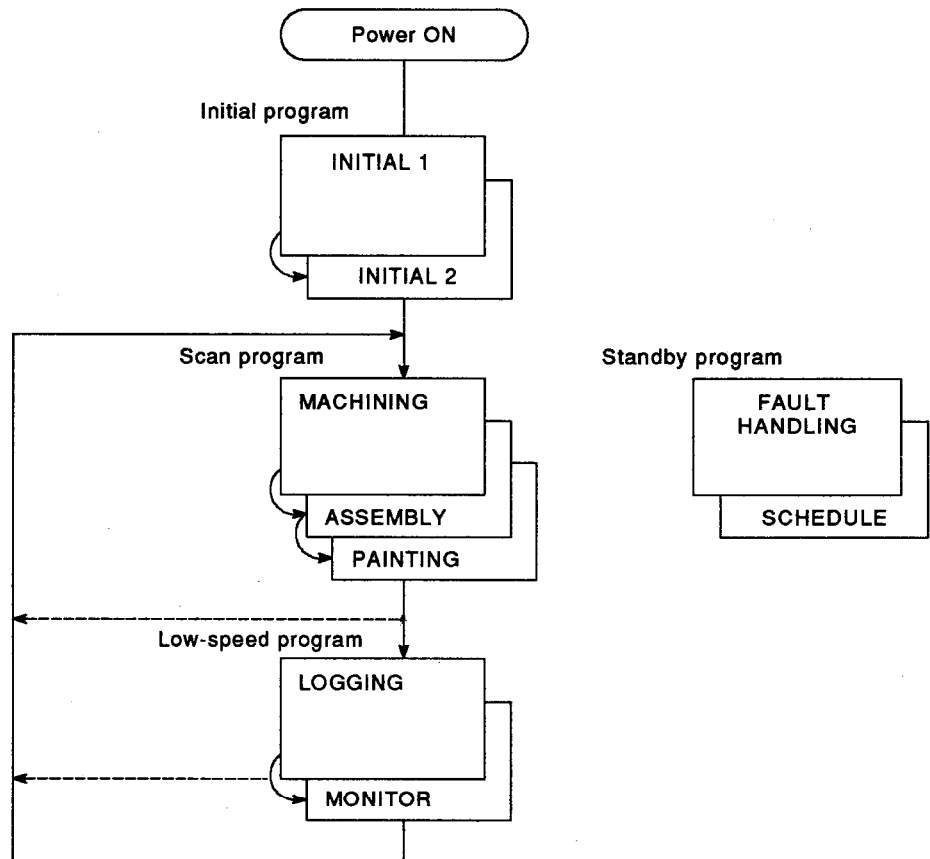
A program which is executed when a certain start condition is met, such as a sub-routine program or an interrupt program.

Used when the sub-routine programs and interrupt programs are held as library data and are handled separately to the main program.



(2) File management

The sequence program and other data handled by the QnACPU is managed as files. Whereas programs were conventionally handled as main programs and sub-programs, the QnACPU allows structured programming, such that each type of program can be broken down for individual programmers or processes.

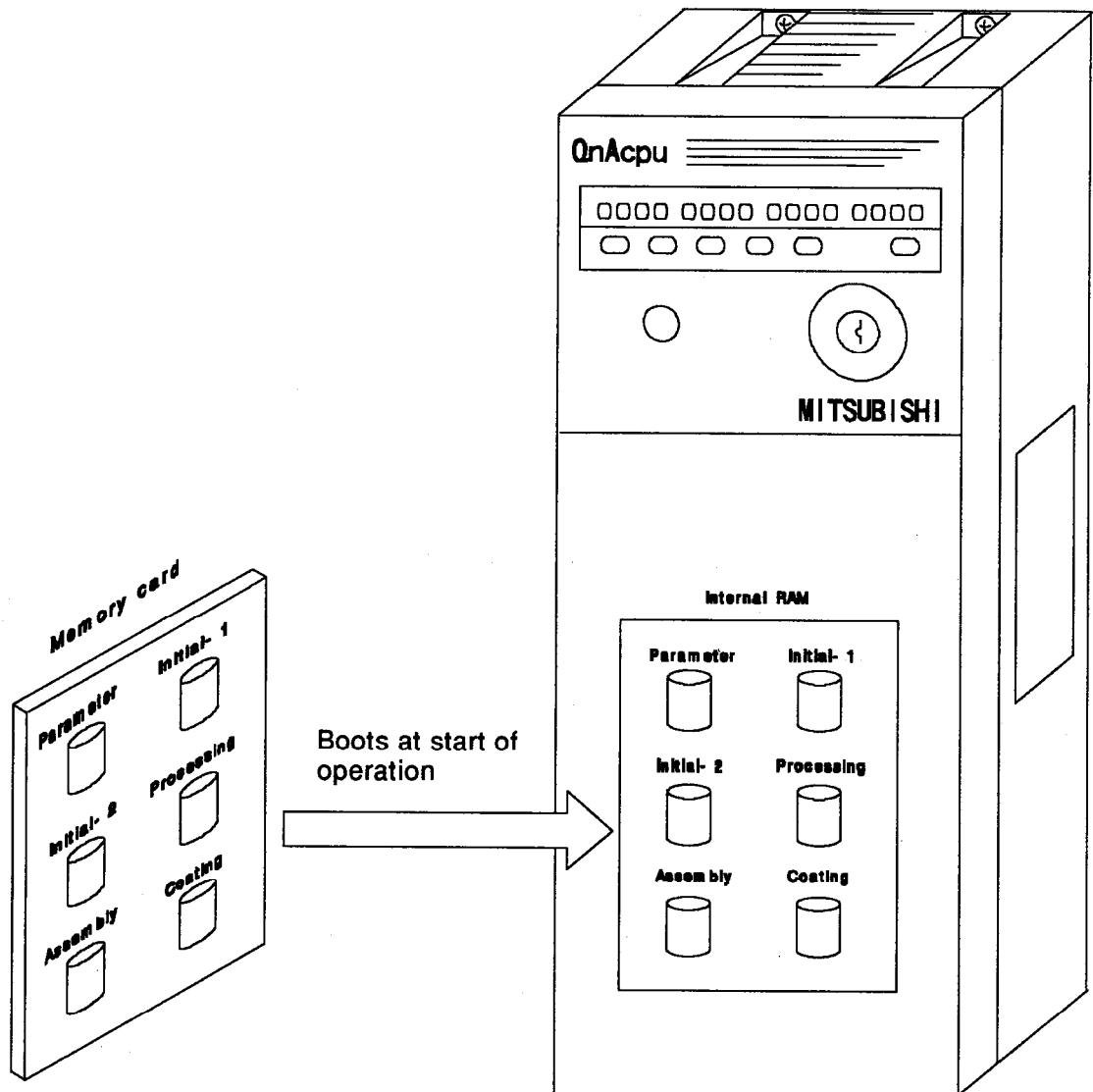


(3) Boot operation

Sequence programs can be stored and executed in QnACPU internal RAM. However, when starting programmable controller operation the sequence program can also be executed after reading from the memory card. This process is called boot operation.

Using the boot operation function the following operations are possible:

- (a) If separate programs are stored in the QnACPU module and memory card, you can easily switch to the program complying with the controls.
- (b) By booting from the memory card to the QnACPU, the program can be written quickly.
Even if there is no peripheral device, high speed program writing can be done simply by booting from the memory card.
- (c) During ROM operation, a program can be read from a ROM memory card and executed.

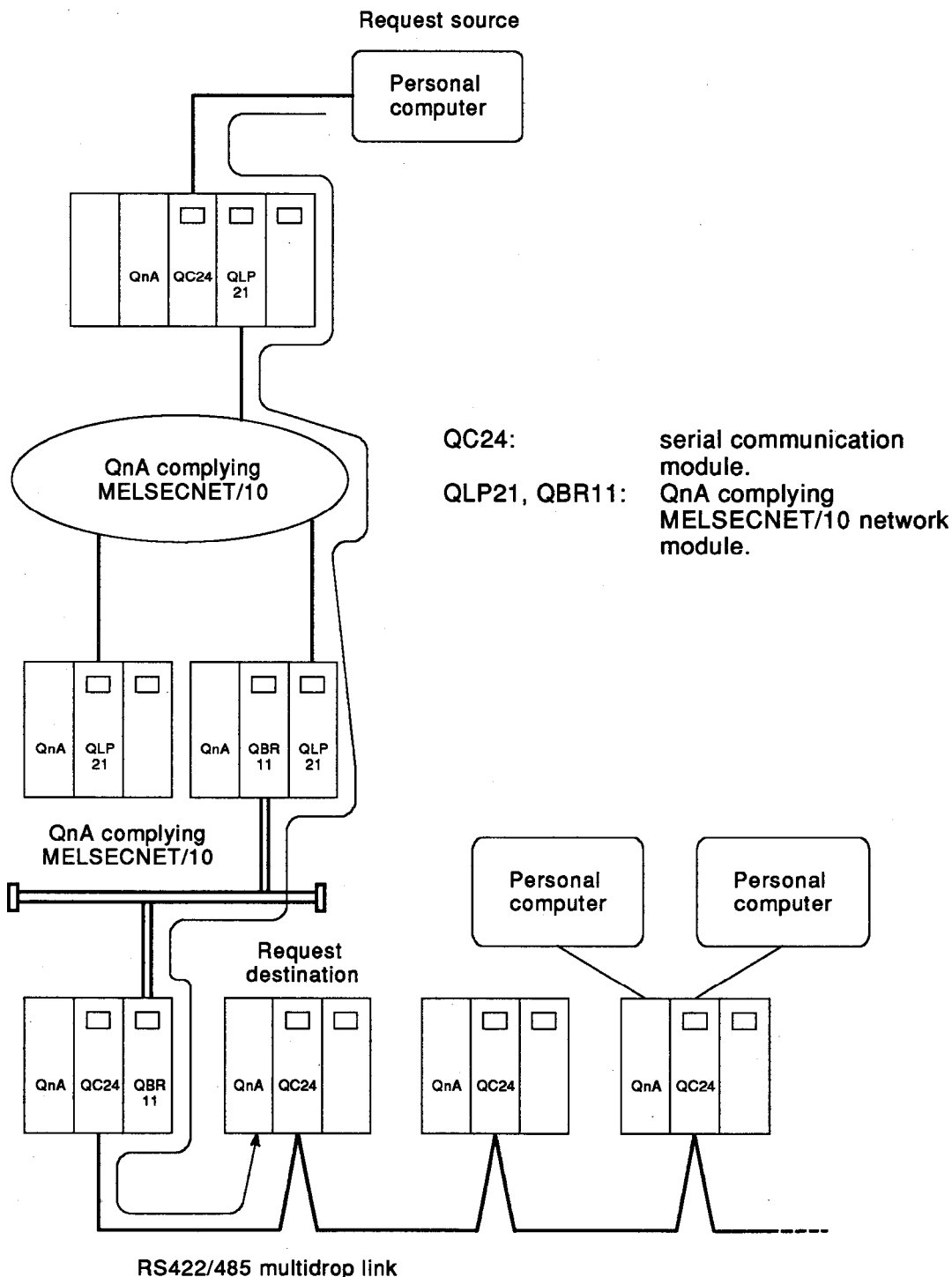


1. GENERAL DESCRIPTION

MELSEC-QnA

(4) Support for total network

Using the QnACPU, you can freely access other stations in a MELSEC-NET/10 network system from sequence programs, serial communication modules, peripheral devices, etc.
 Furthermore, you can similarly access other stations connected in a serial communication multidrop link system.



1.4 Improved Program Productivity

(1) Planning structured programs

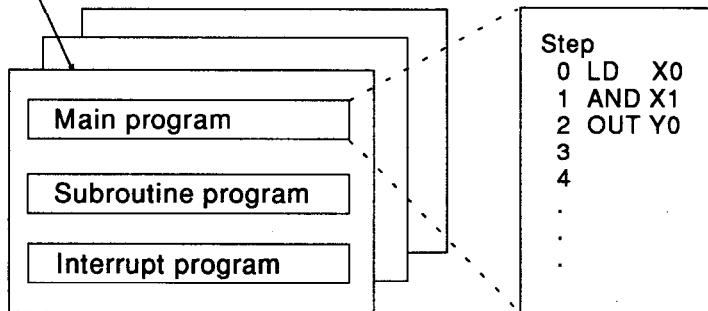
Programs that conventionally would have been a single continuous unit can now be managed in a structured manner as multiple files.

- (a) It is possible to share design tasks among multiple designers, and conduct file management separately for separate programs, divided into areas of function, production, and designers.
- (b) You can manage the file name and the file amendment time simultaneously, allowing you to see the program content and amendment time at a glance. As a result the maintenance efficiency is improved.
- (c) The reading/writing of programs to the QnACPU is conducted in file units, reducing the time required for reading/writing.
- (d) Programs which were developed by different designers can be coupled, and the device numbers can be changed easily.

File	Type	Size	Date	Time	Title
PARAM	Parameter	330	96-05-20	15:13	:Parameters for packing line
INITIAL1	QnA Seq	107	96-05-20	15:32	:Machine module data setting
INITIAL2	QnA Seq	107	96-05-20	14:41	:Production plant data setting
MACHINE	QnA Seq	107	96-05-20	14:42	:Machining process control program
ASSEMBLY	QnA Seq	107	96-05-20	14:42	:Assembly process control program
COATING	QnA Seq	107	96-05-20	14:43	:Coating process control program
MONITOR	QnA Seq	107	96-05-20	14:44	:Line monitor program
LOGGING	QnA Seq	107	96-05-20	14:44	:Logging program

File(s): 8 Free 153411584Byte(s) Execute(Y) Cancel(N)

EqUp:Prev PgDn:Next Ctrl+D:Dir Space>Select Esc:Close

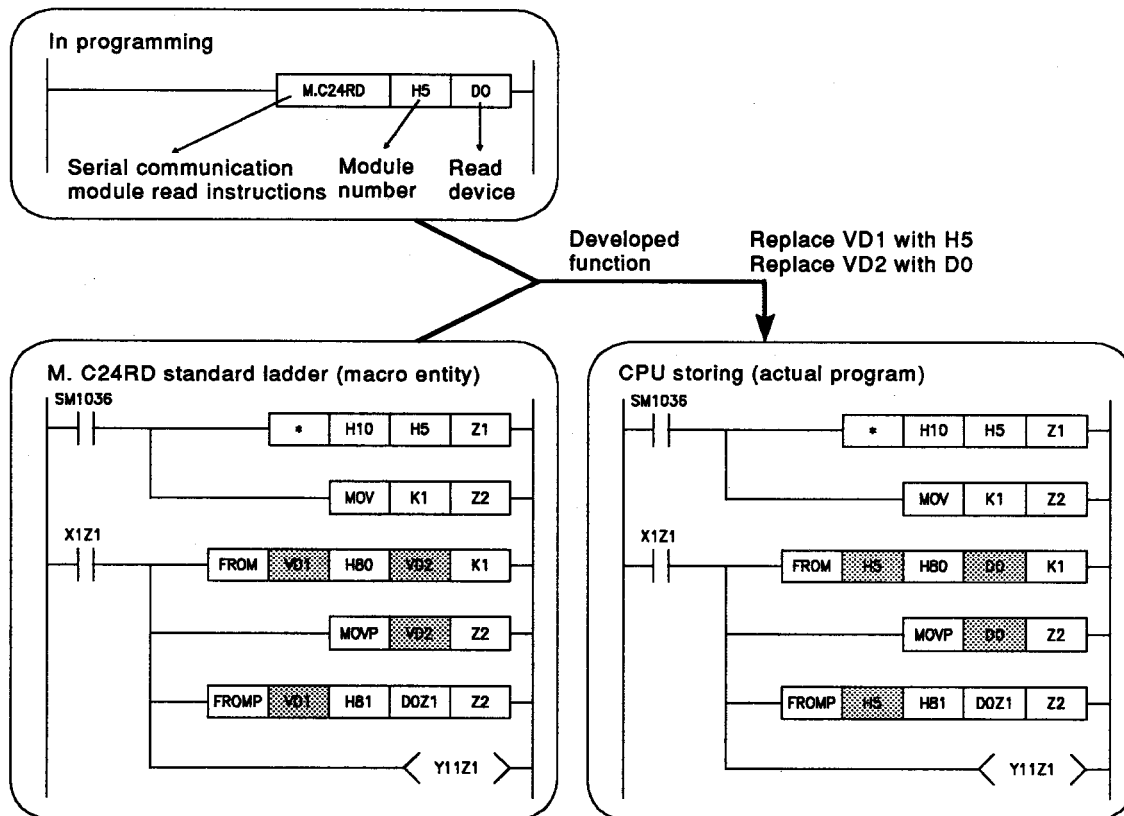


By function/production/designer

1. GENERAL DESCRIPTION

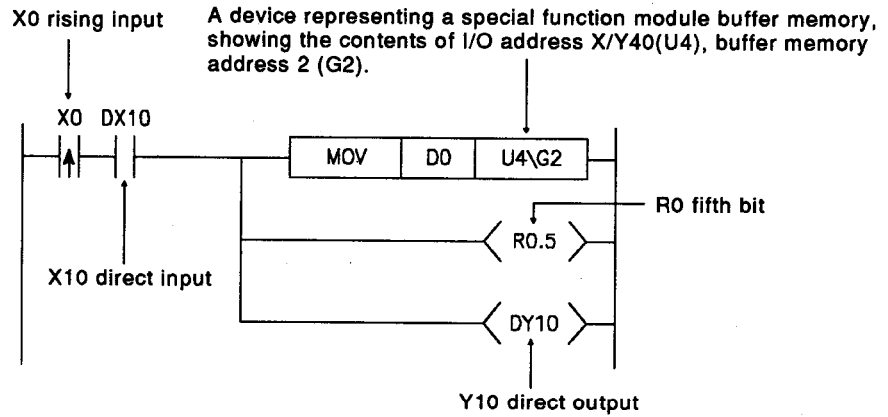
(2) Instructions can be defined in terms of macro instructions

It is possible to freely create and use instructions in accordance with functions, enabling programs to be standardized and simplified.



(3) Completely unrestricted device handling

- (a) You can conduct word device bit operations.
- (b) Differential contacts and direct designations are possible.
- (c) You can program special function module buffer memories as devices.
- (d) You can directly access network module link data if they are programmed as devices.



An Example of Programming

Function	AnUCPU Transcription	QnACPU Transcription
Handling special module buffer memory device		
Simplification of previous dedicated instructions		
Free device expression		

1. GENERAL DESCRIPTION

(4) Editing operability has been improved

(a) Up to 4 programs or data can be edited at the same time.

- It is possible to cut-and-paste between the objects being edited. You can do this easily while confirming on the screen the utilization of programs and data.
- You can switch between the objects being edited at the touch of a button.

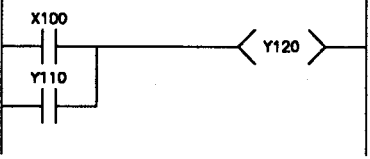
(b) The number of key entries needed to create ladders has been reduced, for example when connecting vertical and horizontal lines by dotted lines, or when inserting parallel coils.

(c) Comments can be displayed while you edit the ladder.

(d) Operations are user-friendly, with pull-down menus and dialog boxes.

You can easily operate the functions which you use frequently by using the function keys.

Editing object 1

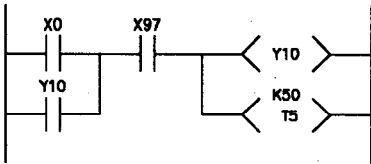


Editing object 2

```

Pointer
Level 1 2 3 4 5 6
P 0 [Assembly process 1 Input subroutine]
P 1 [Assembly process 1 Actual operation subroutine]
P 2 [Assembly process 1 Output subroutine]
P 3 [Assembly process 2 Input subroutine]
P 4 [Assembly process 2 Actual operation subroutine]
P 5 [Assembly process 2 Output subroutine]
P 6
P 7
P 8
P 9
P 10
    
```

Editing object 3



Editing object 4

Station	M Sta-M/R Sub	M Sta-C-M/R Sub	M Sta-M/R Sub	M Sta-C-M/R Sub
	First Last	First Last	First Last	First Last
1	[01]-[01]	[100]-[100]	[0]-[1P]	[100]-[11P]
2	[10]-[1P]	[110]-[11P]	[20]-[3P]	[120]-[13P]
3	[20]-[2P]	[120]-[12P]	[40]-[5P]	[140]-[15P]
4	[1000]-[100P]	[1100]-[110P]	[1000]-[100P]	[1100]-[110P]
	[]-[]	[]-[]	[]-[]	[]-[]
	[]-[]	[]-[]	[]-[]	[]-[]
	[]-[]	[]-[]	[]-[]	[]-[]
	[]-[]	[]-[]	[]-[]	[]-[]

- (5) Monitoring and debugging functions have been enhanced
 - (a) The ladder can be edited while monitoring.
 - (b) You can reference the contact coil at the touch of a button.
 - (c) You can reference ON/OFF causes.
 - (d) You can set conditions which are suitable for debugging the monitoring timing, because step numbers and device status can be designated.
 - (e) You can monitor index qualified devices.
- (6) Strong support is provided for document creation
 - (a) The comment size has been expanded to 32 characters, allowing you to attach more detailed comments.
 - (b) You can now set comments for all the devices.
 - (c) Statements and notes attached to programs can be managed together with the program, simplifying program amendment and utilization operations.
 - (d) Printout time is reduced by storing printout data to files.
 - (e) The number of pages of printing can be reduced by justifying blank lines.
- (7) Powerful support tools
 - (a) Data conversion package

You can create comment data and device data using the spreadsheet software (EXCEL^{*1}, Lotus 1-2-3^{*2}, Multiplan^{*3}) and word processing software that you are familiar with, and then convert this data into files that can be used in QnACPU.
Furthermore, you can also convert QnA files to text data or spreadsheet software data.

^{*1} EXCEL is a registered trademark of US Microsoft Co.
^{*2} Lotus 1-2-3 is a registered trademark of Lotus Co.
^{*3} Multiplan is a registered trademark of US Microsoft Co.
 - (b) Macros, libraries

There are basic sequence programs accessing special function modules, and standard programs such as fault finding and alarm processing programs which are provided as macros and libraries, and these support program standardization.
 - (c) Ladder sequence linking package

This package links a number of sequence programs to make just one sequence program.
It includes a function that automatically allocates the devices of each program without duplication, and which supports the shared design and modularization of programs.
 - (d) CAD interface package

This package for communication with CAD systems treats ladders, instruction lists, comment data and SFC diagrams as CAD data.

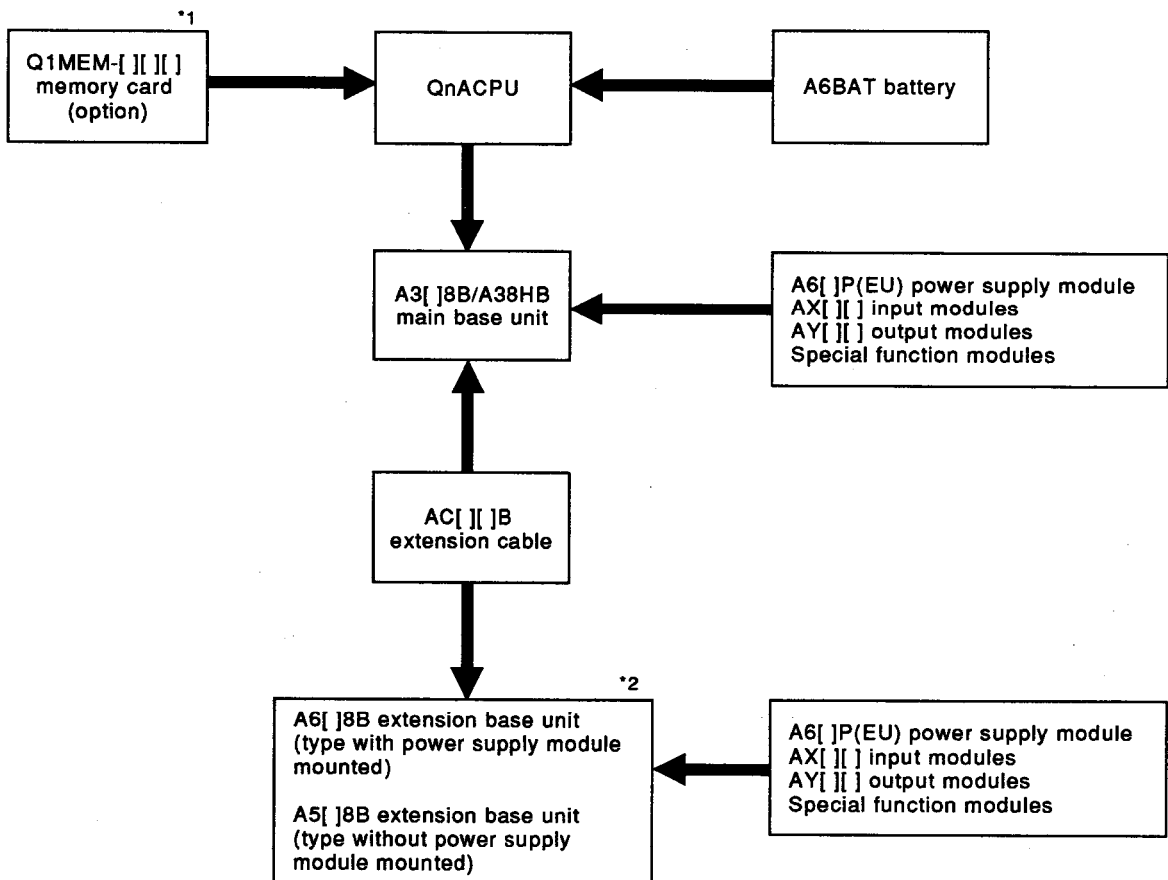
2. SYSTEM CONFIGURATION

This section describes the system configurations that can be used for a system centered on a QnACPU, cautions on configuring the system, and the system equipment.

2.1 System Configuration

The equipment configuration and peripheral device configuration when a QnACPU is used in an independent system is described here.

2.1.1 Equipment configuration in an independent system



POINTS

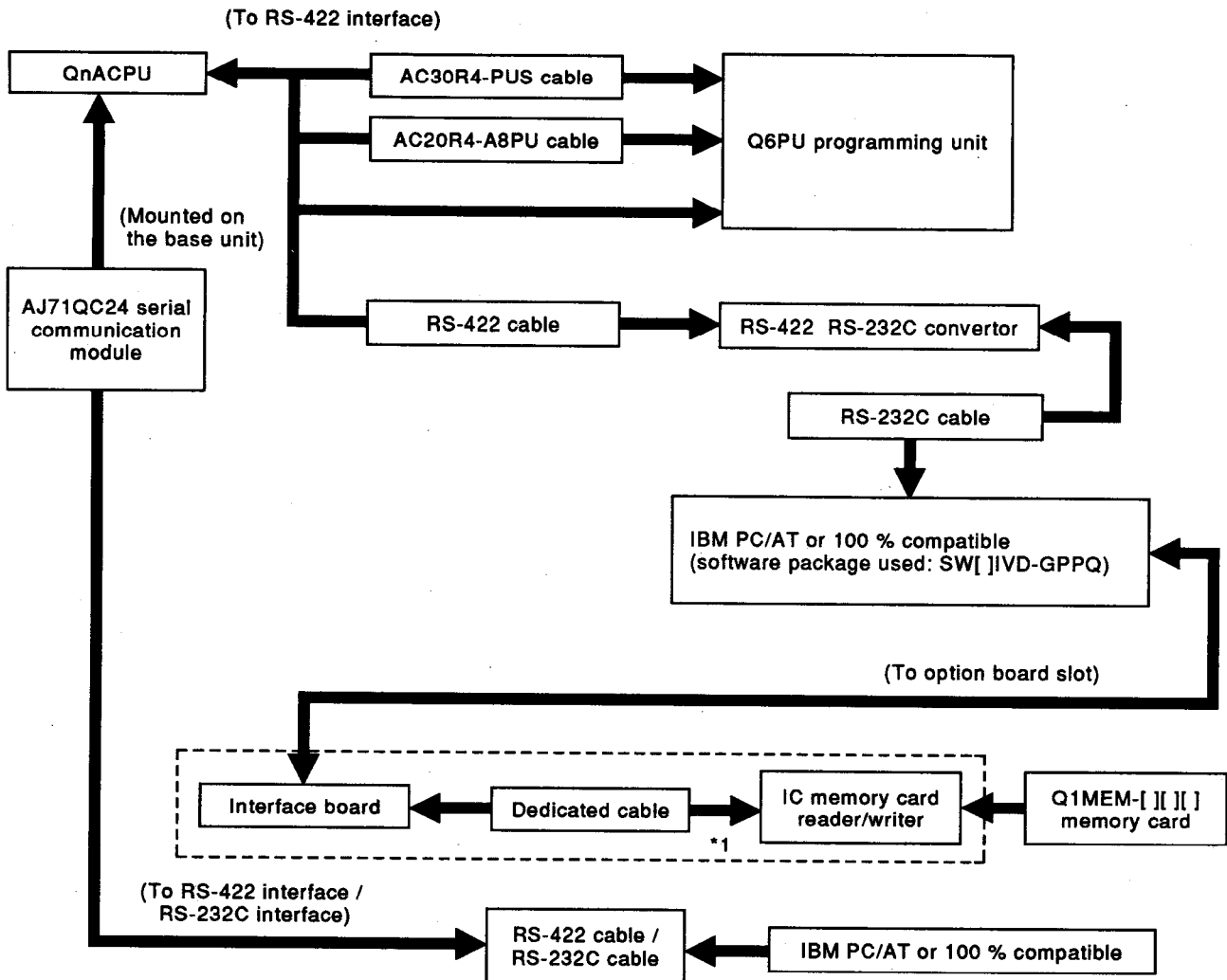
*1 Up to two memory cards can be installed, if required.
SRAM and E²PROM memory cards can read/write files when installed at the CPU.
Flash memory memory cards can only read (not write) files when installed at the CPU.
File writing must be performed from a peripheral device.

*2 When using an A5 J B extension base unit, pay particular attention to the power supply capacity of the main base unit.
In the case of I/O modules with a high internal current consumption, you are recommended to mount the special function module on an A6 J B extension base unit.

2. SYSTEM CONFIGURATION

MELSEC-QnA

2.1.2 Configuration of peripheral devices for QnACPU



*1 For details on the IC memory card reader/writer set, refer to Section 4.2.

2. SYSTEM CONFIGURATION

MELSEC-QnA

2.2 System Equipment List

The system equipment (modules and peripheral devices) that can be used in a QnACPU system are listed here. Descriptions for modules and software packages with some restrictions are given in Section 2.3.

(1) Modules for use with QnACPU

Module	Model	Description	Number of Occupied I/O Points (Module Type in I/O Allocation)	Current Consumption		Remark
				5 VDC (A)	24 VDC (A)	
CPU module	Q2ACPU	Number of I/O points: 512, internal RAM: 28k steps	—	0.3	—	Memory card procured separately. Includes memory card current consumption.
	Q2ACPU-S1	Number of I/O points: 1024, internal RAM: 60k steps		0.3	—	
	Q3ACPU	Number of I/O points: 2048, internal RAM: 92k steps		0.3	—	
	Q4ACPU	Number of I/O points: 4096, internal RAM: 124k steps		0.6	—	
Memory card	Q1MEM-64S	SRAM, 64k bytes	—	—	—	
	Q1MEM-128S	SRAM, 128k bytes				
	Q1MEM-256S	SRAM, 256k bytes				
	Q1MEM-512S	SRAM, 512k bytes				
	Q1MEM-1MS	SRAM, 1 MB				
	Q1MEM-2MS	SRAM, 2 MB				
	Q1MEM-64SE	SRAM, 32k bytes; E ² PROM, 32k bytes				
	Q1MEM-128SE	SRAM, 64k bytes; E ² PROM, 64k bytes				
	Q1MEM-256SE	SRAM, 128k bytes; E ² PROM, 128k bytes				
	Q1MEM-512SE	SRAM, 256k bytes; E ² PROM, 256k bytes				
	Q1MEM-1MSE	SRAM, 512k bytes; E ² PROM, 512k bytes				
	Q1MEM-256SF	SRAM, 128k bytes; flash memory, 128k bytes				
	Q1MEM-512SF	SRAM, 256k bytes; flash memory, 256k bytes				
	Q1MEM-1MSF	SRAM, 512k bytes; flash memory, 512k bytes				
Q1MEM-2MSF	SRAM, 1 MB, flash memory, 1 MB					

2. SYSTEM CONFIGURATION

MELSEC-QnA

Module	Model	Description	Number of Occupied I/O Points (Module Type In I/O Allocation)	Current Consumption		Remark
				5 VDC (A)	24 VDC (A)	
Input modules	AX10	16-input 100 VAC input module	16 (16 inputs)	0.055	—	
	AX11	32-input 100 VAC input module	32 (32 inputs)	0.11	—	
	AX20	16-input 200 VAC input module	16 (16 inputs)	0.055	—	
	AX21	32-input 200 VAC input module	32 (32 inputs)	0.11	—	
	AX31	32-input 12/24 VAC/VDC input module	32 (32 inputs)	0.11	—	
	AX40	16-input 12/24 VDC input module	16 (16 inputs)	0.055	—	
	AX41	32-input 12/24 VDC input module	32 (32 inputs)	0.11	—	
	AX42	64-input 12/24 VDC input module	64 (64 inputs)	0.12	—	
	AX50	16-input 48 VDC sink input module	16 (16 inputs)	0.055	—	
	AX50-S1	16-input 48 VDC sink/source input module	16 (16 inputs)	0.055	—	
	AX60	16-input 100/110/125 VDC sink input module	16 (16 inputs)	0.055	—	
	AX60-S1	16-input 100/110/125 VDC sink/source input module	16 (16 inputs)	0.055	—	
	AX70	16-input input module for sensor	16 (16 inputs)	0.055	—	
	AX71	32-input input module for sensor	32 (32 inputs)	0.11	—	
	AX80	16-input 12/24 VDC source input module	16 (16 inputs)	0.055	—	
	AX80E	16-input 12/24 VDC source input module	16 (16 inputs)	0.055	—	
	AX81	32-input 12/24 VDC source input module	32 (32 inputs)	0.11	—	
	AX81-S2	32-input 48/60 VDC source input module	32 (32 inputs)	0.11	—	
	AX81B	32-input 24 VDC sink/source input module	64 (32 inputs)	0.125	—	
	AX82	64-point 12/24 VDC source input module	64 (64 inputs)	0.12	—	

2. SYSTEM CONFIGURATION

MELSEC-QnA

Module	Model	Description	Number of Occupied I/O Points (Module Type in I/O Allocation)	Current Consumption		Remark	
				5 VDC (A)	24 VDC (A)		
Output modules	AY10	16-output relay contact output module (2 A)	16 (16 outputs)	0.115	0.15	*1: Indicates a source load module. Other modules are sink load modules.	
	AY10A	16-output relay contact output module, for independent contact output	16 (16 outputs)	0.115	0.15		
	AY11	16-output relay contact output module, with surge suppression	16 (16 outputs)	0.115	0.15		
	AY11A	16-output relay contact output module, for independent contact output, with surge suppression	16 (16 outputs)	0.115	0.15		
	AY11E	16-output relay contact output module (fused)	16 (16 outputs)	0.115	0.15		
	AY13	32-point relay contact output module (2 A)	32 (32 outputs)	0.23	0.29		
	AY13E	32-point relay contact output module (fused)	32 (32 outputs)	0.23	0.29		
	AY22	16-point triac output module (2 A, fused)	16 (16 outputs)	0.305	—		
	AY23	32-point triac output module (0.6 A, fused)	32 (32 outputs)	0.59	—		
	AY40	16-output 12/24 VDC transistor output module (0.1 A)	16 (16 outputs)	0.115	0.016		
	AY40A	16-output 12/24 VDC transistor output module, for independent contact output (0.3 A)	16 (16 outputs)	0.19	—		
	AY40P	16-output 12/24 VDC transistor output module, with short protection function and overheat protection function	16 (16 outputs)	0.115	0.03	The short protection and overheat protection functions of the AY40P, AY41P, AY60EP, AY80EP, AY81EP, and AY82EP are described below:	
	AY41	32-output 12/24 VDC transistor output module (0.1 A)	32 (32 outputs)	0.23	0.04		
	AY41P	32-output 12/24 VDC transistor output module, with short protection function and overheat protection function	32 (32 outputs)	0.23	0.06		
	AY42	64-output 12/24 VDC transistor output module (0.1 A)	64 (64 outputs)	0.29	0.08		
	AY42-S3	64-output 12/24 VDC transistor output module (fused)	64 (64 outputs)	0.29	0.08		
	AY42-S4	64-output 12/24 VDC transistor output module, with photocoupler with built-in Zener diode	64 (64 outputs)	0.50	—		
	AY50	16-output 12/24 VDC transistor output module (0.5 A, fused)	16 (16 outputs)	0.115	0.13		Short protection function
	AY51	32-output 12/24 VDC transistor output module (0.5 A)	32 (32 outputs)	0.23	0.10		Function that protects the transistors from overcurrents occurring, for example, due to short circuits in external wiring.
	AY51-S1	32-output 12/24 VDC transistor output module (0.3 A, fused)	32 (32 outputs)	0.31	0.02		
	AY60	16-output 12/24/48 VDC transistor output module (2 A, fused)	16 (16 outputs)	0.115	0.13		
	*1 AY60E	16-output 12/24/48 VDC transistor output module (fused) 12/24 VDC: 2 A, 48 VDC: 0.8 A	16 (16 outputs)	0.115	0.13	Overheat protection function	
	AY60S	16-output 12/24/48 VDC transistor output module (2 A)	16 (16 outputs)	0.075	0.006		
	*1 AY60EP	16-output 12/24 VDC transistor output module (2 A), with short protection function and overheat protection function	16 (16 outputs)	0.115	0.22	Function that protects the transistors from damage due to external temperature rise attributable to external causes.	
	AY70	16-output, TTL/CMOS (5/12 VDC) output module (16 mA)	16 (16 outputs)	0.10	12 VDC 0.11		
	AY71	32-output, TTL/CMOS (5/12 VDC) output module (16 mA)	32 (32 outputs)	0.20	12 VDC 0.20		
	AY72	64-output, TTL/CMOS (5/12 VDC) output module (16 mA)	64 (64 outputs)	0.30	12 VDC 0.60		
	*1 AY80	16-output 12/24/48 VDC transistor output module (0.5 A, fused)	16 (16 outputs)	0.115	0.12		
	*1 AY80EP	16-output 12/24 VDC transistor output module (0.8 A), with short protection function and overheat protection function	16 (16 outputs)	0.115	0.22		
	AY81	32-output 12/24 VDC transistor output module (0.5 A)	32 (32 outputs)	0.23	0.10		
*1 AY81EP	32-output 12/24 VDC transistor output module (0.8 A), with short protection function and overheat protection function	32 (32 outputs)	0.23	0.44			
*1 AY82EP	64-output 12/24 VDC transistor output module (0.1 A), with short protection function and overheat protection function	64 (64 outputs)	0.29	0.10			

2. SYSTEM CONFIGURATION

MELSEC-QnA

Module	Model	Description	Number of Occupied I/O Points (Module Type in I/O Allocation)	Current Consumption		Remark	
				5 VDC (A)	24 VDC (A)		
Dynamic input/output combination module	A42XY	64-input, 64-output, dynamic scanning	64 (64 outputs)	0.11	0.235	Performs I/O processing in 8-point units independently of the CPU module, while scanning.	
Input/output combination module	AH42	32-input, 32-output, 12/24 VDC transistor output module (0.1 A)	64 (64 outputs)	0.245	0.04	The first half 32 points are inputs and the latter half 32 points are outputs.	
Special function modules	Single-axis positioning module	AD70	For single-axis positioning control, speed control, and speed/position switching control. Analog voltage output (0 - ±10 V)	32 (special 32 points)	0.3	—	
		AD70D	1-axis, digital output, for MR-SB(K)/SD	32 (special 32 points)	0.8	—	
	Positioning module	AD71	For positioning control. Pulse train output, 2-axes (independent control, simultaneous 2-axis control, linear interpolation control). When in combination with AD76, stepping motors can be used.	32 (special 32 points)	1.5	—	
		AD71S7					
		AD71S1	For positioning control. (Dedicated to MELDAS-S1 support driver.) Pulse train output, 2-axes (independent control, simultaneous 2-axis control, linear interpolation control).	32 (special 32 points)	1.5	—	
		AD71S2	For positioning control, for high-speed control. Pulse train output, 2-axes (independent control, simultaneous 2-axis control, linear interpolation control). When in combination with AD76, stepping motors can be used.	32 (special 32 points)	1.5	—	
		AD72	For positioning control. Analog voltage output (0 - ±10 V) 2-axes (independent control, simultaneous 2-axis control, linear interpolation control).	48 first half: vacant 16 points latter half: special 32 points	0.9	—	
		AD76	Driver for stepping motor. Used in combination with AD71 or AD71S2.	16 (vacant 16 points)	—	—	
	Position detection module	A61LS	Absolute detection method Resolution: One resolver revolution = 4096 divisions Response speed: within 6 ms	48 first half: special 32 points latter half: vacant 16 points	0.8	—	
		A62LS	Absolute detection method, multiple rotation type Resolution: One resolver revolution = 4096 divisions Response speed: 2 ms	48 first half: special 32 points latter half: vacant 16 points	1.5	—	The resolution depends on the connected resolver.
	High-speed counter module	AD61	24-bit binary, 1/2 phase input, reversible counter, 50 kPPS, 2 channels	32 (special 32 points)	0.3	—	
		AD61S1	24-bit binary, 1/2 phase input, reversible counter, 1 phase ... 10 kPPS, 2 phases ... 7 kPPS 2 channels	32 (special 32 points)	0.3	—	
	A-D converter module	A68AD	4 to 20 mA / 0 to ±10 V, Analog input, 8 channels	32 (special 32 points)	0.9	—	
		A68AD-S2					
		A68ADN	0 to ±20 mA / 0 to ±10 V, Analog input, 8 channels	32 (special 32 points)	0.4	—	
A616AD		4 to 20 mA / 0 to ±10 V, Analog input, 16 channels Expansion to maximum of 121 channels possible by using A60MX (R)	32 (special 32 points)	1.0	—		
A60MX		Multiplex module (IC relay) Analog input, 16 channels	16 (vacant 16 points)	0.65	—	Used in combination with A616AD or A616TD.	
A60MXR	Multiplex module (mercury relay) Analog input, 16 channels	16 (vacant 16 points)	0.5	—			

2. SYSTEM CONFIGURATION

MELSEC-QnA

Module	Model	Description	Number of Occupied I/O Points (Module Type in I/O Allocation)	Current Consumption		Remark	
				5 VDC (A)	24 VDC (A)		
Special function modules	Temperature-digital converter module	A616TD	For temperature detection by thermocouple (when connected to A60MXT) 0 to ±10 V / 0 to 20 mA (when connected to A60MX(R))	32 (special 32 points)	1.0	—	
		A60MXT	Multiplex module. Temperature input: 15 channels. Temperature detection by thermocouple when used in conjunction with A616TD.	32 [first half: vacant 16 points latter half: vacant 16 points]	0.8	—	Used in combination with A616TD.
		A68RD3	- 180 to 600 °C temperature input module (For 3-wire type platinum resistor)	32 (special 32 points)	0.94	—	
		A68RD4	-180 to 600 °C temperature input module (For 4-wire type platinum resistor)		0.75	—	
	D-A converter module	A68DAV	0 to ±10 V, analog output, 8 channels.	32 (special 32 points)	0.15	0.2	
		A68DAI	0 to 20 mA, analog output, 8 channels.	32 (special 32 points)	0.15	0.4	
		A62DA	4 to 20 mA / 0 to ±10 V, Analog output 12 bits, 2 channels	32 (special 32 points)	0.6	0.35	
		A62DA-S1	4 to 20 mA / 0 to ±10 V, Analog output, 2 channels				
		A616DAI	4 to 20 mA. Resolution: 1/4000 Analog output, 16 channels	32 (special 32 points)	0.3	—	15 VDC (A68P) is required. +0.53 A -0.125 A
		A616DAV	0 to ±10 V/0 to ±5 V. Resolution: 1/4000 Analog output, 16 channels	32 (special 32 points)	0.38	—	15 VDC (A68P) is required. +0.2 A -0.17 A
	A-D, D-A converter module	A84AD	4 to 20 mA / 0 to ±10 V, Analog input/output 15 bits, 4 channels	48 [first half: vacant 16 points latter half: special 32 points]	0.24	0.53	
	CRT control module	AD57	CRT display, semigraphic Selection between color/monochrome possible.	64 (special 64 points)	1.21	0.16	To create canvas and character generator ROMs, use SW:JGP-AD57P.
		AD57S1			1.55		
	Memory card, Centronics interface module	AD59	32k byte memory battery backup Can be connected to printer conforming to Centronics standards.	32 (special 32 points)	0.3	—	0.35 A when connected to AD59MEF.
		AD59-S1			0.32		
Voice output module	A11VC	Messages can be recorded and played back on a maximum of 60 channels. The following recording times can be selected for each channel: 1 second, 2 seconds, 4 seconds, 8 seconds. The total recording time is 64 seconds.	16 (special 16 points)	0.6	0.38		
Network module	AJ71QLP21	For MELSECNET/10 optical loop networks	32 (special 32 points)	0.65	—	Up to 4 modules can be used with one CPU.	
	AJ71QLP21S	For MELSECNET/10 optical loop networks Network backup by external power supply	32 (special 32 points)	0.65	0.2		
	AJ71QBR11	For MELSECNET/10 coaxial bus networks	32 (special 32 points)	0.8	—		
	AJ71QLP25	For MELSECNET/10 optical loop network remote I/O stations	—	0.8	—		
	AJ71QBR15	For MELSECNET/10 coaxial bus network remote I/O stations	—	0.9	—		
Data link module	2 AJ71AP21	For MELSECNET II optical data links.	32 (special 32 points)	0.5	—	Up to 2 modules can be used with one CPU.	
	2 AJ71AR21	For MELSECNET II coaxial data links.	32 (special 32 points)	0.9	—		
	2 AJ71AT21B	For MELSECNET/B data links.	32 (special 32 points)	0.72	—		

2. SYSTEM CONFIGURATION

MELSEC-QnA

Module	Model	Description	Number of Occupied I/O Points (Module Type in I/O Allocation)	Current Consumption		Remark
				5 VDC (A)	24 VDC (A)	
Data link module	² AJ72T25B	For MELSECNET/B data link remote I/O stations.	—	0.3	—	
Serial communications module	AJ71QC24	Link module that communicates data with a computer. Transmission speed: 300 bps to 19.2 kbps RS-232C, RS422/485: one channel each	32 (special 32 points)	0.3	—	
	AJ71QC24-R2	Link module that communicates data with a computer. Transmission speed: 300 bps to 19.2 kbps RS-232C: two channels	32 (special 32 points)	0.2	—	
	AJ71QC24-R4	Link module that communicates data with a computer. Transmission speed: 300 bps to 19.2 kbps RS-422: two channels	32 (special 32 points)	0.38	—	
Computer link module (for AnUCPU)	² AJ71UC24	Link module that communicates data with a computer. Transmission speed: 300 bps to 19.2 kbps RS-232C, RS-422: one channel each, compatible with RS485	32 (special 32 points)	1.4	—	
Computer link module	³ AJ71C24(S3)	Link module that communicates data with a computer. Transmission speed: 300 bps to 19.2 kbps RS-232C, RS-422: one channel each	32 (special 32 points)	1.4	—	A total of up to 6 modules can be used with one CPU. *4 When power supply equipment is not connected. 0.7A
	² AJ71C24-S6/S8					
Intelligent communication module	³ AD51-S3	GPC-BASIC, maximum of 8 tasks RS-232C, RS-422: two channels each available.	48 { first half: vacant 16 points latter half: special 32 points }	1.3	—	
	² AD51H-S3	AD51H-BASIC, maximum of 8 tasks Equipped with IC memory card interface				
External fault diagnosis module	² AD51FD-S3	6 types of fault detection. Can output alarms and fault diagnosis data to external destinations.	48 { first half: vacant 16 points latter half: special 32 points }	1.0	—	
Graphic controller unit	² AD57G-S3	Monitor screen, monitor condition record, CRT, indicator, keyboard can be used.	48 { first half: vacant 16 points latter half: special 32 points }	1.3	—	
Ethernet interface module	² AJ71E71	10BASE5/10BASE2 specifications Transmission speed: 10 Mbps	32 (special 32 points)	1.5	—	
SUMINET interface module	³ AJ71P41	For SUMINET-3200 networks Communication speed: 2 Mbps	32 (special 32 points)	0.4 [*]	—	
Host controller high-speed link module	² AJ71C23-S3	Link module that sends/receive data at high speed to/from a computer. Transmission speed: 500 bps RS-422: one channel each	32 (special 32 points)	1.5	—	
Terminal interface module	AJ71C21	Link module that communicates data using a BASIC function terminal interface or the no-protocol mode. Transmission speed: 600 bps to 19.2 kbps RS-232C, RS-422: one channel each	32 (special 32 points)	0.8	—	
	AJ71C21S1					
Multidrop data link module	AJ71C22S1	Sends and receives bit data to a maximum of 8 slave stations to which it is connected in a multidrop system. Used for the master station of a multidrop link. Transmission speed: 38.4 kbps RS-422: one channel each	32 (special 32 points)	1.4	—	
	A0J2C25	Used as a remote I/O station of a multidrop link.	—	—	—	
	A0J2C214	Used as a local station in a multidrop link. (In A0J2CPU and A0J2HCPU systems, can also be used as the master station in computer links and multidrop data links.)	64 points	0.3	—	
Graphic operation terminal	² A64GOT-L A64GOT-LT21B	Compact graphic operation terminal Integral monitor (monochrome liquid crystal) Number of monitor screens: 250, number of parts: 255	—	—	0.4	

2. SYSTEM CONFIGURATION

MELSEC-QnA

Module	Model	Description	Number of Occupied I/O Points (Module Type in I/O Allocation)	Current Consumption		Remark	
				5 VDC (A)	24 VDC (A)		
Graphic operation terminal	2 A77GOT-S5	Large scale graphic operation terminal Integral monitor (monochrome liquid crystal, color liquid crystal, EL) Number of monitor screens: 250, number of parts: 255	—	—	—	When a bus connection is made, the number of occupied points is 32 (special).	
MELSEC-NET/MINI-S3 data link module	AJ71PT32-S3	Used for remote I/O control and remote terminal control with a total of 512 I/O points, for a maximum of 64 MELSECNET/MINI-S3 master stations.	I/O dedicated mode: 32 (special 32 points)	0.34	—		
	AJ71T32-S3		Extension mode: 48 (special 48 points)				
B/NET interface module	AJ71B62-S3	Used for B/NET transmission terminal control. Up to 63 stations can be controlled per module.	32 (special 32 points)	0.17	—		
Interrupt module	A161	Used to designate execution of interrupt programs (16 interrupt inputs).	32 (special 32 points)	0.14	—	Only one module can be used per CPU.	
Dummy module	AG62	Module allows selection of 16, 32, 48, or 64 points.	Number of set points ([] inputs)	0.07	—	Has 16 simulation switches.	
Blank cover	AG60	Keeps unused slots free of dust.	16 (vacant 16 points)	—	—		
Power supply module	Mounting position: power supply slot	A61P	Input: 100/200 VAC Output: 5 VDC 8 A	—	—		
		A61PEU					
		A62P	Input: 100/200 VAC Output: 5 VDC 5 A, 24 VDC 0.8 A				
		A62PEU					
		A63P	Input: 24 VDC Output: 5 VDC 8 A				
		A65P	Input: 100/200 VAC Output: 5 VDC 2 A, 24 VDC 1.5 A				
	A67P	Input: 110 VDC Output: 5 VDC 8A					
	Mounting position: I/O slot	A66P	Input: 100/200 VAC Output: 24 VDC 1.2 A				16 (vacant 16 points)
A68P		Input: 100 /200 VAC Output: +15 VDC 1.2 A, -15 VDC 0.7 A	32 (first half: vacant 16 points latter half: vacant 16 points)				
Base unit	Main base unit	A38HB	Can accommodate 8 I/O modules.	—	—	—	
		A38B	Can accommodate 8 I/O modules.				
		A35B	Can accommodate 5 I/O modules.				
		A32B	Can accommodate 2 I/O modules.				
		A32B-S1	Can accommodate 2 I/O modules.				
	Extension base unit	A68B	Can accommodate 8 I/O modules.	—	—	—	Power supply module required.
		A65B	Can accommodate 5 I/O modules.				
		A62B	Can accommodate 2 I/O modules.				
		A58B	Can accommodate 8 I/O modules.				
		A55B	Can accommodate 5 I/O modules.				
A52B	Can accommodate 2 I/O modules.						

2. SYSTEM CONFIGURATION

MELSEC-QnA

Module	Model	Description		Number of Occupied I/O Points (Module Type in I/O Allocation)	Current Consumption		Remark
					5 VDC (A)	24 VDC (A)	
Extension cable	AC06B	600 mm (23.62 in) long	Cables for connections between base units	—	—	—	
	AC12B	1200 mm (47.24 in) long					
	AC30B	3000 mm (118.11 in) long					
Simulation switch	A6SW16	16-point simulation switch		—	—	—	Installed in an input module.
	A6SW32	32-point simulation switch					
Battery	A6BAT	IC-RAM memory backup		—	—	—	
Miscellaneous Fuses	For AY11E, AY13E	MF51NM8	Cartridge type, 8 A	—	—	—	
	For AY22	HP-70K	Plug type, 7 A				
	For AY23	HP-32	Plug type, 3.2 A				
	For AY50, AY80	MP-20	Plug type, 2 A				
	For AY60	MP-32	Plug type, 3.2 A				
	For AY60E	MP-50	Plug type, 5 A				
	For power supply	GTH4	Cartridge type, 4 A				
	For A63P	SM6.3A	Cartridge type, 6.3 A				

*2: Only internal devices within the AnACPU range can be accessed (file registers cannot be accessed).

*3: Only internal devices within the A3HCPU range can be accessed (file registers cannot be accessed).

2. SYSTEM CONFIGURATION

MELSEC-QnA

Module	Model	Description	Applicable Models
Battery	A6BAT	IC-RAM memory backup	Installed in QnACPU module
Connector/terminal block converter module	A6TBXY36	For sink type input modules and sink type output modules (standard type)	AX42(S1), AY42(S1/S3/S4), AH42
	A6TBXY64	For sink type input modules and sink type output modules (2-wire type)	
	A6TBX70	For sink type input modules (3-wire type)	AX42(S1), AH42
	A6TBX36-E	For source type input modules (standard type)	AX82
	A6TBY36-E	For source type output modules (standard type)	AY82EP
	A6TBX54-E	For source type input modules (2-wire type)	AX82
	A6TBY54-E	For source type output modules (2-wire type)	AY82EP
	A6TBX70-E	For source type input modules (3-wire type)	AX82
Cable for connector/terminal block converter module	AC05TB	0.5 m (1.64 ft), for source module	A6TBXY36 A6TBXY64 A6TBX70
	AC10TB	1 m (3.28 ft), for source module	
	AC20TB	2 m (6.56 ft), for source module	
	AC30TB	3 m (9.84 ft), for source module	
	AC50TB	5 m (16.4 ft), for source module	
	AC05TB-E	0.5 m (1.64 ft), for source module	A6TBX36-E A6TBY36-E A6TBX54-E A6TBY54-E A6TBX70-E
	AC10TB-E	1 m (3.28 ft), for source module	
	AC20TB-E	2 m (6.56 ft), for source module	
	AC30TB-E	3 m (9.84 ft), for source module	
	AC50TB-E	5 m (16.4 ft), for source module	
Relay terminal module	A6TE2-16SR	For sink type output module	AY42, AY42-S1, AY42-S3, AY42-S4, AH42
Cable for connecting relay terminal module	AC06TE	0.6 m (1.97 ft) long	A6TE2-16SR
	AC10TE	1 m (3.28 ft) long	
	AC30TE	3 m (9.84 ft) long	
	AC50TE	5 m (16.4 ft) long	
	AC100TE	10 m (32.8 ft) long	

(2) Peripheral devices

Device	Model	Remark
Programming unit	Q6PU	Connected to the CPU module by an RS-422 cable (AC30R4-PUS, AC20R4-A8PU); for program writing and reading. (5 VDC 0.4A)
RS-422 cable	AC30R4-PUS	Cable for connection between CPU module and Q6PU. 3 m (9.84 ft) long
	AC20R4-A8PU	Cable for connection between CPU module and Q6PU. 2 m (6.56 ft) long

2. SYSTEM CONFIGURATION

2.3 Cautions about the System Configuration

This section describes the A series modules and software packages that can be used with QnACPU.

(1) Input/output modules

All building block type I/O modules can be used.

(2) The A38HB high speed access main base unit

By using the A38HB high speed access base unit, the data transfer speed between the QnACPU and the special function module becomes about twice as fast.

The scan time can also be reduced when a module that handles a large capacity of data, such as a network module, is mounted on the A38HB base unit.

(3) Special function module

(a) The following special function modules cannot be used.

- AD57-S2 (CRT controller module)
- AJ71LP21, AJ71BR11 (AnU complying MELSECNET/10 network module)
- AJ71C23 (Host controller high speed link module)
- AD51 (Intelligent communication module)
 - AD51 not complying with A3HCPU (those manufactured prior to March 1987)
- AJ71C24 (computer link module)
 - AJ71C24 not complying with A3HCPU (those manufactured prior to February 1987)

(b) Special function modules with partial restrictions

Model Name	Function Restrictions	Restrictions on Number of Modules Mounted	
AD51(S3) AJ71C24(S3) AJ71P41	Only the devices ranges shown below corresponding to A3HCPU can be accessed. (However, file register and program reading/writing is not available) X/Y0 to 7FF, M/S/L0 to 2047, B0 to 3FF, T0 to 255, C0 to 255, D0 to 1023, W0 to 3FF, F0 to 255	Up to 6 modules can be mounted.	
AD51H(S3) AD51FD-S3 AJ71C23-S3 AJ71C24-S6/S8 AJ71UC24 AJ71E71	Only the device ranges shown below corresponding to A3ACPU can be accessed. (However, file register and program reading/writing is not available) X/Y0 to 7FF, M/S/L0 to 8191, B0 to FFF, T0 to 2047, C0 to 1023, D0 to 6143, W0 to FFF, F0 to 2047		
AJ71AP21 AJ71AR21 AJ71AT21B	—	Maximum of 2 modules can be mounted	Maximum of 4 modules in total can be mounted
AJ71QLP21 AJ71QBR11	—	Maximum of 4 modules can be mounted	

REMARK

There is no restriction on the number of AJ71QC24/AJ71QC24-R2/AJ71QC24-R4 serial communication modules that can be mounted.

(4) Graphic operation terminal (GOT)

Only GOTs which are in the device range corresponding to A3ACPU can access QnACPU.

There are no restrictions on the GOT and QnACPU connection method.

Model Name	Connection Method	Accessible Device Range
A64GOT-LT21B	CPU module direct connection, MELSECNET/B connection possible	Only the device ranges shown below corresponding to A3ACPU can be accessed. (However, file register reading/writing, system monitoring and ladder monitoring is not available) X/Y0 to 7FF, M/S/L0 to 8191, B0 to FFF, T0 to 2047, C0 to 1023, D0 to 6143 W0 to FFF, F0 to 2047
A77GOT	CPU module direct connection, MELSECNET(II) and MELSECNET/B connections possible	
A77GOT-S3	CPU module direct connection, MELSECNET(II), MELSECNET/B and MELSECNET/10 connections possible	
A77GOT-S5	CPU module direct connection, MELSECNET(II), MELSECNET/B, MELSECNET/10, computer link module connection and bus connections possible	

(5) Software packages

The system startup software packages that can be used to create programs for QnACPU are indicated below.

Model Name	Connection Method	Accessible Device Range
SW0IVD-GPPQ	Software package capable of GPP functions (containing SFC functions)	IBM PC/AT or 100 % compatible (See Section 4.2)

REMARK

The peripheral devices and software packages that can be used with QnACPU are listed below:

- Programming unit..... A6PU, A7PU, A7PUS, A8PU
- P-ROM writer unit A6WU
- Data access unit A6DU-B
- Modem interface unit..... A6TEL
- Intelligent GPP..... A6GPP
- Handy graphic programmer A6HGP
- Plasma handy graphic programmer A6PHP
- System startup software package for ACPU SW[]-GPPA, SW[]-SAP2
- Utility software package for ACPU SW[]GHP-UTLP-FN0,
SW[]GHP-UTLP-FN1

2.4 Selecting Memory Card Models

Since the QnACPU is provided with an internal memory to store parameters and programs as a standard feature, programs can be executed without installing a memory card.

The program capacities of the internal memories of the various models of CPU module are indicated below.

Q2ACPU	28k steps	(112k byte internal memory)
Q2ACPU-S1	60k steps	(240k byte internal memory)
Q3ACPU	92k steps	(368k byte internal memory)
Q4ACPU	124k steps	(496k byte internal memory)

(1) Applications of memory cards

A memory card is required in the following cases:

(a) To perform a boot operation

Parameters, programs, device initial values, separate comments, and boot files are stored in the memory card; on program execution they are read to internal memory and executed.

(b) To use file registers.**(c) To use local devices.****(d) To use the sampling trace function.****(e) To use the status latch function.****(f) To use the program trace function.****(g) To store the fault history in a file.****(h) To execute the maximum number of steps possible with a QnACPU.**

When a program of the maximum capacity is stored in the internal memory, the parameter files and device initial values must be stored in a memory card.

(i) To use the SFC trace function.

2. SYSTEM CONFIGURATION

MELSEC-QnA

(2) Selecting memory card

Select the memory card according to the type of files to be stored in the memory card and the size of these files. The sizes of files is calculated using the formulae presented below.

Function	Approximate File Capacity (Units: Bytes)
Drive comment	64
Password	72
Parameters	MELSECNET, NET/10 None → 330 When MELSECNET (II, /B) set → max. 4096 per module
Boot file	(Number of files x 18) + 67
Sequence program	(Number of steps x 4) + 122
Device comments	(34 x number of comment points) + (device types ^{*1} x 10) + 64
Device initial values	(Number of device points x 2) + (device types x 44) + 66
File registers	Number of file register points x 2 bytes
Simulation data	(Number of word device points x 2) + $\frac{(\text{number of bit device points} + 16) \times 2}{\text{Rounded up}}$ + (device ranges ^{*2} x 44) + 66
Sampling trace data	{Added information + (number of word devices x 2) + $\frac{(\text{number of bit device points} + 16) \times 2}{\text{Rounded up}}$ x trace count + (number of device points x 12) + 362
Status latch data	For all devices : 58576 For designated devices : (Number of word device points x 2) + $\frac{(\text{number of bit device points} + 16) \times 2}{\text{Rounded up}}$ x 2 + (device types x 8) + 352
Program trace data	Same as sampling trace
Fault history data	54 x number of faults stored + 72 bytes
SFC trace data	Max. 48 k (in 1 kbyte units)

*1: "Device types" means the number of registered device names.

For example, if D, W, and T are registered, it is 3 types.

*2: "Device ranges" means the number of registered range settings.

<NOTE>

Note that capacities will be secured by rounding up in the units indicated below according to the memory area used for storage:

Internal memory 4096 bytes (1k steps) units
Memory card 512 byte units

3. QnACPU

3.1 General Specifications

The common specifications for each type of module are indicated in the table below.

Table 3.1 General Specifications

Item	Specifications				
Operating ambient temperature	0 to 55 °C				
Storage ambient temperature	-20 to 75 °C				
Operating ambient humidity	10 to 90% RH (no dewing)				
Storage ambient humidity	10 to 90% RH (no dewing)				
Vibration resistance	Conforms to JIS B 3501, IEC 1131-2 ³	Intermittent vibration			10 times in each of X, Y and Z directions (for 80 minutes)
		Frequency	Acceleration	Amplitude	
		10 to 57 Hz	—	0.075 mm (0.003 in)	
		57 to 150 Hz	9.8 m/s ² {1G}	—	
		Continuous vibration			
		Frequency	Acceleration	Amplitude	
		10 to 57 Hz	—	0.035 mm (0.001 in)	
		57 to 150 Hz	4.9 m/s ² {0.5G}	—	
Shock resistance	Conforms to JIS B 3501, IEC 1131-2 (147 m/s ² {15G}, 3 times in each of 3 directions)				
Operating atmosphere	Free of corrosive gases				
Operating altitude	No greater than 2000 m (6561.7 ft)				
Installation site	Control panel				
Overvoltage category ^{*1}	11 or lower				
Contamination level ^{*2}	2 or lower				

*1 Shows which power distribution part is assumed to be connected from between the public power distribution network and the internal device apparatus.

Category 11 applies to devices that receive power from fixed facilities.

Machines rated up to 300 V can withstand voltage surges of up to 2,500 V.

*2 An indicator showing the rate of occurrence of conductivity material in the environment used by the device.

Contamination level 2 only occurs with non-conductive contamination.

However, in this environment temporary conductivity might occur due to condensation.

*3 JIS: Japanese Industrial Standard

3. QnACPU

3.2 Performance Specifications

3.2.1 Performance specifications

This section describes the performance specifications for the QnACPU module.

Item	Model Name				Remarks	
	Q2ACPU	Q2ACPU-S1	Q3ACPU	Q4ACPU		
Control mode	Stored program repeated operations					
I/O control mode	Refresh mode				Direct I/O possible according to device name	
Programming language	Language dedicated to sequence control					
	Relay symbol language, logic symbolic language, MELCAP-3 (SFC)					
Processing speed (sequence instructions) (μs/step)	LD	0.2	0.15	0.075		
	MOV	0.6	0.45	0.225		
Number of instructions (types)	Sequence instructions	39				
	Basic instructions	230				
	Application instructions	321				
	Special dedicated instructions	171				
Constant scan (ms) (program starts at fixed time interval)	5 to 2000 (can be set in 5 ms units)				Set according to parameters	
Memory capacity	Installed memory card memory (maximum 2036 KB)					
Program capacity	Number of steps	Maximum 28k	Maximum 60k	Maximum 92k	Maximum 124k	
	Number of files	28	60	92	124	
Number of I/O device points (points)	8192 (X/Y0 to 1FFF)				Number of points that can be used in program	
Number of I/O points (points)	512 (X/Y0 to 1FF)	1024 (X/Y0 to 3FF)	2048 (X/Y0 to 7FF)	4096 (X/Y0 to FFF)	Number of points that can access actual input/output module	
Number of device points	Internal relay [M] (points)	Default 8192 (M0 to 8191)				Set number of points used according to parameters
	Latch relay [L] (points)	Default 8192 (L0 to 8191)				
	Link relay [B] (points)	Default 8192 (B0 to 1FFF)				
	Timer [T] (points)	Default 2048 (T0 to 2047) (shared low speed/high speed) Low speed/high speed switching is set in instructions Low speed/high speed measurement unit is set in parameters (Low speed: Set in 10 ms units in range 10 to 1000 ms, default 100 ms) (High speed: Set in 1 ms units in range 1 to 100 ms, default 10 ms)				
	Retentive timer [ST] (points)	Default 0 (ST0 to 2047)				
	Counter [C] (points)	• Normal counter default 1024 (C0 to 1023) • Interrupt counter maximum 48 (default 0 points, set according to parameters)				
	Data register [D] (points)	Default 12288 (D0 to 12287)				
	Link register [W] (points)	Default 8192 (W0 to 1FFF)				
	Annunciator [F] (points)	Default 2048 (F0 to 2047)				
	Edge relay [V] (points)	Default 2048 (V0 to 2047)				
	Special link relay [SB] (points)	Default 2048 (SB0 to 7FF)				
	Special link register [SW] (points)	Default 2048 (SW0 to 7FF)				
	File register [R] (points)	32768 (R0 to 32767), maximum of 1042432 points can be used by block switching				

3. QnACPU

MELSEC-QnA

Item	Model Name				Remarks
	Q2ACPU	Q2ACPU-S1	Q3ACPU	Q4ACPU	
Number of device points	File register [R] (points)	1042432 (ZR0 to 1042431), block switching not required			Set number of points used according to parameters
	Step relay [S] (points)	8192 (S0 to 8191)			Number of device points is fixed
	Index register [Z] (points)	16 (Z0 to 15)			
	Pointer [P] (points)	4096 (P0 to 4095), file pointer/common pointer usage range set according to parameters			
	Interrupt pointer [I] (points)	48 (I0 to 47)			
	Special relay [SM] (points)	2048 (SM0 to 2047)			
	Special register [SD] (points)	2048 (SD0 to 2047)			
	Input bit condition device for subroutine call with argument [FX] (points)	16 (FX0 to F)			
	Output bit condition device for subroutine call with argument [FY] (points)	16 (FY0 to F)			
	Input/output data condition device for subroutine call with argument [FD] (points)	5 (FD0 to 4)			
Link direct device	Device directly accessing link device. Only for MELSECNET/10. Designated format: J[]N[]				
Special function module direct device	Device directly accessing special function module buffer memory. Designated format: U[]G[]				
Latch (memory back up) range	L0 to 8191 (fixed) (Latch range can be set for B, F, V, T, ST, C, D, W)			Set according to parameters	
Remote RUN/PAUSE contact	1 point can be set for each RUN/PAUSE contact from X0 to 1FFF				
Clock function	Year, month, day, hour, minute, second, day of week (automatically detects leap years) Precision -2.3 to +4.4 s (TYP. +1.8 s)/d at 0 °C Precision -1.1 to +4.4 s (TYP. +2.2 s)/d at 25 °C Precision -9.6 to +2.7 s (TYP. +2.4 s)/d at 55 °C				
5 VDC internal current consumption (A)	0.3		0.6		

3.2.2 Devices

The names and data ranges of devices which can be used at the QnACPU are shown in Table below.

Device List

Class	Type	Device Name	Default Values		Parameter Designated Setting Range
			Number of Points	Range Used	
Internal user devices	Bit devices	Input	8192 points	X0 to X1FFF	Change possible for 28.75k words or less. (However, the number of step relay points is fixed at 8192.)
		Output	8192 points	Y0 to Y1FFF	
		Internal relay	8192 points	M0 to M8191	
		Latch relay	8192 points	L0 to L8191	
		Annunciator	2048 points	F0 to F2047	
		Edge relay	2048 points	V0 to V2047	
		Link relay	8192 points	B0 to B1FFF	
		Special link relay	2048 points	SB0 to SB7FF	
		Step relay	8192 points	S0 to S511 per block	
	Word devices	Timer *1	2048 points	T0 to T2047	
		Retentive timer *1	0 points	(ST0 to ST2047)	
		Counter *1	1024 points	C0 to C1023	
		Data register	12288 points	D0 to D12287	
		Link register	8192 points	W0 to W1FFF	
Special link register		2048 points	SW0 to SW7FF		
Internal system devices	Bit devices	Function input	16 points	FX0 to FXF	Impossible
		Function output	16 points	FY0 to FYF	
		Special relay	2048 points	SM0 to SM2047	
	Word devices	Function register	5 points	FD0 to FD4	
		Special register	2048 points	SD0 to SD2047	
Link direct devices	Bit device	Link Input	8192 points	Jn\X0 to Jn\X1FFF	Impossible
		Link output	8192 points	Jn\Y0 to Jn\Y1FFF	
		Link relay	8192 points	Jn\B0 to Jn\B1FFF	
		Link special relay	2048 points	Jn\SB0 to Jn\SB7FF	
	Word device	Link register	8192 points	Jn\W0 to Jn\W1FFF	
		Link special register	2048 points	Jn\SW0 to Jn\SW7FF	
Special direct device	Word device	Buffer register	16384 points	Un\G0 to Un\G16383	Impossible
Index register	Word device	Index register	16 points	Z0 to Z15	Impossible
File register	Word device	File register	0 points	—	Set the number of points with the parameters.

Class	Type	Device Name	Default Values		Parameter Designated Setting Range
			Number of Points	Range Used	
Nesting	—	Nesting	15 points	N0 to N14	Impossible
Pointers	—	Pointer	4096 points	P0 to P4095	Impossible
		Interrupt pointer	48 points	I0 to I47	
Other	Bit devices	SFC block	320 points	BL0 to BL319	Impossible
		SFC transition device	512 points	TR0 to TR511	
	—	Network No.	—	J1 to J255	
		I/O No.	—	U0 to UFF	
Constants	—	Decimal constants	K-2147473648 to K214747364		
		Hexadecimal constants	H0 to HFFFFFFF		
		Real number constants	E ± 1.17549 ⁻³⁸ to E ± 3.40282 ^{*38}		
		Character string constants	"ABC", "123" etc.		

REMARK

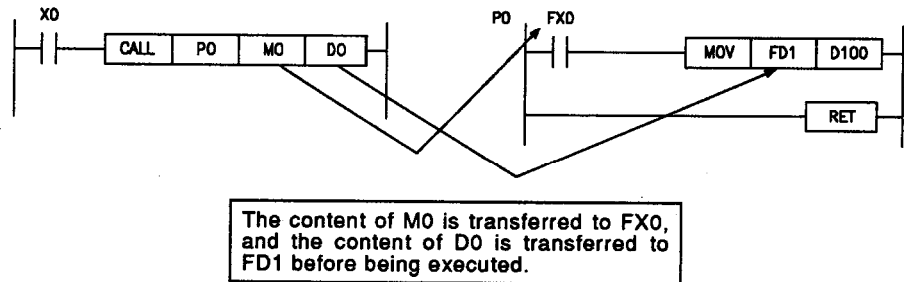
*1: For the timer, retentive timer, and counter, bit devices are used for the "number of points" and the "coil", and the word device is used for the "present value".

The following devices are added from the MELSEC-A series, making program creation easier. This section describes these added devices.

(1) Argument devices (FX, FY, FD)

These devices are used in data transfer between a call source and call destination using subroutine calls with arguments (CALL).

- Each device is used as outlined below.
- FX → Bit conditions input by subroutine
- FY → Output bit conditions
- FD → Input/output data conditions



(2) Special relays/special registers (SM/SD)

The special relays/special registers are for conducting communications between the QnACPU and user programs. The special relays/special registers (SM/SD) include the following:

- SM1 Self-diagnosis error
- SM52 Low battery voltage

(3) Step relays (S)

These are SFC dedicated relays showing the active status of each step in the SFC.

The step relay can be designated and blocks attached in the sequence program.

Example: BL2\S1. . . . Designate block No. 2 step relay 1

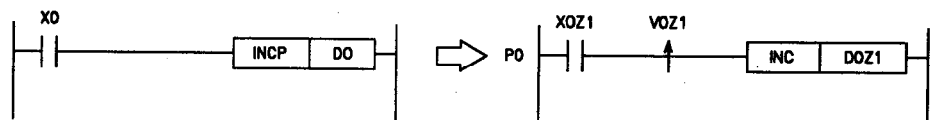
(4) Edge relays (V)

These devices are used in programs that are executed repeatedly when conducting pulse conversion, such as subroutine and interrupt programs.

Pulse conversion instructions, such as subroutine and interrupt programs, become easier to use by using edge relays.

Example: Normal program

Subroutine program



(5) Retentive timer (ST)

The device name for the retentive timer is ST so as to be able to distinguish it from the normal timer.

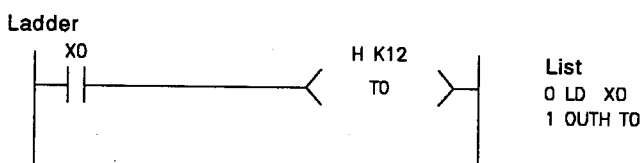
Example: OUT ST100 K500

(6) Low speed timer/high speed timer (T)

It is now possible to change the measurement unit. Change the settings in the parameters.

You can also now distinguish between low speed timers and high speed timers in programs.

Example: Low speed timer OUT T200 K120
High speed timer OUTH T200 K120



(7) Link direct devices (J[]N[])

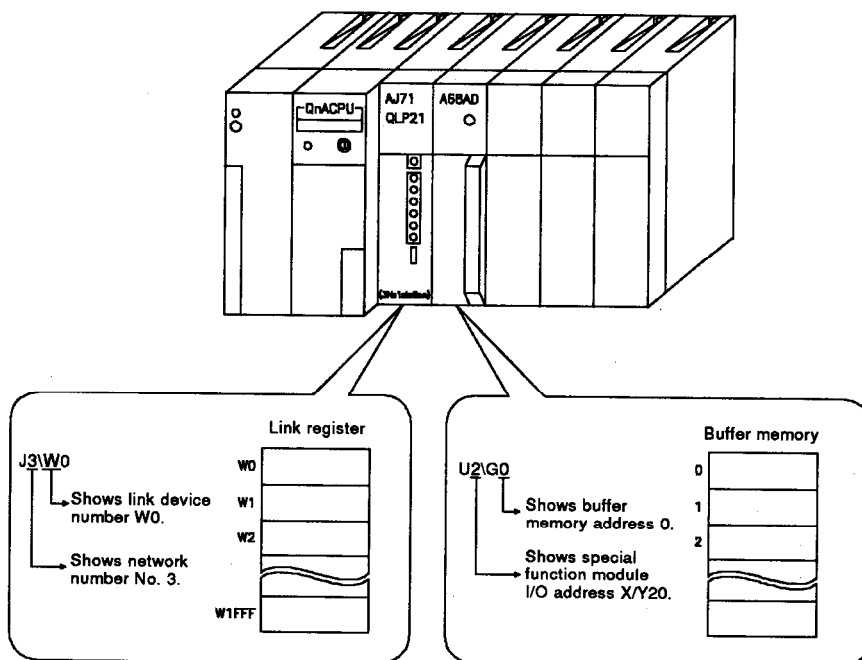
These devices allow you to directly access MELSECNET/10 network module link devices.

By using these devices, you can reduce the link device transmission time.

You can also access link ranges that are not set in the network refresh parameters.

(8) Special direct devices (U[]G[])

This device can be programmed to handle the special function module buffer memory like a data register.

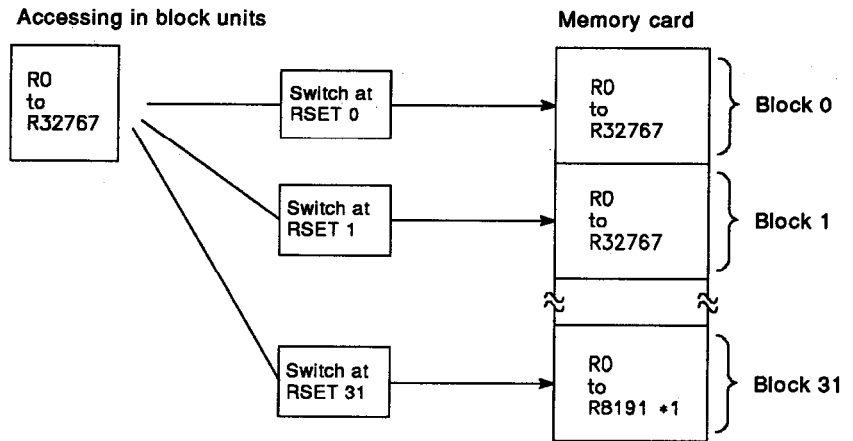


(9) File registers (R/ZR)

These are registers for extending data registers. To use file registers you must have a memory card.

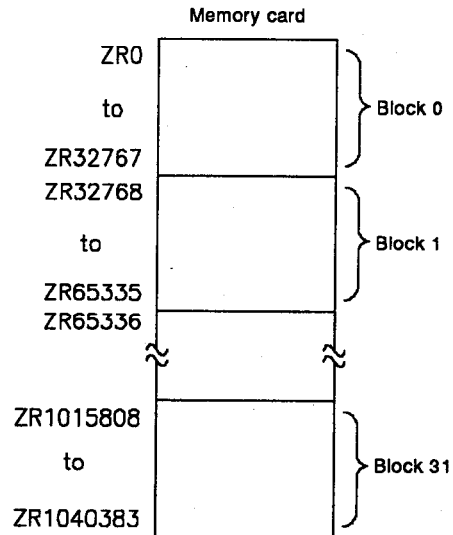
The maximum file register capacity is 1018k words, which can be accessed in 32k word block units, or instead the whole file register can be accessed using serial numbers.

(a) Accessing file registers in block units



*1 There are 8192 points in block No. 31.













(b) Accessing the file register using serial numbers



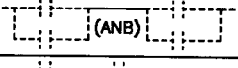
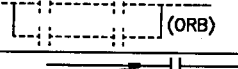
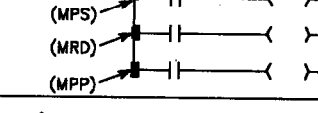
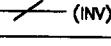
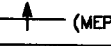
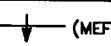
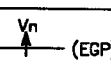
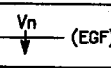
3.3 Instruction Lists

3.3.1 Sequence instructions

(1) Contact instruction

Category	Symbol	Processing Details
Contact	 (LD)	• Starts logic operation (Starts N/O contact logic operation)
	 (LDI)	• Starts logical NOT operation (Starts N/C contact logic operation)
	 (AND)	• Logical product (N/O contact series connection)
	 (ANI)	• Logical product NOT (N/C contact series connection)
	 (OR)	• Logical sum (N/O contact parallel connection)
	 (ORI)	• Logical sum NON (N/C contact parallel connection)
	 (LDP)	• Starts leading edge pulse operation
	 (LDF)	• Starts trailing edge pulse operation
	 (ANDP)	• Leading edge pulse series connection
	 (ANDF)	• Trailing edge pulse series connection
	 (ORP)	• Leading edge pulse parallel connection
	 (ORF)	• Trailing edge pulse parallel connection

(2) Connection instructions

Category	Symbol	Processing Details
Connection	 (ANB)	• AND between logical blocks (Series connection between logical block)
	 (ORB)	• OR between logical blocks (Series connection between logical blocks)
	 (MPS), (MRD), (MPP)	• Memory storage of operation results • Read operation of operation results stored with MPS instruction • Read and reset of operation results stored with MPS instruction
	 (INV)	• Inversion of operation result
	 (MEP)	• Conversion of operation result to leading edge pulse
	 (MEF)	• Conversion of operation result to trailing edge pulse
	 (EGP)	• Conversion of operation result to leading edge pulse (Stored at Vn)
	 (EGF)	• Conversion of operation result to trailing edge pulse (Stored at Vn)

(3) Output instructions

Category	Symbol	Processing Details
Output		• Device output
		• Set device
		• Reset device
		• Generates 1 cycle program pulse at leading edge of input signal
		• Generates 1 cycle program pulse at trailing edge of input signal
		• Reversal of device output
		• Pulse conversion of direct output

(4) Shift instructions

Category	Symbol	Processing Details
Shift		• 1-bit shift of device

(5) Master control instructions

Category	Symbol	Processing Details
Master control		• Starts master control
		• Resets master control

(6) Termination instruction

Category	Symbol	Processing Details
Program end		• Termination of main program
		• Termination of sequence program


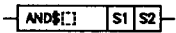
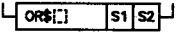
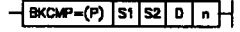
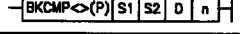
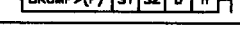
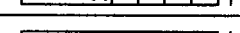
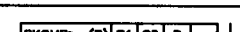

(7) Other instructions

Category	Symbol	Processing Details
Stop		<ul style="list-style-type: none"> • Terminates sequence operation after input condition has been met • Sequence program is executed by placing the RUN/STOP key switch back in the RUN position
Ignored	(NOP)	• Ignored (For program deletion or space)
		• Ignored (To change pages during printouts)
		• Ignored (Subsequent programs will be controlled from step 0 of page n)

3.3.2 Basic instructions

(1) Comparison operation instruction

Category	Symbol	Processing Details
16-bit data comparisons		<ul style="list-style-type: none"> Conductive state when (S1) = (S2) Non-conductive state when (S1) ≠ (S2)
		<ul style="list-style-type: none"> Conductive state when (S1) ≠ (S2) Non-conductive state when (S1) = (S2)
		<ul style="list-style-type: none"> Conductive state when (S1) > (S2) Non-conductive state when (S1) ≤ (S2)
		<ul style="list-style-type: none"> Conductive state when (S1) ≤ (S2) Non-conductive state when (S1) > (S2)
		<ul style="list-style-type: none"> Conductive state when (S1) < (S2) Non-conductive state when (S1) ≥ (S2)
		<ul style="list-style-type: none"> Conductive state when (S1) ≥ (S2) Non-conductive state when (S1) < (S2)
32-bit data comparisons		<ul style="list-style-type: none"> Conductive state when (S1+1, S1) = (S2+1, S2) Non-Conductive state when (S1+1, S1) ≠ (S2+1, S2)
		<ul style="list-style-type: none"> Conductive state when (S1+1, S1) ≠ (S2+1, S2) Non-Conductive state when (S1+1, S1) = (S2+1, S2)
		<ul style="list-style-type: none"> Conductive state when (S1+1, S1) > (S2+1, S2) Non-Conductive state when (S1+1, S1) ≤ (S2+1, S2)
		<ul style="list-style-type: none"> Conductive state when (S1+1, S1) ≤ (S2+1, S2) Non-Conductive state when (S1+1, S1) > (S2+1, S2)
		<ul style="list-style-type: none"> Conductive state when (S1+1, S1) > (S2+1, S2) Non-Conductive state when (S1+1, S1) ≥ (S2+1, S2)
		<ul style="list-style-type: none"> Conductive state when (S1+1, S1) ≥ (S2+1, S2) Non-Conductive state when (S1+1, S1) < (S2+1, S2)
Real number data comparisons		<ul style="list-style-type: none"> Conductive state when (S1+1, S1) = (S2+1, S2) Non-Conductive state when (S1+1, S1) ≠ (S2+1, S2)
		<ul style="list-style-type: none"> Conductive state when (S1+1, S1) ≠ (S2+1, S2) Non-Conductive state when (S1+1, S1) = (S2+1, S2)
		<ul style="list-style-type: none"> Conductive state when (S1+1, S1) > (S2+1, S2) Non-Conductive state when (S1+1, S1) ≤ (S2+1, S2)
		<ul style="list-style-type: none"> Conductive state when (S1+1, S1) ≤ (S2+1, S2) Non-Conductive state when (S1+1, S1) > (S2+1, S2)
		<ul style="list-style-type: none"> Conductive state when (S1+1, S1) < (S2+1, S2) Non-Conductive state when (S1+1, S1) ≥ (S2+1, S2)
		<ul style="list-style-type: none"> Conductive state when (S1+1, S1) ≥ (S2+1, S2) Non-Conductive state when (S1+1, S1) < (S2+1, S2)

Category	Symbol	Processing Details
Character string data comparisons		<ul style="list-style-type: none"> • Compares character string S1 and character string S2 one character at a time. <ul style="list-style-type: none"> Match: All characters in the strings must match Larger string: If character strings are different, determines the string with the largest number of character codes. If the lengths of the character strings are different, determines the longest character string Smaller string: If the character strings are different, determines the string with the smallest number of character codes. If the lengths of the character strings are different, determines the shortest character string
		<ul style="list-style-type: none"> • Conductive state when (character string S1) = (character string S2) • Non-conductive state when (character string S1) ≠ (character string S2)
		<ul style="list-style-type: none"> • Conductive state when (character string S1) > (character string S2) • Non-conductive state when (character string S1) ≤ (character string S2)
		<ul style="list-style-type: none"> • Conductive state when (character string S1) < (character string S2) • Non-conductive state when (character string S1) ≠ (character string S2)
Block data comparisons		<ul style="list-style-type: none"> • Compares n points of data from S1 to n points of data from S2 in 1-word units, and stores the results of the comparison at n points from the bit device designated by (D).
		
		
		
		
		

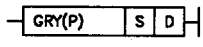
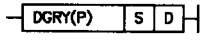
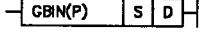
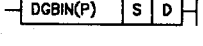
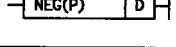
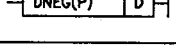
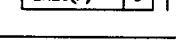
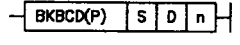

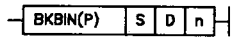

(2) Arithmetic operation instructions

Category	Symbol	Processing Details
BIN 16-bit addition and subtraction operations	$\text{+}(P) \quad S \quad D$	• $(D) + (S) \rightarrow (D)$
	$\text{+}(P) \quad S1 \quad S2 \quad D$	• $(S1) + (S2) \rightarrow (D)$
	$\text{-}(P) \quad S \quad D$	• $(D) - (S) \rightarrow (D)$
	$\text{-}(P) \quad S1 \quad S2 \quad D$	• $(S1) - (S2) \rightarrow (D)$
BIN 32-bit addition and subtraction operations	$\text{D+}(P) \quad S \quad D$	• $(D + 1, D) + (S + 1, S) \rightarrow (D + 1, D)$
	$\text{D+}(P) \quad S1 \quad S2 \quad D$	• $(S1 + 1, S1) + (S2 + 1, S2) \rightarrow (D + 1, D)$
	$\text{D-}(P) \quad S \quad D$	• $(D + 1, D) - (S + 1, S) \rightarrow (D + 1, D)$
	$\text{D-}(P) \quad S1 \quad S2 \quad D$	• $(S1 + 1, S1) - (S2 + 1, S2) \rightarrow (D + 1, D)$
BIN 16-bit multiplication and division operations	$\text{*}(P) \quad S1 \quad S2 \quad D$	• $(S1) \times (S2) \rightarrow (D + 1, D)$
	$\text{/}(P) \quad S1 \quad S2 \quad D$	• $(S1) / (S2) \rightarrow \text{Quotient}(D), \text{Remainder}(D + 1)$
BIN 32-bit multiplication and division operations	$\text{D*}(P) \quad S1 \quad S2 \quad D$	• $(S1 + 1, S1) \times (S2 + 1, S2) \rightarrow (D + 3, D + 2, D + 1, D)$
	$\text{D/}(P) \quad S1 \quad S2 \quad D$	• $(S1 + 1, S1) / (S2 + 1, S2) \rightarrow \text{Quotient}(D + 1, D), \text{Remainder}(D + 3, D;2)$
BCD 4-digit addition and subtraction operations	$\text{B+}(P) \quad S \quad D$	• $(D) + (S) \rightarrow (D)$
	$\text{B+}(P) \quad S1 \quad S2 \quad D$	• $(S1) + (S2) \rightarrow (D)$
	$\text{B-}(P) \quad S \quad D$	• $(D) - (S) \rightarrow (D)$
	$\text{B-}(P) \quad S1 \quad S2 \quad D$	• $(S1) - (S2) \rightarrow (D)$
BCD 8-digit addition and subtraction operations	$\text{DB+}(P) \quad S \quad D$	• $(D + 1, D) + (S + 1, S) \rightarrow (D + 1, D)$
	$\text{DB+}(P) \quad S1 \quad S2 \quad D$	• $(S1 + 1, S1) + (S2 + 1, S2) \rightarrow (D + 1, D)$
	$\text{DB-}(P) \quad S \quad D$	• $(D + 1, D) - (S + 1, S) \rightarrow (D + 1, D)$
	$\text{DB-}(P) \quad S1 \quad S2 \quad D$	• $(S1 + 1, S1) - (S2 + 1, S2) \rightarrow (D + 1, D)$
BCD 4-digit multiplication and division operations	$\text{B*}(P) \quad S1 \quad S2 \quad D$	• $(S1) \times (S2) \rightarrow (D + 1, D)$
	$\text{B/}(P) \quad S1 \quad S2 \quad D$	• $(S1) / (S2) \rightarrow \text{Quotient}(D), \text{Remainder}(D + 1)$
BCD 8-digit multiplication and division operations	$\text{DB*}(P) \quad S1 \quad S2 \quad D$	• $(S1 + 1, S1) \times (S2 + 1, S2) \rightarrow (D + 3, D + 2, D + 1, D)$
	$\text{DB/}(P) \quad S1 \quad S2 \quad D$	• $(S1 + 1, S1) / (S2 + 1, S2) \rightarrow \text{Quotient}(D + 1, D), \text{Remainder}(D + 3, D + 2)$
Floating decimal point data addition and subtraction operations	$\text{E+}(P) \quad S \quad D$	• $(D + 1, D) + (S + 1, S) \rightarrow (D + 1, D)$
	$\text{E+}(P) \quad S1 \quad S2 \quad D$	• $(S1 + 1, S1) + (S2 + 1, S2) \rightarrow (D + 1, D)$
	$\text{E-}(P) \quad S \quad D$	• $(D + 1, D) - (S + 1, S) \rightarrow (D + 1, D)$
	$\text{E-}(P) \quad S1 \quad S2 \quad D$	• $(S1 + 1, S1) - (S2 + 1, S2) \rightarrow (D + 1, D)$

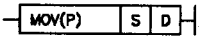
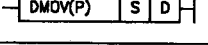
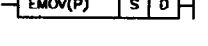
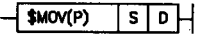

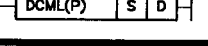
Category	Symbol	Processing Details
Floating decimal point data multiplication and division operations	$\text{E}\times(\text{P}) \quad \text{S1} \quad \text{S2} \quad \text{D}$	• $(\text{S1} + 1, \text{S1}) \times (\text{S2} + 1, \text{S2}) \rightarrow (\text{D} + 1, \text{D})$
	$\text{E}/(\text{P}) \quad \text{S1} \quad \text{S2} \quad \text{D}$	• $(\text{S1} + 1, \text{S1}) / (\text{S2} + 1, \text{S2}) \rightarrow \text{Quotient } (\text{D} + 1, \text{D})$
Character string data addition	$\text{\$}+(\text{P}) \quad \text{S} \quad \text{D}$	• Links character string designated with (S) to character string designated with (D), and stores the result from (D) onward.
	$\text{\$}+(\text{P}) \quad \text{S1} \quad \text{S2} \quad \text{D}$	• Links character string designated with (S2) to character string designated with (S1), and stores the result from (D) onward.
BIN block addition and subtraction operations	$\text{BK}+ \quad \text{S1} \quad \text{S2} \quad \text{D} \quad \text{n}$	• Adds data n points from (S1) to data n points from (S2) in batch.
	$\text{BK}- \quad \text{S1} \quad \text{S2} \quad \text{D} \quad \text{n}$	
BIN data increment	$\text{INC}(\text{P}) \quad \text{D}$	• $(\text{D}) + 1 \rightarrow (\text{D})$
	$\text{DINC}(\text{P}) \quad \text{D}$	• $(\text{D} + 1, \text{D}) + 1 \rightarrow (\text{D})$
BIN data decrement	$\text{DEC}(\text{P}) \quad \text{D}$	• $(\text{D}) - 1 \rightarrow (\text{D})$
	$\text{DDEC}(\text{P}) \quad \text{D}$	• $(\text{D} + 1, \text{D}) - 1 \rightarrow (\text{D})$

(3) Data conversion instructions

Category	Symbol	Processing Details
BCD conversions	$\text{BCD}(\text{P}) \quad \text{S} \quad \text{D}$	• $(\text{S}) \xrightarrow{\text{BCD conversion}} (\text{D})$ BIN (0 to 9999)
	$\text{DBCDC}(\text{P}) \quad \text{S} \quad \text{D}$	• $(\text{S} + 1, \text{S}) \xrightarrow{\text{BCD conversion}} (\text{D} + 1, \text{D})$ BIN (0 to 99999999)
BIN conversions	$\text{BIN}(\text{P}) \quad \text{S} \quad \text{D}$	• $(\text{S}) \xrightarrow{\text{BIN conversion}} (\text{D})$ BCD (0 to 9999)
	$\text{DBIN}(\text{P}) \quad \text{S} \quad \text{D}$	• $(\text{S} + 1, \text{S}) \xrightarrow{\text{BIN conversion}} (\text{D} + 1, \text{D})$ BCD (0 to 99999999)
Conversion from floating decimal point to BIN	$\text{INT}(\text{P}) \quad \text{S} \quad \text{D}$	• $(\text{S} + 1, \text{S}) \xrightarrow{\text{BIN conversion}} (\text{D})$ Real number (-32768 to 32767)
	$\text{DINT}(\text{P}) \quad \text{S} \quad \text{D}$	• $(\text{S} + 1, \text{S}) \xrightarrow{\text{BIN conversion}} (\text{D})$ Real number (-214783648 to 2147483647)
BIN to Conversion from floating decimal point	$\text{FLT}(\text{P}) \quad \text{S} \quad \text{D}$	• $(\text{S} + 1, \text{S}) \xrightarrow{\text{Conversion to floating decimal point}} (\text{D})$ Real number (-32768 to 32767)
	$\text{DFLT}(\text{P}) \quad \text{S} \quad \text{D}$	• $(\text{S} + 1, \text{S}) \xrightarrow{\text{Conversion to floating decimal point}} (\text{D} + 1, \text{D})$ Real number (-214783648 to 2147483647)
Conversion between BIN 16-bit and 32-bit	$\text{DBL}(\text{P}) \quad \text{S} \quad \text{D}$	• $(\text{S}) \xrightarrow{\text{Conversion to 32-bit data}} (\text{D} + 1, \text{D})$ BIN (-32768 to 32767)
	$\text{WORD}(\text{P}) \quad \text{S} \quad \text{D}$	• $(\text{S} + 1, \text{S}) \xrightarrow{\text{Conversion to 16-bit data}} (\text{D})$ BIN (-32768 to 32767)

Category	Symbol	Processing Details
Conversion from BIN to gray code		<ul style="list-style-type: none"> • (S) → (D) Conversion to gray code BIN (-32768 to 32767)
		<ul style="list-style-type: none"> • (S + 1, S) → (+ 1, DD) Conversion to gray code BIN (-32768 to 32767)
Conversion from gray code to BIN		<ul style="list-style-type: none"> • (S) → (D) Conversion to gray code Gray code (-32768 to 32767)
		<ul style="list-style-type: none"> • (S + 1, S) → (D + 1, D) Conversion to gray code Gray code (-2147483648 to 2147483647)
Complement to 2		<ul style="list-style-type: none"> • (D) → (D) BIN data
		<ul style="list-style-type: none"> • (D + 1, D) → (D + 1, D) BIN data
		<ul style="list-style-type: none"> • (D + 1, D) → (D + 1, D) Real number data
Block conversions		<ul style="list-style-type: none"> • Batch converts BIN data n points from (S) to BCD data and stores the result from D onward. 
		<ul style="list-style-type: none"> • Batch converts BCD data n points from (S) to BIN data and stores the result from D onward. 

(4) Data transfer instructions

Category	Symbol	Processing Details
16-bit data transfer		<ul style="list-style-type: none"> • (S) → (D)
32-bit data transfer		<ul style="list-style-type: none"> • (S1 + 1, S) → (D + 1, D)
Floating decimal point data transfer		<ul style="list-style-type: none"> • (S1 + 1, S) → (D + 1, D)
Character string data transfer		<ul style="list-style-type: none"> • Transfers character string designated by (S) to device designated by (D) onward.
16-bit data negation transfer		<ul style="list-style-type: none"> • (S) → (D)
32-bit data negation transfer		<ul style="list-style-type: none"> • (S1 + 1, S) → (D + 1, D)

Category	Symbol	Processing Details
Block transfer		
Multiple transfers of same data block		
16-bit data exchange		• (S) \longleftrightarrow (D)
32-bit data exchange		• (S1 + 1, S) \longleftrightarrow (D + 1, D)
Block data exchange		
Exchange of upper and lower bytes		<p>b15 to b8 b7 to b0</p> <p>(S) </p> <p>b15 to b8 b7 to b0</p> <p>(D) </p> <p><i>(Note: Arrows in the original diagram indicate a swap of the two 8-bit fields between S and D.)</i></p>

(5) Program branch instruction

Category	Symbol	Processing Details
Jump		• Jumps to Pn when input conditions are met
		• Jumps to Pn from the scan after the meeting of input condition
		• Jumps unconditionally to Pn
		• Jumps to END instruction when input condition is met

(6) Program execution control instructions

Category	Symbol	Processing Details
Disable interrupts		• Prohibits the running of an interrupt program
Enable interrupts		• Resets interrupt program execution prohibition
Interrupt disable/enable setting		• Prohibits or permits interrupts for each interrupt program
Return		• Returns to sequence program following an interrupt program

(7) I/O refresh instructions

Category	Symbol	Processing Details
I/O Refresh		• Refreshes the relevant I/O area during scan

(8) Other convenient instructions

Category	Symbol	Processing Details
Up/Down Counter		<p>Current Cn value 0 1 2 3 4 5 6 7 6 5 4 3 2 1 0 -1 -2 -3 -2 -1 0</p> <p>Cn contact point</p>
		<p>Current Cn value 0 1 2 3 4 5 4 3 2 1 0 -1</p> <p>Cn contact point</p>
Teaching timer		<ul style="list-style-type: none"> • (Time that TTMR is ON) x n → (D) n = 0 : 1, n = 1 : 10, n = 2 : 100
Special timer		<ul style="list-style-type: none"> • The 4 points from the bit device designated by (D) operate as shown below, depending on the ON/OFF status of the input conditions for the STMR instruction: (D) + 0: Off delay timer output (D) + 1: One shot timer output after off (D) + 2: One shot timer output after on (D) + 3: On delay timer output
Nearest path control		• Rotates a rotary table with n1 divisions from the stop position to the position designated by (S+1) by the nearest path.
Ramp signal		• Changes device data designated by D1 from n1 to n2 in n3 scans.
Pulse density		• Counts pulse input from device designated by D1 for time designated by n, and stores the count at device designated by (D).
Pulse output		• (n1) Hz → Output n2 times → (D)
Pulse width modulation		<p>(D)</p>
Matrix input		• Successively inputs data of 16 points times n strings starting from the device designated by S1 and stores it from the device designated by D2 onward.

3.3.3 Application instructions

(1) Logical operation instructions

Category	Symbol	Processing Details
Logical product		• $(D) \wedge (S) \rightarrow (D)$
		• $(S1) \wedge (S2) \rightarrow (D)$
		• $(D + 1, D) \wedge (S + 1, S) \rightarrow (D + 1, D)$
		• $(S1 + 1, S1) \wedge (S2 + 1, S2) \rightarrow (D + 1, D)$
Logical sum		• $(D) \vee (S) \rightarrow (D)$
		• $(S1) \vee (S2) \rightarrow (D)$
		• $(D + 1, D) \vee (S + 1, S) \rightarrow (D + 1, D)$
		• $(S1 + 1, S1) \vee (S2 + 1, S2) \rightarrow (D + 1, D)$
Exclusive OR		• $(D) \nabla (S) \rightarrow (D)$
		• $(S1) \nabla (S2) \rightarrow (D)$
		• $(D + 1, D) \nabla (S + 1, S) \rightarrow (D + 1, D)$
		• $(S1 + 1, S1) \nabla (S2 + 1, S2) \rightarrow (D + 1, D)$
NON exclusive logical sum		• $(D) \nabla (S) \rightarrow (D)$
		• $(S1) \nabla (S2) \rightarrow (D)$
		• $(D + 1, D) \nabla (S + 1, S) \rightarrow (D + 1, D)$
		• $(S1 + 1, S1) \nabla (S2 + 1, S2) \rightarrow (D + 1, D)$

(2) Rotation instructions

Category	Symbol	Processing Details
Right rotation		<p>b15 (D) b0 SM700</p> <p>Rotates n bits to the right</p>
		<p>b15 (D) b0 SM700</p> <p>Rotates n bits to the right</p>
Left rotation		<p>SM700 b15 (D) b0</p> <p>Rotates n bits to the left</p>
		<p>SM700 b15 (D) b0</p> <p>Rotates n bits to the left</p>
Right rotation		<p>(D+1) (D)</p> <p>b31 to b16 b15 to b0 SM700</p> <p>Rotates n bits to the right</p>
		<p>(D+1) (D)</p> <p>b31 to b16 b15 to b0 SM700</p> <p>Rotates n bits to the right</p>
Left rotation		<p>(D+1) (D)</p> <p>SM700 b31 to b16 b15 to b0</p> <p>Rotates n bits to the left</p>
		<p>SM700 b31 to b16 b15 to b0</p> <p>Rotates n bits to the left</p>

(3) Shift Instructions

Category	Symbol	Processing Details
n-bit shift		<p>b15 bn b0</p> <p>b15 b0 SM700</p> <p>0 to 0</p>
		<p>b15 bn b0</p> <p>SM700 b15 b0</p> <p>0 to 0</p>

Category	Symbol	Processing Details
1-bit shift		
1-word shift		

(4) Bit processing instructions

Category	Symbol	Processing Details
Bit set and reset		
Bit tests		
Batch reset of bit devices Reset		

(5) Data processing instructions

Category	Symbol	Processing Details
Data Searches		
Bit checks		
Decode		
Encode		
7-segment decode		
Separating and Linking		<ul style="list-style-type: none"> Separates 16-bit data designated by (S) into 4-bit units, and stores at the lower 4 bits n points from (D). ($n \leq 4$)
		<ul style="list-style-type: none"> Links the lower 4 bits n points from the device designated by (S) and stores at the device designated by (D). ($n \leq 4$)
		<ul style="list-style-type: none"> Separates the data at the devices below that designated by (S) into bits designated below (S2) and stores in sequence from the device designated by (D).
		<ul style="list-style-type: none"> Links the data at the devices below that designated by (S) in the bits designated below (S2) and stores in sequence from the device designated by (D).
		<ul style="list-style-type: none"> Breaks n-points of 16-bit data from the device designated by (S) into 8-bit units, and stores in sequence at the device designated by (D).
		<ul style="list-style-type: none"> Links the lower 8 bits of n-points of 16-bit data from the device designated by (S) into 16-bit units, and stores in sequence at the device designated by (D).

Category	Symbol	Processing Details
Search		<ul style="list-style-type: none"> Searches the data n-points from the device designated by (S) in 16-bit units, and stores the maximum value at the device designated by (D).
		<ul style="list-style-type: none"> Searches the data n-points from the device designated by (S) in 16-bit units, and stores the minimum value at the device designated by (D).
		<ul style="list-style-type: none"> Searches the data 2 x n-points from the device designated by (S) in 32-bit units, and stores the maximum value at the device designated by (D).
		<ul style="list-style-type: none"> Searches the data 2 x n-points from the device designated by (S) in 32-bit units, and stores the minimum value at the device designated by (D).
Sort	<ul style="list-style-type: none"> S2: Number of comparisons made during one run D1: Device to turn ON when sort is completed D2: For system use 	<ul style="list-style-type: none"> Sorts data n-points from device designated by (S1) in 16-bit units. [[n x (n - 1)] / 2 scans required]
	<ul style="list-style-type: none"> S2: Number of comparisons made during one run D1: Device to turn ON when sort is completed D2: For system use 	<ul style="list-style-type: none"> Sorts data 2 x n-points from device designated by (S1) in 32-bit units. [[n x (n - 1)] / 2 scans required]

(6) Structure creation instructions

Category	Symbol	Processing Details
No. of repeats		<ul style="list-style-type: none"> Executes n times between [FOR] and [NEXT].
		<ul style="list-style-type: none"> Forcibly ends the execution of the [FOR] to [NEXT] cycle and jumps pointer to Pn.
Sub-routine program calls		<ul style="list-style-type: none"> Executes sub-routine program Pn when input condition is met. (S1 to Sn are arguments sent to sub-routine program. 0 ≤ n ≤ 5)
		<ul style="list-style-type: none"> Returns from sub-routine program
		<ul style="list-style-type: none"> Performs non-execution processing on sub-routine program Pn if input conditions have not been met.
	 * : Program name	<ul style="list-style-type: none"> Executes sub-routine program Pn from within designated program name when input condition is met. (S1 to Sn are arguments sent to sub-routine program. 0 ≤ n ≤ 5)
	 * : Program name	<ul style="list-style-type: none"> Performs non-execution processing of sub-routine program Pn from within designated program name if input condition is not met.
		<ul style="list-style-type: none"> Performs link refresh and general data processing.
Fixed index qualification	 	<ul style="list-style-type: none"> Conducts index qualification for individual devices used in device qualification ladder.
Fixed index qualification	 Designates qualification value	<ul style="list-style-type: none"> Stores qualification value used for index qualification performed between [IX] and [IXEND] in the device below that designated by D.

(7) Table operation instructions

Category	Symbol	Processing Details
Table Processing		

(8) Buffer memory access instructions

Category	Symbol	Processing Details
Data read		• Reads data in 16-bit units from special function module
		• Reads data in 32-bit units from special function module
Data write		• Writes data in 16-bit units to special function module
		• Writes data in 32-bit units to special function module

(9) Display instructions

Category	Symbol	Processing Details
ASCII print	* When SM701 OFF 	• Outputs ASCII code 8 points (16 characters) from device designated by (S) to output module.
	* When SM701 ON 	• Outputs ASCII code from device designated by (S) to 00H to output module.

Category	Symbol	Processing Details
ASCII print		<ul style="list-style-type: none"> Converts comments from device designated by (S) to ASCII code and outputs to output module.
Display		<ul style="list-style-type: none"> Displays ASCII code 8 points (16 characters) from the device designated by (S) at the LED display device on the front of the CPU
		<ul style="list-style-type: none"> Displays the comments from the device designated by (S) at the LED display device on the front of the CPU.
Reset		<ul style="list-style-type: none"> Resets annunciator and display unit display.

(10) Debugging and failure diagnosis instructions

Category	Symbol	Processing Details
Failure checks		<ul style="list-style-type: none"> CHK instruction is executed when CHKST is executable. Jumps to the step following the CHK instruction when CHKST is in a non-executable state.
		<ul style="list-style-type: none"> During normal conditions → SM80: OFF, SD80: 0 During abnormal conditions → SM80: ON, SD80: Failure No.
		<ul style="list-style-type: none"> Starts update in ladder pattern being checked by CHK instruction
		<ul style="list-style-type: none"> Ends update in ladder pattern being checked by CHK instruction
Status latch		<ul style="list-style-type: none"> Executes status latch
		<ul style="list-style-type: none"> Resets status latch to enable re-execution
Sampling trace		<ul style="list-style-type: none"> Applies trigger to sampling trace
		<ul style="list-style-type: none"> Resets sampling trace to enable re-execution
Program trace		<ul style="list-style-type: none"> Applies trigger to program trace
		<ul style="list-style-type: none"> Resets program trace to enable re-execution
		<ul style="list-style-type: none"> Executes program trace

(11) Character string processing instructions

Category	Symbol	Processing Details
From BIN to decimal ASCII		<ul style="list-style-type: none"> Converts 1-word BIN value designated by (S) to a 5-digit, decimal ASCII value, and stores it at the word device designated by (D).
		<ul style="list-style-type: none"> Converts 2-word BIN value designated by (S) to a 10-digit, decimal ASCII value, and stores it at word devices following the word device number designated by (D).
From BIN to hexadecimal ASCII		<ul style="list-style-type: none"> Converts 1-word BIN value designated by (S) to a 4-digit, hexa decimal ASCII value, and stores it at a word device following the word device designated by (D).
		<ul style="list-style-type: none"> Converts 2-word BIN value designated by (S) to an 8-digit, hexadecimal ASCII value, and stores it at word devices following the word device designated by (D).

Category	Symbol	Processing Details
From BCD to ASCII		• Converts 1-word BCD value designated by (S) to a 4-digit, decimal ASCII value, and stores it at a word device following the word device designated by (D).
		• Converts 2-word BCD value designated by (S) to an 8-digit, decimal ASCII value, and stores it at word devices following the word device number designated by (D).
From decimal ASCII to BIN		• Converts a 5-digit, decimal ASCII value designated by (S) to a 1-word BIN value, and stores it at a word device number designated by (D).
		• Converts a 10-digit, decimal ASCII value designated by (S) to a 2-word BIN value, and stores it at a word device number designated by (D).
From hexadecimal ASCII to BIN		• Converts a 4-digit, hexadecimal ASCII value designated by (S) to a 1-word BIN value, and stores it at a word device number designated by (D).
		• Converts an 8-digit, hexadecimal ASCII value designated by (S) to a 2-word BIN value, and stores it at a word device number designated by (D).
From ASCII to BCD		• Converts a 4-digit, decimal ASCII value designated by (S) to a 1-word BCD value, and stores it at a word device number designated by (D).
		• Converts an 8-digit, decimal ASCII value designated by (S) to a 2-word BCD value, and stores it at a word device number designated by (D).
Device comment read operation		• Stores comment from device designated by (S) at a device designated by (D).
Character string length detection		• Stores data length (number of characters) in character string designated by (S) at a device designated by (D).
BIN to decimal character string		• Converts a 2-word BIN value designated by S1 to a decimal character string with the total number of digits and the number of decimal fraction digits designated by S1 and stores this at a device designated by (D).
		• Converts a 2-word BIN value designated by S2 to a decimal character string with the total number of digits and the number of decimal fraction digits designated by S1 and stores this at a device designated by (D).
Decimal character string to BIN		• Converts a character string including decimal point designated by (S) to a 1-word BIN value and the number of decimal fraction digits, and stores at devices designated by D1 and D2.
		• Converts a character string including decimal point designated by (S) to a 2-word BIN value and the number of decimal fraction digits, and stores at devices designated by D1 and D2.
Floating decimal point to character string		• Converts floating decimal point data designated by (S) to character string, and stores in a device designated by (D).
Character string to floating decimal point		• Converts character string designated by (S) to floating decimal point data, and stores in a device designated by (D).
Hexadecimal BIN to ASCII		• Converts 1-word BIN value following device number designated by (S) to hexadecimal ASCII, and stores designated number of characters following a word device number designated by (D).
ASCII to hexadecimal BIN		• Converts just the number of characters designated by n from hexadecimal ASCII data following word device designated by (S) to BIN value, and stores following device number designated by (D).

Category	Symbol	Processing Details
Character string processing	$\text{RIGHT}(P) \quad S \quad D \quad n$	• Stores n number of characters from the end of a character string designated by (S) at a device designated by (D).
	$\text{LEFT}(P) \quad S \quad D \quad n$	• Stores n number of characters from the beginning of a character string designated by (S) at a device designated by (D).
	$\text{MIDR}(P) \quad S1 \quad D \quad S2$	• Stores a designated number of characters from the position designated by (S2) of a character string designated by (S1) to a device designated by (D).
	$\text{MIDW}(P) \quad S1 \quad D \quad S2$	• Stores a character string designated by (S1) a designated number of characters from the position designated by (S2) of a device designated by (D).
	$\text{INSTR}(P) \quad S1 \quad S2 \quad D \quad n$	• Searches character string (S1) from the nth character of character string (S2), and stores matched positions at (D).
Floating decimal point to BCD	$\text{EMOD}(P) \quad S1 \quad S2 \quad D$	• Converts floating decimal point data (S1) to BCD data with number of decimal fraction digits designated by (S2), and stores at device designated by (D).
BCD to floating decimal point data	$\text{EREXP}(P) \quad S1 \quad S2 \quad D$	• Converts BCD data (S1) to floating decimal point data with the number of decimal fraction digits designated by (S2), and stores at device designated by (D).

(12) Special function instructions

Category	Symbol	Processing Details
Trigonometric functions (Floating decimal point data)	$\text{SIN}(P) \quad S \quad D$	• $\text{Sin}(S + 1, S) \rightarrow (D + 1, D)$
	$\text{COS}(P) \quad S \quad D$	• $\text{Cos}(S + 1, S) \rightarrow (D + 1, D)$
	$\text{TAN}(P) \quad S \quad D$	• $\text{Tan}(S + 1, S) \rightarrow (D + 1, D)$
	$\text{ASIN}(P) \quad S \quad D$	• $\text{Sin}^{-1}(S + 1, S) \rightarrow (D + 1, D)$
	$\text{ACOS}(P) \quad S \quad D$	• $\text{Cos}^{-1}(S + 1, S) \rightarrow (D + 1, D)$
	$\text{ATAN}(P) \quad S \quad D$	• $\text{Tan}^{-1}(S + 1, S) \rightarrow (D + 1, D)$
Conversion between angles and radians	$\text{RAD}(P) \quad S \quad D$	• $(S + 1, S) \rightarrow (D + 1, D)$ Conversion from angles to radians
	$\text{DEG}(P) \quad S \quad D$	• $(S + 1, S) \rightarrow (D + 1, D)$ Conversion from radians to angles
	$\text{SQR}(P) \quad S \quad D$	• $\sqrt{(S + 1, S)} \rightarrow (D + 1, D)$
Exponent operations	$\text{EXP}(P) \quad S \quad D$	• $e^{(S + 1, S)} \rightarrow (D + 1, D)$
Natural logarithms	$\text{LOG}(P) \quad S \quad D$	• $\text{Log } e(S + 1, S) \rightarrow (D + 1, D)$
Square root	$\text{BSQR}(P) \quad S \quad D$	• $\sqrt{(S)} \rightarrow (D) + 0$ [Integer part] + 1 [Decimal fraction part]
	$\text{BDSQR}(P) \quad S \quad D$	• $\sqrt{(S + 1, S)} \rightarrow (D) + 0$ [Integer part] + 1 [Decimal fraction part]
Trigonometric function	$\text{BSIN}(P) \quad S \quad D$	• $\text{Sin}(S) \rightarrow (D) + 0$ [Sign] + 1 [Integer part] + 2 [Decimal fraction part]
	$\text{BCOS}(P) \quad S \quad D$	• $\text{Con}(S) \rightarrow (D) + 0$ [Sign] + 1 [Integer part] + 2 [Decimal fraction part]

Category	Symbol	Processing Details
Trigonometric function		<ul style="list-style-type: none"> • $\tan(S)$ → (D) + 0 [Sign] + 1 [Integer part] + 2 [Decimal fraction part]
		<ul style="list-style-type: none"> • $\sin^{-1}(S)$ → (D) + 0 [Sign] + 1 [Integer part] + 2 [Decimal fraction part]
		<ul style="list-style-type: none"> • $\cos^{-1}(S)$ → (D) + 0 [Sign] + 1 [Integer part] + 2 [Decimal fraction part]
		<ul style="list-style-type: none"> • $\tan^{-1}(S)$ → (D) + 0 [Sign] + 1 [Integer part] + 2 [Decimal fraction part]

(13) Data control instructions

Category	Symbol	Processing Details
Upper and lower limit controls		<ul style="list-style-type: none"> • Process the value designated in S3 according to the upper/lower limits set in S1 and S2 so that the value fits in a certain range, and stores the processed value in the word device number designated in D. <ul style="list-style-type: none"> • When $S3 < S1$ Store value of S1 at (D) • When $S1 \leq S3 \leq S2$. Store value of S3 at (D) • When $S2 < S3$ Store value of S2 at (D)
		<ul style="list-style-type: none"> • Process the value designated in (S3 + 1, S3) according to the upper/lower limits set in (S1 + 1, S1) and (S2 + 1, S2) so that the value fits in a certain range, and stores the processed value in the word device number designated in (D + 1, D). <ul style="list-style-type: none"> • When $(S3 + 1, S3) < (S1 + 1, S1)$ Store value of (S1 + 1, S1) at (D + 1, D) • When $(S1 + 1, S1) \leq (S3 + 1, S3) \leq (S2 + 1, S2)$ Store value of (S3 + 1, S3) at (D + 1, D) • When $(S2, S2 + 1) < (S3, S3 + 1)$ Store value of (S2 + 1, S2) at (D + 1, D)
Dead band controls		<ul style="list-style-type: none"> • When the value designated in S3 is within the dead band designated in S1 and S2, "0" is stored in the word device number designated in D. When the value is outside the dead band, input value - dead band upper/lower limit is calculated and stored in the word device number. <ul style="list-style-type: none"> • When $S1 \leq S3 \leq S2$. 0 → D • When $S3 < S1$ $S3 - S1$ → D • When $S3 > S2$ $S3 - S2$ → D
		<ul style="list-style-type: none"> • When the value designated in (S3 + 1, S3) is within the dead band designated in (S1 + 1, S1) and (S2 + 1, S2), "0" is stored in the word device number designated in D. When the value is outside the dead band, input value - dead band upper/lower limit is calculated and stored in the word device number. <ul style="list-style-type: none"> • When $(S1 + 1, S1) \leq (S3 + 1, S3) \leq (S2 + 1, S2)$ 0 → (D + 1, D) • When $(S3 + 1, S3) < (S1 + 1, S1)$ $(S3 + 1, S3) - (S1 + 1, S1)$ → (D + 1, D) • When $(S3 + 1, S3) > (S2 + 1, S2)$ $(S3 + 1, S3) - (S2 + 1, S2)$ → (D + 1, D)

Category	Symbol	Processing Details
Zone controls	$\overline{\text{ZONE(P)}} \quad \boxed{\text{S1}} \quad \boxed{\text{S2}} \quad \boxed{\text{S3}} \quad \boxed{\text{D}}$	<ul style="list-style-type: none"> • By setting positive and negative bias values in S1 and S2 for the value designated in S3, S1 + bias value is calculated and stored in the word device designated in D. <ul style="list-style-type: none"> • When S3 = 0 .. 0 → D • When S3 > 0 .. S3 + S2 → D • When S3 < 0 .. S3 - S1 → D
	$\overline{\text{DZONE(P)}} \quad \boxed{\text{S1}} \quad \boxed{\text{S2}} \quad \boxed{\text{S3}} \quad \boxed{\text{D}}$	<ul style="list-style-type: none"> • By setting positive and negative bias values in (S1 + 1, S1) and (S2 + 1, S2) to the value designated in (S3 + 1, S3), S1 + bias value is calculated and stored in the word device designated in (D + 1, D). <ul style="list-style-type: none"> • When (S3 + 1, S3) = 0 .. 0 → (D + 1, D) • When (S3 + 1, S3) > 0 .. (S3 + 1, S3) - (S2 + 1, S2) → (D + 1, D) • When (S3 + 1, S3) < 0 .. (S3 + 1, S3) + (S1 + 1, S1) → (D + 1, D)

(14) Switching Instructions

Category	Symbol	Processing Details
Block No. Designations	$\overline{\text{RSET(P)}} \quad \boxed{\text{S}}$	• Converts extension file register block number to number designated by (S).
	$\overline{\text{QDRSET(P)}} \quad \boxed{\text{File name}}$	• Sets file names used as file registers.
	$\overline{\text{QCDSET(P)}} \quad \boxed{\text{File name}}$	• Sets file names used as comment files.

(15) Clock instructions

Category	Symbol	Processing Details							
Read/write clock data	$\overline{\text{DATERD(P)}} \quad \boxed{\text{D}}$	(Clock device) → (D) + 0 <table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td>Year</td></tr> <tr><td>Month</td></tr> <tr><td>Day</td></tr> <tr><td>Hour</td></tr> <tr><td>Minute</td></tr> <tr><td>Second</td></tr> <tr><td>Day of week</td></tr> </table>	Year	Month	Day	Hour	Minute	Second	Day of week
	Year								
Month									
Day									
Hour									
Minute									
Second									
Day of week									
	$\overline{\text{DATEWR(P)}} \quad \boxed{\text{S}}$	(D) + 0 <table border="1" style="display: inline-table; vertical-align: middle;"> <tr><td>Year</td></tr> <tr><td>Month</td></tr> <tr><td>Day</td></tr> <tr><td>Hour</td></tr> <tr><td>Minute</td></tr> <tr><td>Second</td></tr> <tr><td>Day of week</td></tr> </table> → (Clock device)	Year	Month	Day	Hour	Minute	Second	Day of week
Year									
Month									
Day									
Hour									
Minute									
Second									
Day of week									

Category	Symbol	Processing Details															
Read/write clock data		<p>(S1) (S2) (D)</p> <table border="1"> <tr><td>Hour</td><td>+</td><td>Hour</td><td>→</td><td>Hour</td></tr> <tr><td>Minute</td><td></td><td>Minute</td><td></td><td>Minute</td></tr> <tr><td>Second</td><td></td><td>Second</td><td></td><td>Second</td></tr> </table>	Hour	+	Hour	→	Hour	Minute		Minute		Minute	Second		Second		Second
	Hour	+	Hour	→	Hour												
	Minute		Minute		Minute												
	Second		Second		Second												
	<p>(S1) (S2) (D)</p> <table border="1"> <tr><td>Hour</td><td>-</td><td>Hour</td><td>→</td><td>Hour</td></tr> <tr><td>Minute</td><td></td><td>Minute</td><td></td><td>Minute</td></tr> <tr><td>Second</td><td></td><td>Second</td><td></td><td>Second</td></tr> </table>	Hour	-	Hour	→	Hour	Minute		Minute		Minute	Second		Second		Second	
Hour	-	Hour	→	Hour													
Minute		Minute		Minute													
Second		Second		Second													
	<p>(S) (D)</p> <table border="1"> <tr><td>Hour</td><td>→</td><td>Second (lower bits)</td></tr> <tr><td>Minute</td><td></td><td>Second (upper bits)</td></tr> <tr><td>Second</td><td></td><td></td></tr> </table>	Hour	→	Second (lower bits)	Minute		Second (upper bits)	Second									
Hour	→	Second (lower bits)															
Minute		Second (upper bits)															
Second																	
	<p>(S) (D)</p> <table border="1"> <tr><td>Second (lower bits)</td><td>→</td><td>Hour</td></tr> <tr><td>Second (upper bits)</td><td></td><td>Minute</td></tr> <tr><td></td><td></td><td>Second</td></tr> </table>	Second (lower bits)	→	Hour	Second (upper bits)		Minute			Second							
Second (lower bits)	→	Hour															
Second (upper bits)		Minute															
		Second															

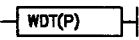
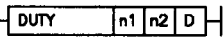
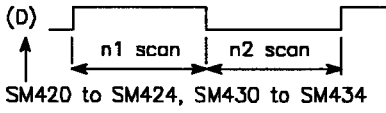
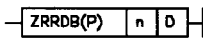
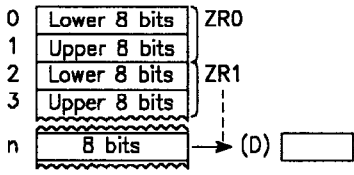
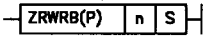
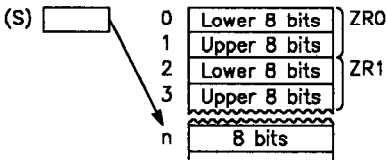
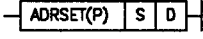
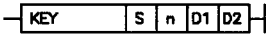
(16) Peripheral device instructions

Category	Symbol	Processing Details
Input/Output to peripheral devices		<ul style="list-style-type: none"> Stores message designated by (S) at QnACPU. This message is displayed at the peripheral device
		<ul style="list-style-type: none"> Data input from the peripheral device is stored at device designated by (D).

(17) Program instructions

Category	Symbol	Processing Details
Program execution status switching		<ul style="list-style-type: none"> Places designated program in standby status
		<ul style="list-style-type: none"> Turns OUT instruction coil of designated program OFF, and places program in standby status.
		<ul style="list-style-type: none"> Registers designated program as scan execution program.
		<ul style="list-style-type: none"> Registers designated program as low-speed execution program.

(18) Other

Category	Symbol	Processing Details
WDT reset		<ul style="list-style-type: none"> Resets watchdog timer during sequence program
Timing clock		 <p>(D) SM420 to SM424, SM430 to SM434</p>
Direct read/write operations in 1-byte units		
Direct read/write operations in 1-byte units		
Indirect address designation		<p>(S) → (D)</p> <p>↑ Indirect address of designated device</p> <p>Device name</p>
Numerical key input from keyboard		<ul style="list-style-type: none"> Takes in ASCII data for 8 points of input unit designated by (S), converts to hexadecimal value following device number designated by D1, and stores.

3.3.4 Data link instructions

(1) Link refresh instructions

Category	Symbol	Processing Details
Designated network refresh	$\overline{J(P).ZCOM} \quad J_n$	• Link refreshes the network module with the designated network number in network n.
	$\overline{G(P).ZCOM} \quad U_n$	• Link refreshes the network module with the designated I/O number in network n.

(2) QnA link dedicated instructions

Category	Symbol	Processing Details
Read/write other station data	$\overline{JP.READ} \quad J_n \quad S_1 \quad S_2 \quad D_1 \quad D_2$	• Reads data from other station word devices.
	$\overline{GP.READ} \quad U_n \quad S_1 \quad S_2 \quad D_1 \quad D_2$	
	$\overline{JP.WRITE} \quad J_n \quad S_1 \quad S_2 \quad D_1 \quad D_2$	• Writes data to other station word devices.
	$\overline{GP.WRITE} \quad U_n \quad S_1 \quad S_2 \quad D_1 \quad D_2$	
Data communications with other station	$\overline{JP.SEND} \quad J_n \quad S_1 \quad S_2 \quad D$	• Transmits data (messages) to other station.
	$\overline{GP.SEND} \quad U_n \quad S_1 \quad S_2 \quad D$	
	$\overline{JP.RECV} \quad J_n \quad S \quad D_1 \quad D_2$	• Reads data (messages) received from other station.
	$\overline{GP.RECV} \quad U_n \quad S \quad D_1 \quad D_2$	
Processing request to other station	$\overline{JP.REQ} \quad J_n \quad S_1 \quad S_2 \quad D_1 \quad D_2$	• Conducts other station remote RUN/STOP.
	$\overline{GP.REQ} \quad U_n \quad S_1 \quad S_2 \quad D_1 \quad D_2$	
Read/write data from remote I/O station special function module	$\overline{J(P).ZNFR} \quad J_n \quad S_1 \quad S_2 \quad D$	• Reads data from special function module installed in MELSECNET/10 remote I/O station.
	$\overline{G(P).ZNFR} \quad U_n \quad S_1 \quad S_2 \quad D$	
	$\overline{J(P).ZNTD} \quad J_n \quad S_1 \quad S_2 \quad D$	• Writes data of special function module installed in MELSECNET/10 remote I/O station.
	$\overline{G(P).ZNTD} \quad U_n \quad S_1 \quad S_2 \quad D$	

(GP.*** instructions can also be used in AJ71QC24)

(3) A-series compatible link instructions

Category	Symbol	Processing Details
Read designated station word device	$\overline{J(P).ZNRD} \quad J_n \quad n_1 \quad S \quad D_1 \quad n_2 \quad D_2$	• Reads other station T, C, D, W data in MELSECNET(II), MELSECNET/10 systems.
Write word device to designated station	$\overline{J(P).ZNWR} \quad J_n \quad n_1 \quad D_1 \quad S \quad n_2 \quad D_2$	• Writes data to other station T, C, D, W on MELSECNET(II), MELSECNET/10.

Category	Symbol	Processing Details
Read/write data from remote I/O station special function module	$\overline{G(P).RFRP} \quad U_n \quad n1 \quad D1 \quad n2 \quad D2$	• Reads data from special function module installed in MELSECNET(II) remote I/O station.
	$\overline{G(P).RTOP} \quad U_n \quad n1 \quad S \quad n2 \quad D$	• Writes data to special function module installed in MELSECNET(II) remote I/O station.

(4) Routing parameter instructions

Category	Symbol	Processing Details
Read routing information	$\overline{Z.RTREAD} \quad n \quad D$	• Reads data of destination network No. designated at n by routing parameters, and stores from D onwards.
Register routing information	$\overline{Z.RTWRITE} \quad n \quad S$	• Registers routing data from S onwards to the destination network No. area designated at the routing parameter n.

3.3.5 PID control instructions

Category	Symbol	Processing Details
Set PID control data	$\overline{PIDINIT} \quad S$	• Registers the PID control data set from the device number designated at S onwards to the PC CPU.
Execute PID control	$\overline{PIDCONT} \quad S$	• Conducts PID operations based on the set value (SV) and process value (PV) set from the device number designated at S onwards, and stores the operation results to the automatic manipulated value (MV) area.
Monitor PID control status	$\overline{PID57} \quad n \quad S1 \quad S2$	• Displays by bar graph the PID control status of the loop No. designated at S1 on the AD57 indicator designated at n. When the PID control monitor function is initially being executed, static screen components other than bar graphs and numeric data are displayed by issuing the initial screen display request designated by S2.
Stop designated loop operation	$\overline{PIDSTOP} \quad n$	• Stops the operation of the loop No. designated at n.
Start designated loop operation	$\overline{PIDRUN} \quad n$	• Starts the operation of loop No. designated at n.
Change designated loop parameter	$\overline{PIDPRMW} \quad n \quad S$	• Changes the operation parameters of the loop No. designated at n, to data set from the device number designated at S onwards.

3.3.6 Special function module instructions

Category	Function	Instruction Symbol
AD61(S1) control instructions	Sets preset data	RVWR1, RVWR2
	Sets larger/smaller/equal judgement set value data	SVWR1, SVWR2
	Reads present value	PVRD1, PVRD2
AD59(S1) control instructions	Prints any number of characters	PRN
	Prints characters up to 00H code	PR
	Reads data from memory card	GET
	Writes data to memory card	PUT
AJ71C24(-S3/S6/S8) control instructions	Sends designated number of bytes of data by no-protocol mode	PRN
	Sends data up to 00H by no-protocol mode	PR
	Receives data by no-protocol mode	INPUT
	Reads communication status	SPBUSY
	Forced interruption of communication processing	SPCLR
AJ71C21(S1) control instructions	Send designated number of bytes of data	PRN2, PRN4
	Send data up to 00H code	PR2, PR4
	Receive data	INPUT2, INPUT4
	Read RAM memory	GET
	Write to RAM memory	PUT
	Forced interruption of communication processing	SPBUSY
AJ71PT32-S3 control instructions	Key operation from operation box	INPUT
	Send designated number of bytes of data by no-protocol mode	PRN
	Send data up to 00H code by no-protocol mode	PR
	Receive data by no-protocol mode	INPUT
	Communication with remote terminal module	MINI, MINIEND
	Error reset for remote terminal module	MINIERR
	Read communication status	SPBUSY
	Forced interruption of communication processing	SPCLR
AD57 control instructions	Set display mode	CMODE
	Display canvas screen	CPS1
	Change VRAM display address	CPS2
	Transfer canvas to VRAM area	CMOV

Category	Function	Instruction Symbol
AD57 control instructions	Clear display screen	CLS
	Clear VRAM area	CLV
	Screen scroll	CSCRU, CSCRD
	Cursor display	CON1, CON2
	Cursor clear	COFF
	Set cursor position	LOCATE
	Designate character normal/highlight	CNOR, CREV
	Switch between character normal/highlight	CRDSP, SRDSPV
	Designate character display color	COLOR
	Change character color	CCDSP, CCDSPV
	Display ASCII character	PR, PRN
	Write ASCII character to VRAM	PRV, PRNV
	Display character	EPR, EPRN
	Write character to VRAM	EPRV, EPRNV
	Link display same characters	CR1, CR2, CC1, CC2
	- (minus) display	CINMP
	- (hyphen) display	CINHP
	. (period, decimal point) display	CINPT
	Numeral display	CIN0 to CIN9
	Alphabet letter display	CINA to CINZ
	Space display	CINSP
	Clear display of designated area	CINCLR
	ASCII code conversion of display text	INPUT
	Read VRAM data	GET
	Write VRAM data	PUT
	Read display status	STAT
AJ71ID[]-R4 instructions	ID controller initial setting	IDINIT1, IDINIT2
	Read from ID data carrier	IDRD1, IDRD2
	Write to ID data carrier	IDWD1, IDWD2
	Continuous reading from ID data carrier	IDARD1, IDARD2
	Continuous writing to ID data carrier	IDAWD1, IDAWD2
	Data comparison with ID data carrier	IDCMP1, IDCMP2
	Batch writing of same data to ID data carrier	IDFILL1, IDFILL2

Category	Function	Instruction Symbol
AJ71ID[]-R4 instructions	Copying between ID data carriers	IDCOPY1, IDCOPY2
	Clear ID data carrier	IDCLR1, IDCLR2
	ID data carrier use end	IDOFF1, IDOFF2
	ID data carrier use start	IDON1, IDON2
AJ71QC24 instructions*	Write user registration frame to AJ71QC24 EEPROM	PUTE
	Read user registration frame from AJ71QC24 EEPROM	GETE
	Data transmission by on-demand functions in dedicated protocol	ONDEMAND
	Send designated number of bytes of data by no-protocol mode	OUTPUT
	Send following send schedule table by no-protocol mode	PRR
	Receive data by no-protocol mode	INPUT
	Send data by bidirectional protocol	BIDOUT
	Receive data by bidirectional protocol	BIDIN
	Read communication status	SPBUSY
	Read device from other station	READ
	Write device from other station	SWRITE
	Send data to other station	SEND
	Receive data from other station	RECV
	Other station transient request	REQ

* The AJ71QC24 can use the QnA link instructions in the special function module designation (G(P).***).

3.4 Functions

The main QnACPU module functions are shown in the following table.

Function	Content	Reference
Constant scan	A function that executes the sequence program at a specified time interval, regardless of the sequence program processing time.	—
Latch (Memory back up)	A function that holds the contents of the device if the power supply is turned OFF, if the module is reset, or if there is a momentary power interruption of 20 ms or longer.	—
Remote RUN/STOP/RESET	Remote RUN/STOP can be conducted from an external switch or peripheral device. Furthermore, QnACPU reset and latch clear operations can also be conducted from a peripheral device.	—
PAUSE	A function that stops operation while holding the output (Y) statuses as they are.	—
Status latch	A function that stores the content of all devices in the memory card should an error occur, and allows monitoring of that data at a peripheral device.	Section 4.3.3 (4)
Sampling trace	A device that stores the data of the designated device to the memory card at designated intervals, and monitors that data by a peripheral device in order to confirm the changed status of the device.	Section 4.3.3 (3)
Step operation	A function that can stop the execution of a sequence program at a designated step.	—
Skip operation	A function that can jump one part of a sequence program, not executing that step operation.	—
Clock	Reads/writes the data from the clock built into the QnACPU. The clock data consists of the year, month, day, hour, minute, second, and day of the week.	—
I/O module replacement during online	A function for replacing I/O modules while the QnACPU is executing operations.	—
Interrupt processing	A function that, when an interruption occurs, executes a program in response an interruption cause.	—
Diagnosis function	Diagnoses the presence or absence of errors using the self-diagnosis function, and conducts fault detection and QnACPU stop. Also allows batch control of user program error checks as a diagnosis function.	Section 3.4.3
ERROR LED display priority	Can set the on/off for the ERROR LED for when errors occur.	—
File management	Manages parameters, sequence programs, device comments, and file registers, etc., as files.	Section 3.4.2
Program structuring	Allows selection of program execution type in accordance with the purpose of the program. Also, each program can be divided into separate parts for different designers, production processes, etc.	Section 3.4.1
I/O allocation	Allows unrestricted I/O allocation in units of modules, regardless of the module installation position.	Section 3.4.4
Boot operation	A function that runs the sequence program stored in the memory card when the QnACPU is in RUN status, after reading the QnACPU module internal memory.	Section 3.4.5
Fault history	A function storing the latest 16 points of errors detected by the diagnosis function in the QnACPU. However, if the memory card is designated as the storage destination, then a maximum of 100 points can be stored.	—

Function	Content	Reference
Data protection function	A function prohibiting the reading/writing from peripheral devices of programs or comments in the QnACPU module internal memory or memory cards.	Section 3.4.6
Device initial values	A function for setting initial values in device memory, file registers, special function modules, etc., when QnACPU goes from STOP to RUN.	Section 3.4.7

3.4.1 Program structuring

You can select the program execution type to suit the application of your control program. Each program execution type allows structuring of the program by dividing it into multiple files for different designers or production processes.

(1) There are four kinds of program execution types.

Please set the execution type in the parameters.

(a) Initial execution type

- These are programs that are executed only once, when the power supply goes from ON to RUN, or when the CPU goes from STOP to RUN.
- You can reduce the scan time by setting initial data setting programs (with ACPU these would be programs executed on reception of contact M9038 signal) as this initial execution type.

(b) Scan execution type

- These are programs that are executed at all times. They correspond to the usual step 0 to END programs.

(c) Low speed execution type

- These are programs executed only in the waiting time during constant scan operation or in the low speed program execution time.
- This type is selected for programs that can be execute at a low frequency, such as periodic inspection programs. You can reduce the processing time of scan execution programs by using this execution type.
- You can set the time taken for low speed programs to complete execution in the parameters, and monitor this time.

(d) Standby type

- This creates a program such as a subroutine program or an interrupt program which is only executed when start causes occur.
- This execution type is used to create library data, and when you want to execute control separately from the main program. This execution method is the same as for subroutine or interrupt programs in scan execution type or low speed execution type programs.

REMARK

If scan execution type programs are structured as one program, then the same program configuration can be achieved as for MELSEC-A programs.

- (2) A multiple number of programs can be linked together and executed for each of the execution types.
Set the file names to be executed and the execution order in the parameters.

Example: The following example shows the program setting and program execution order.

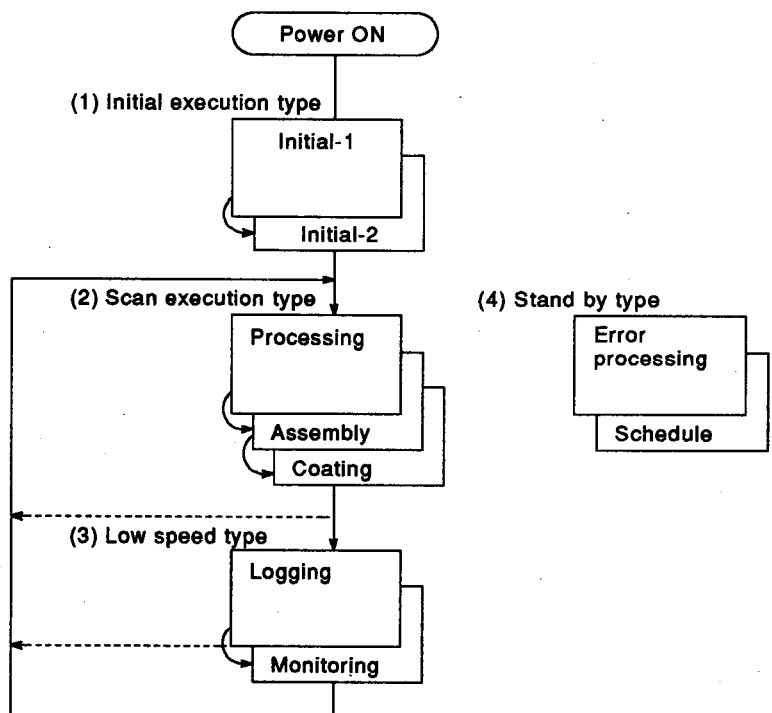
- (a) When the power supply is switched ON, the (1) initial execution type programs are executed in registration order (initial-1 to initial-2).
- (b) After execution of the initial execution type programs, the (2) scan execution type and (3) low speed execution type programs are executed.
 - The scan execution type program is executed every scan in the registered order (process → assembly → coat).
 - The low speed execution type program is executed in only the constant scan excess time or in the set time, in the registered order (monitoring → logging).
 If program execution cannot be completed in one scan it is continued in the next scan.

[Program setting]

#	Program	Execute
1	[INITIAL1]	<Init>
2	[INITIAL2]	<Init>
3	[MACHINE]	<Scan>
4	[ASSEMBLY]	<Scan>
5	[COATING]	<Scan>
6	[ERRPROCS]	<Wait>
7	[SCHEDULE]	<Wait>
8	[MONITOR]	<Slow>
9	[LOGGING]	<Slow>
10	[]	< >
11	[]	< >
12	[]	< >

PgUp:Prev PgDn:Next Esc:Close

[Program execution order]



REMARK

When the scan execution type consists of only one program, then you do not need to set parameter program settings.

3.4.2 File management

You can manage the various data stored in QnACPU internal memory and in memory cards as files.

- (1) There are the following advantages in being able to manage data as files.
 - (a) You can freely store data to memory in units of files, and any subsequent addition to or change of files will not affect other files. Previously, when the program capacity was increased you had to change the parameter settings, and rewrite comments and file registers to the CPU module, in addition to programs and parameters.
 - (b) You can manage programs by attaching names and times to them.
 - (c) You can write protect each file.

(2) File types

(a) Program

- Parameter
- Sequence programs

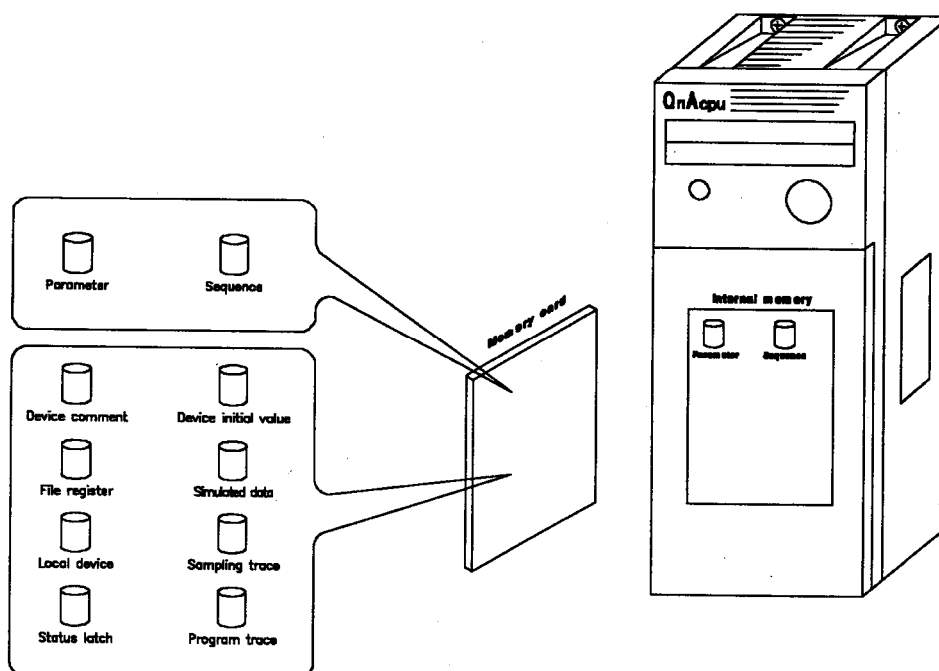
(b) Device

- Device comment
- Device initial value
- File register
- Simulated data
- Local device

(c) Debugging

- Sampling trace
- Status latch
- Program trace

(d) Fault history data



3.4.3 Diagnosis function

The diagnosis function includes self-diagnosis and external fault diagnosis, and these can be handled together for diagnosis purposes. In the diagnostic function, apart from the conventional error codes there is also diagnostic information used for confirming the details of the error content. The file name, time (set value), program error location, parameter No., annunciator (F) No., CHK instructions, and fault No., etc are stored from SD0 to SD20. These error codes can be confirmed at peripheral devices, and read in programs.

(1) Interruption at error occurrence

This function executes an interrupt program whenever an error occurs. When you are using this function, make the parameter setting for operation to continue when the relevant error occurs. The applicable errors are shown in the following table.

Interrupt Pointer	Corresponding Error Message
I32	Stop error related
I33	AC DOWN
I34	UNIT VERIFY ERR. FUZE BRAKE OFF SP. UNIT ERROR
I35	OPERATION ERROR SFCP OPE. ERROR SFCP EXE. ERROR
I36	ICM. OPE. ERROR FILE OPE. ERROR
I37	EXTEND INS. ERR.
I38	PRG. TIME OVER
I39	CHK instructions Annunciator detection

Only continues with the interrupt program at operation resumption

(2) Fault history

The latest 16 points of the diagnosis results are stored in the QnACPU module as the fault history.

Furthermore, you can store up to 100 points of diagnosis results by setting the parameters to store to a file in the memory card.

You can confirm this data by using the peripheral device fault history display. This fault history data can be cleared from a peripheral device.

Example: The following diagram shows the fault history display.

- (a) The error code, message, date and time of occurrence, and error step are displayed in the fault history. There is also a HELP display shown for the error.

[Fault History]		
#	Error Message	Date&Time
2110	SP.UNIT ERROR	96-05-17
	File:MAIN	20:14:17
	Err Step1	
1500	AC DOWN	96-05-17
		20:20:21
2501	CAN'T EXE.PRG	96-05-17
		20:52:48

- (b) You can select two kinds of HELP displays, either the SW01VD-GPPQ standard error detailed HELP, or the user detailed HELP which displays messages you have registered.

[Error Detail HELP]	
1. Error #	[2110]
2. HELP Message	
[Instruction]	FROM H0 K0 D0 K1
[Error Unit]	
Guidance	:Being specified by FROM/TO instruction but it is not a Special Function Unit.

[User Detail HELP]	
1. Error #	[2100]
2. User Message	
[Cause]	Cannot execute FROM/TO instruction
[Remedy]	Exchange the specified Special Function Unit
[Contact]	Production Engineering Dept. Ext.2345

When the error involves an error step, displays the step instructions.

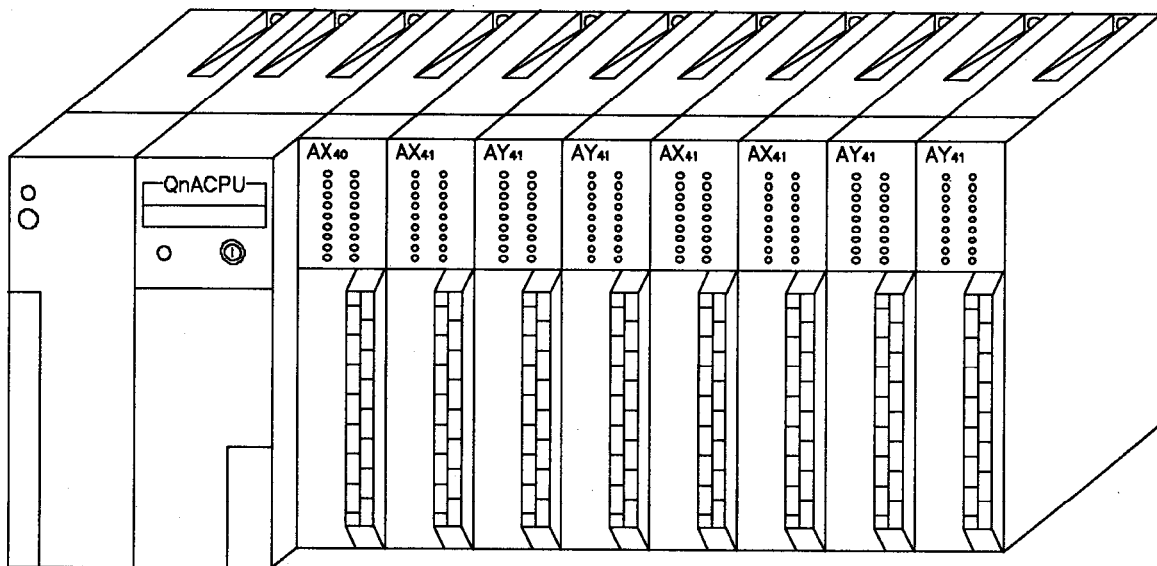
When the error involves an error module, displays the model name of that module (when there is an I/O allocation in the parameters).

3.4.4 Unrestricted I/O allocation in module units

You can allocate the module I/O address to any address in the parameters, regardless of the module's mounting position. If you do not conduct I/O allocation, then the QnACPU automatically conducts the allocation in accordance with the order in which the modules are mounted.

For example, if you must replace a 16-point input/output module with a 32-point module to expand the system, then that module can be allocated to an unused I/O address. Accordingly, only parts of the program relating to the replaced module need to be amended, while parts relating to other modules can be used without amendment.

Example: The diagram below shows an example of I/O allocation that allows the I/O addresses of the modules in slots 1 to 7 to be used without change when replacing an AX40 (16 points) with an AX41 (32 points) at slot 0.



I/O addresses before change	X0	X10	Y30	Y50	X70	X90	YB0	YD0
	XF	X2F	Y4F	Y6F	X8F	XAF	YCF	YEF
					↓ Set I/O allocation			
I/O addresses after change	XF0	X10	Y30	Y50	X70	X90	YB0	YD0
	X10F	X2F	Y4F	Y6F	X8F	XAF	YCF	YEF

[I/O Allocation]						Label :
Slot	Type	Items	1stXY	Type Name		
0(0-0)	<Inp>	<32Pt>	[F0]	[AX41]	Basic
1(0-1)	<Inp>	<32Pt>	[10]	[AX41]	[
2(0-2)	< >	< >	[]	[]	Power Supply
3(0-3)	< >	< >	[]	[]	[
4(0-4)	< >	< >	[]	[]	Extension Cable
5(0-5)	< >	< >	[]	[]	[
6(0-6)	< >	< >	[]	[]	[
7(0-7)	< >	< >	[]	[]	[

3.4.5 Boot operation

You can start PC operations by reading the sequence program from the memory card and then executing it. This is called boot operation. In addition to boot operation, programs can be stored in QnACPU internal memory and executed.

The boot operation is conducted after setting the QnACPU module switches.

Using the boot operation function, you can do the following operations.

- (1) If separate programs are stored in the QnACPU module and memory card, then you can easily switch programs according to the required control.
- (2) By booting from the memory card to the QnACPU module, you can write the program quickly.
Even if you do not have a peripheral device, high speed program writing is possible simply by booting from the memory card.
- (3) During ROM operation, you can read programs from a ROM memory card and execute them.

Example: The following table shows an example of a parameter boot file setting for boot operations having stored parameters or programs in memory card 1, and control data in memory card 2.

[Boot File Setting]		Label :		
#	Program	Type	TX Src Drive	TX Dest Drive
1	[PARAM]	<Parameter >	[1]	[0]
2	[INITIAL1]	<Sequence >	[1]	[0]
3	[INITIAL2]	<Sequence >	[1]	[0]
4	[MACHINE]	<SPC >	[1]	[0]
5	[ASSEMBLY]	<Sequence >	[1]	[0]
6	[TRANSFER]	<Sequence >	[1]	[0]
7	[ERRPROCE]	<Sequence >	[1]	[0]
8	[SCHEDULE]	<Sequence >	[1]	[0]
9	[MONITOR]	<Sequence >	[1]	[0]
10	[LOGGING]	<Sequence >	[1]	[0]
11	[TYPEDATA]	<Dev Init >	[2]	[0]
12	[]	< >	[]	[]

PgUp:Prev PgDn:Next Esc:Close

3.4.6 Data protection function

This function can prohibit the writing/reading of program files, comment files, etc., in the QnACPU module internal memory or memory cards from peripheral devices or serial communication modules.

You can use this function to prevent the careless rewriting of data, or the efflux of program content.

The two methods of protecting data are registering an entry code and changing file attributes.

(1) Entry code registration

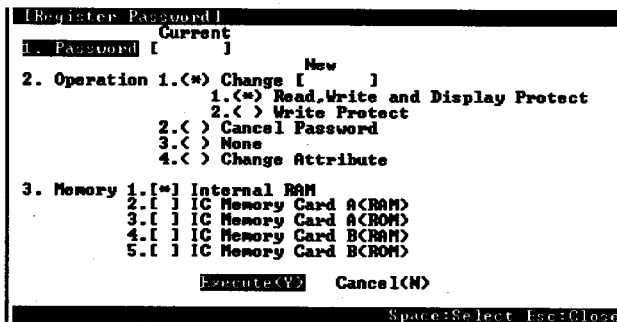
Entry codes can be registered in the internal memory or in memory cards per drive, and the following settings can be made.

(a) Prohibit read/write display

This setting prohibits the reading, writing, referencing, and file list display, of the parameter and program files in the applicable drives.

(b) Write disable

This setting prohibits writing to the parameter and program files of the applicable drives. However, it still permits reading, referencing, and displaying file lists.



(2) Changing file attributes

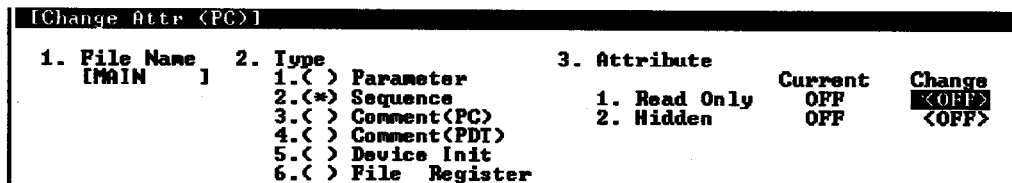
You can change file attributes per file, and make the following settings.

(a) Read Only Files

This prohibits writing to the applicable file. However, reading, referencing, and file list display are still available.

(b) Hidden Files

This prohibits reading, writing, referencing and file list display for the applicable files.



3.4.7 Device initial values

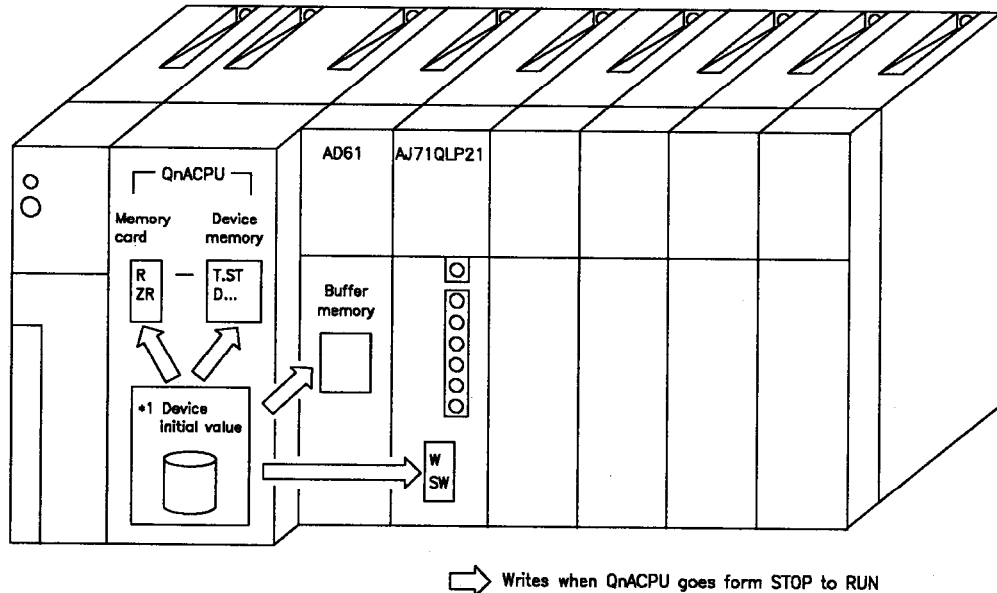
This function automatically writes initial values to device memory, file register and special function modules when the QnACPU starts operating.

The device initial values are created in advance in the peripheral device, and are written to QnACPU internal RAM or the memory card.

Accordingly, device initial setting programs are no longer required and indeed can be discarded. Furthermore, the device initial value data can also be set as a ROM.

Device initial value settings can be made for the following devices.

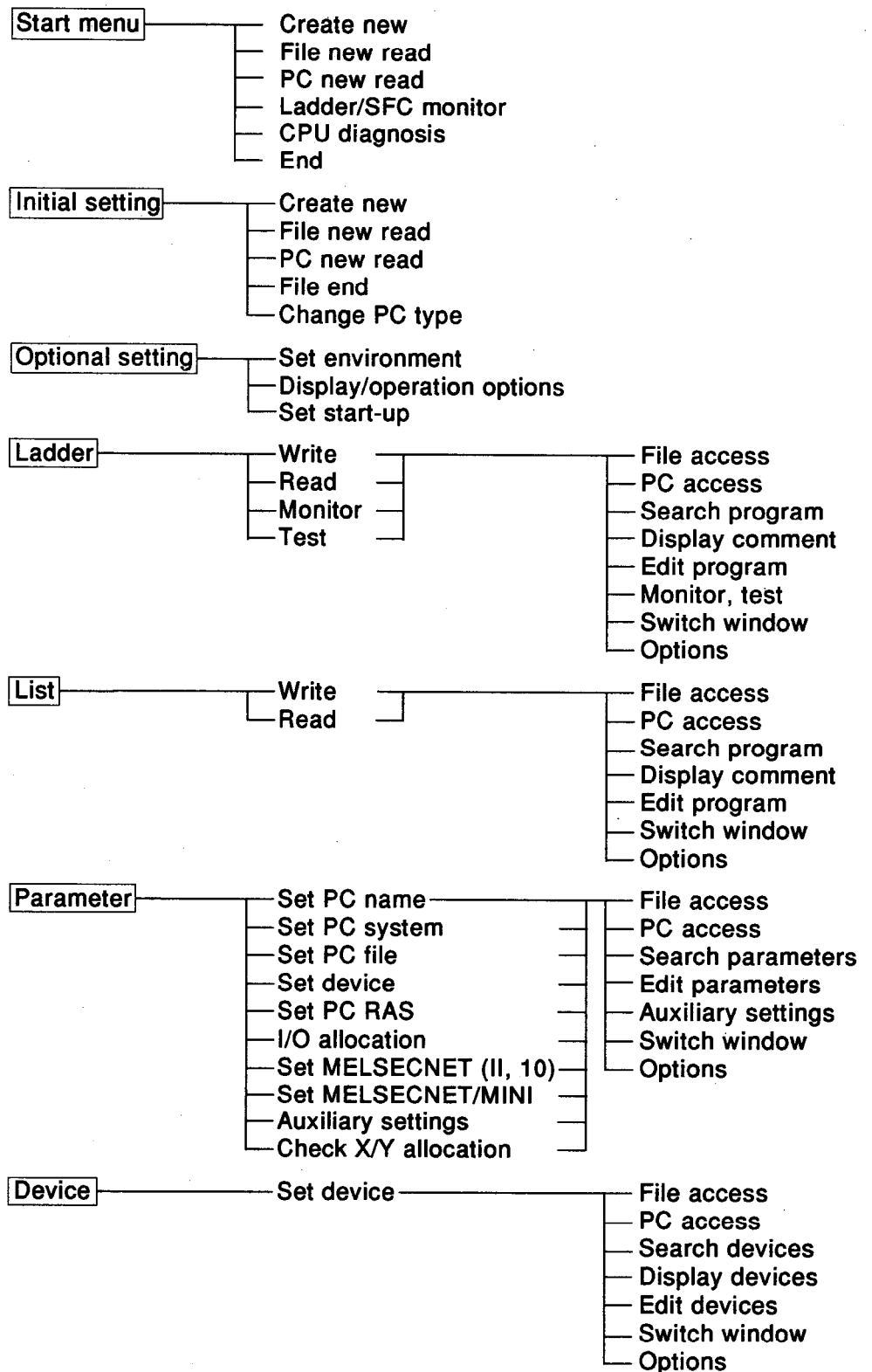
- Timer (T, ST) present value
- Counter (C) present value
- Data register (D)
- Special register (SD)
- Link register (W)
- Link special register (SW)
- File register (R, ZR)
- Special direct device (U[]\G[])
- Link direct device (J[]\W[], J[]\SW[])



*1 The device initial value file is stored in QnACPU internal RAM, or in the memory card.

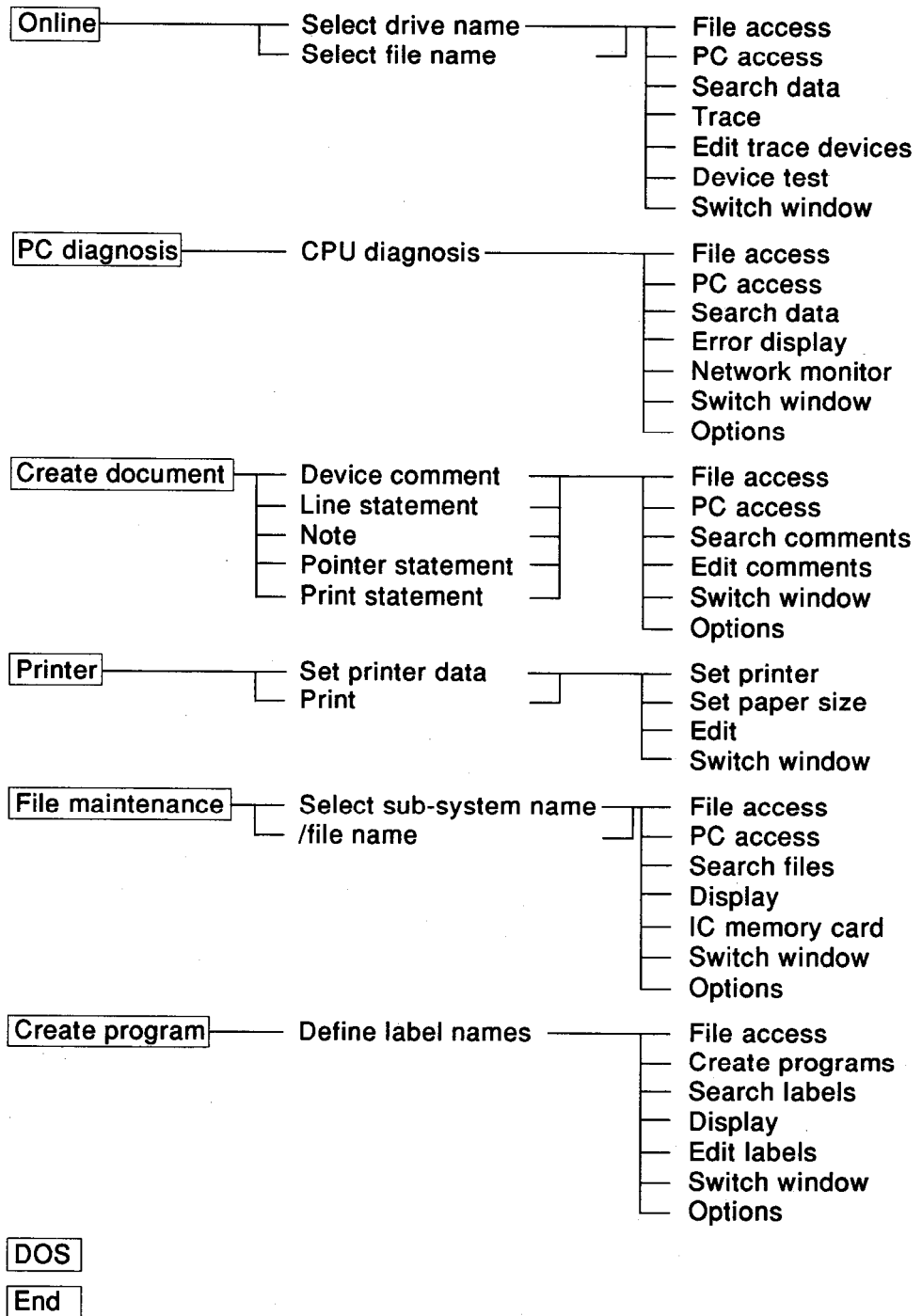
4. GPP FUNCTION SOFTWARE PACKAGE (SW01VD-GPPQ)

4.1 Function List



4. GPP FUNCTION SOFTWARE PACKAGE (SW01VD-GPPQ)

MELSEC-QnA

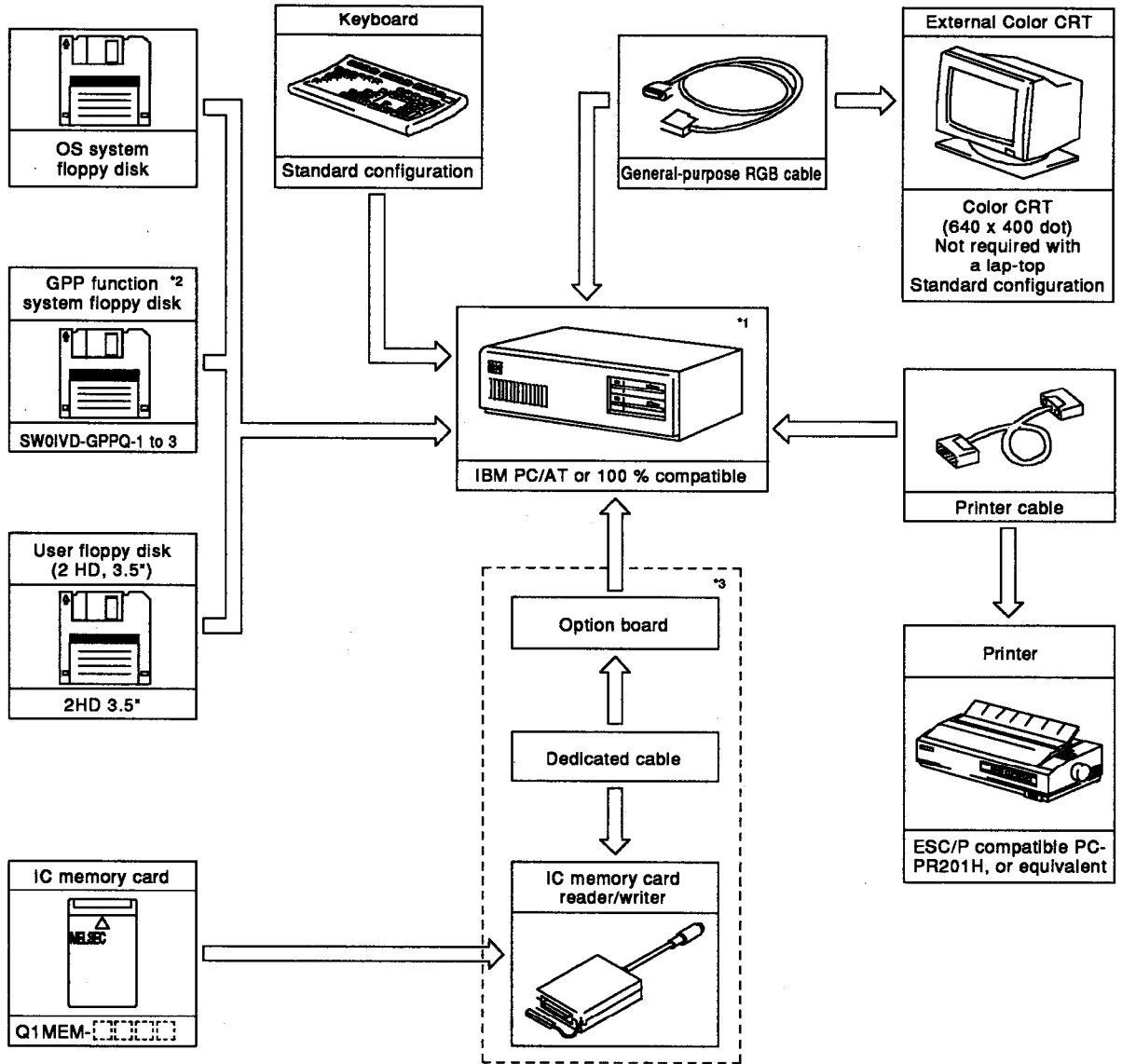


4. GPP FUNCTION SOFTWARE PACKAGE (SW0IVD-GPPQ)

MELSEC-QnA

4.2 System Configuration

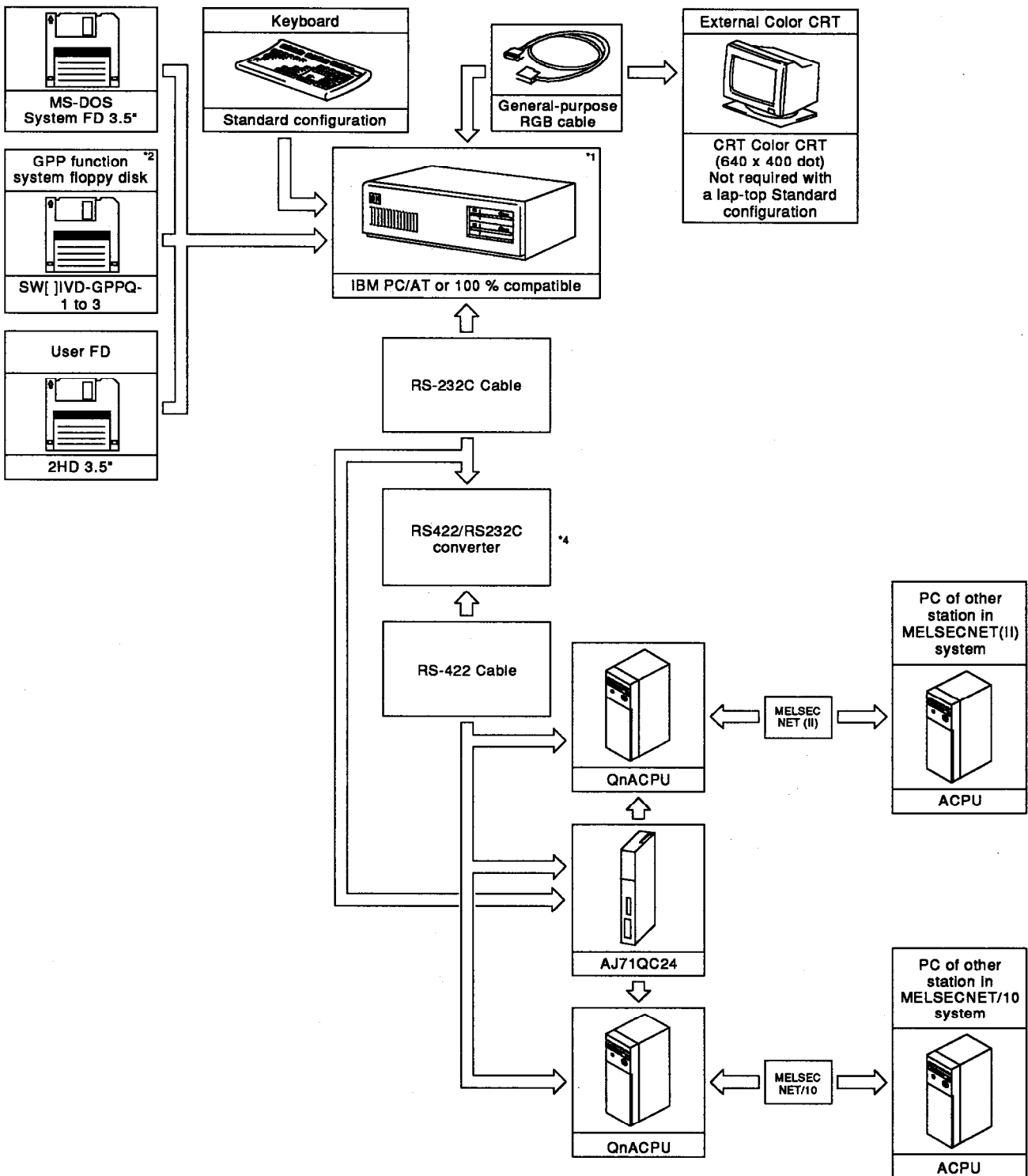
(1) System configuration for offline operation.



4. GPP FUNCTION SOFTWARE PACKAGE (SW01VD-GPPQ)

MELSEC-QnA

(2) System Configuration for Online Operation



4. GPP FUNCTION SOFTWARE PACKAGE (SW0IVD-GPPQ)

MELSEC-QnA

REMARKS

*1 The specifications of applicable IBM PC/AT or 100% compatible computers and basic software which can be used are listed below.

CPU	:80486SX(33 MHz), or higher (recommended: 80486DX2(66MHz))
EMS	:4 MB min.
HD free space	:20 MB min.
Floppy disks	:Reading and writing of 2HD, 3.5" floppy disks formatted for 1.44 MB
OS Version	
PC-DOS version	:5.0, or higher
MS-DOS version	:5.0, or higher
MS Windows	:3.1 (to run GPP functions with DOS conversion box)
EMS driver	:EMM386 (supplied with DOS)
	:QEMM (Version 7.03)
Other	:Install HIMEM and SMARTDRV

We have confirmed the operation of the following IBM PC/AT and compatibles:

IBM	:9545LJG, 2620 2JD
DEC	:FR832JA WB, FR P74WC AJ
COMPAQ	:4/75CX M510W
DELL	:XPS4100/MXV, XPI P90T
TOSHIBA	:FV4755TW

*2 The SW0IVD-GPPQ configuration is shown below.

SW0IVD-GPPQ-1	... 2 floppy disks (copy protected)
SW0IVD-GPPQ-2	... 1 floppy disk
SW0IVD-GPPQ-3	... 1 floppy disk

Before use, install the system in the hard disk.

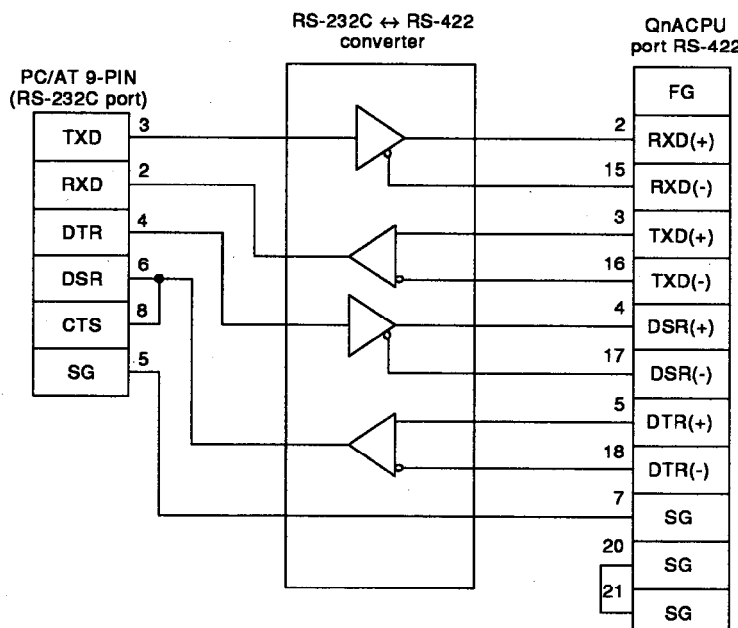
*3 To use the IC memory card reader/writer, the IC memory card reader/writer driver must be incorporated in CONFIG.SYS.

See the IC memory card reader/writer instruction manual for details.

*4 RS-232C ↔ RS-422 converter

The computer and the QnACPU are connected by means of an RS-232C ↔ RS-422 converter.

Shown below is an example of the connections between the computer and the QnACPU through the RS-232C ↔ RS-422 converter. (Connect the wires to the RS-232C ↔ RS-422 as illustrated in the following figure.)



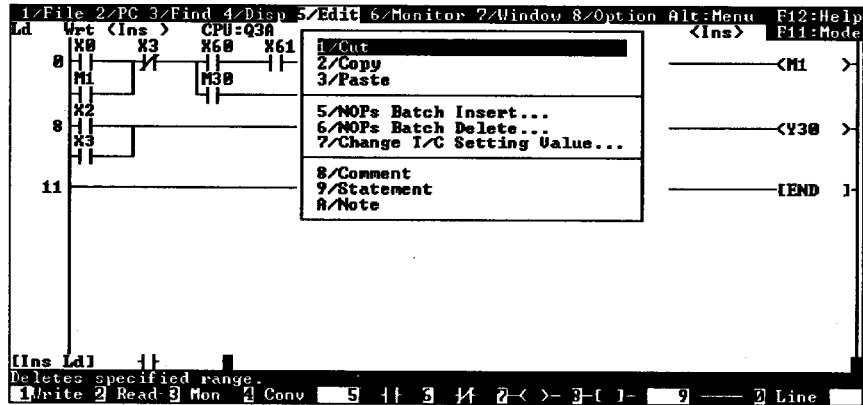
4.3 Functions

4.3.1 Enhanced editing functions

Editing functions, such as cut and paste, replacing, search, ladder line insertion/deletion, and multiple editing object functions provides operability similar to a general-purpose editor, and enable you to reduce the program creation time.

(1) Combining the pull-down menu and function keys

You can easily conduct related operations and mode switching operations, with improved operability.

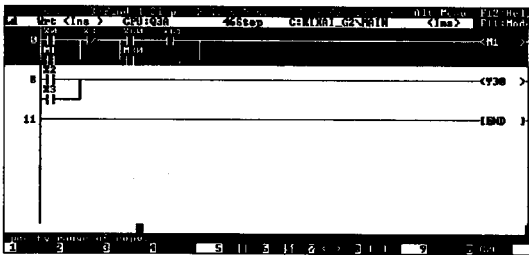


(2) Cut and paste, undo

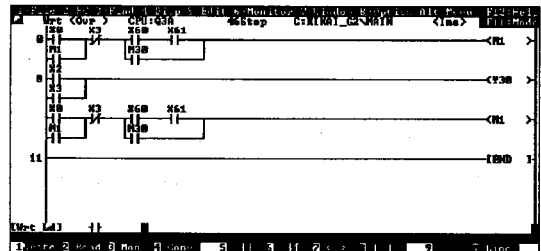
Editing operations such as ladder copy and delete are easier.

Setting copy range

Copy completed



Designate copy destination

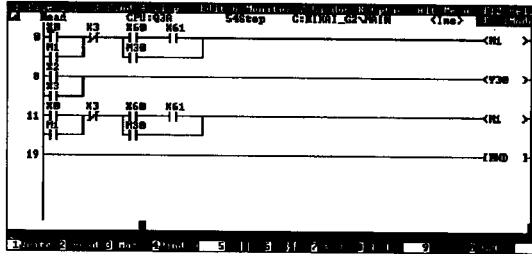


(3) Edit four programs or data at the same time

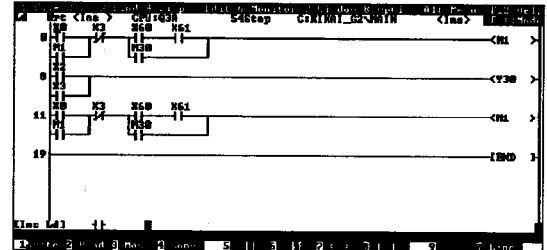
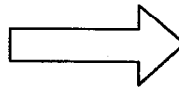
You can designate up to four editing objects at the same time. The different editing objects can be quickly switched between by using the function keys.

(4) Edit while checking the previous and next programs while ladder writing.

The ladder display is shown unchanged, even in edit mode, allowing you to edit while checking the previous and next ladders.



Designate write mode with the cursor at the position step 0 X0

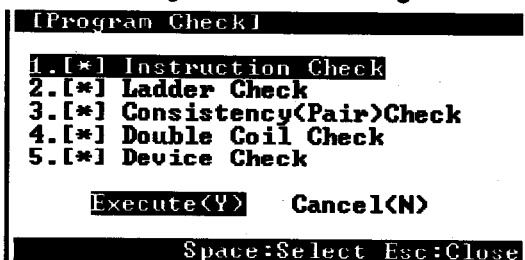


(5) Check already created programs

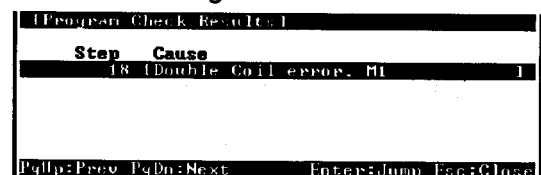
You can check for logical mistakes and input mistakes occurring in programs. The following five types of checks can be done.

- Instruction check : This checks whether or not the instruction code is correct.
- Ladder check : This checks for items which cannot be converted to ladder form when programs are created in the list mode; for example 17 or more ANB/ORB instructions used in succession, or ladders in which the relationships between MPS/MRD/MPP instructions are incorrect.
- Consistency (pair) check : This checks the consistency of instructions. For example, it checks whether there are subroutines without RETs, or whether there are jump destinations without pointers, etc.
- Duplicate coil check : This checks device numbers which are duplicated when using OUT(H), SET, SFT(P), PLS and PLF instructions.
- Device check : This checks whether or not device numbers are within the parameter range.

Program check setting



Program check result



4.3.2 Program standardization

(1) Macro definition

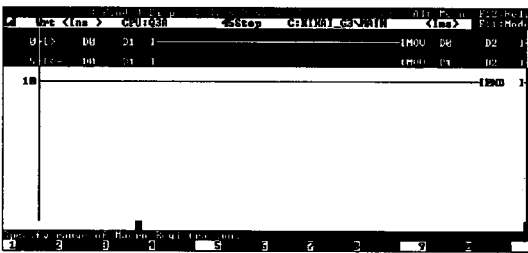
You can register your created programs as macro instructions. These macro instructions can be used as one instruction when creating sequence programs.

Programs after macro magnification are converted to the same instructions as a normal sequence program. Then when the ladder display is done by the peripheral device, statements are automatically attached to the macro instructions, allowing confirmation of the macro instructions.

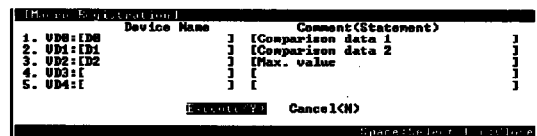
Using macro instructions cannot reduce the number of program steps, however even if you use them a multiple number of times in one program, it is easier to debug than subroutine programs. Furthermore, you can also use the device as an argument, and so can designate devices suitable for processing using macros.

(a) Macro registration

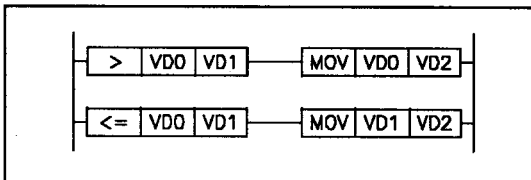
1) Macro ladder registration



2) Macro argument setting

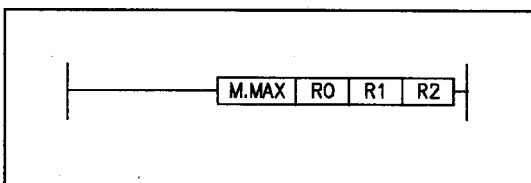


3) Macro instruction entity

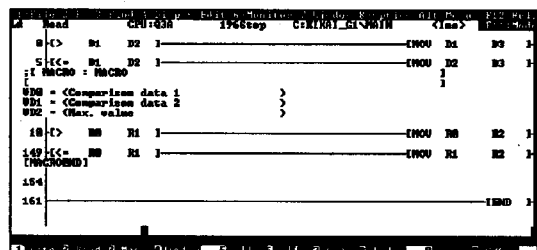


(b) Macro utilization

1) At program input



2) Program after macro magnification



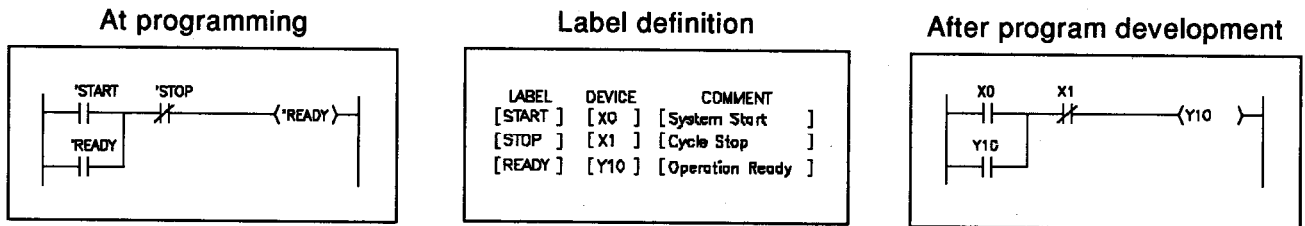
4. GPP FUNCTION SOFTWARE PACKAGE (SW0IVD-GPPQ)

MELSEC-QnA

(2) Programming with "general-purpose" CPU type

With standardized programs, programming is carried out without deciding the PC type or device numbers, and with the CPU type set to "general-purpose". Programming is carried out by using labels instead of devices.

After programming, you can produce programs to fit particular systems by making allocations for the CPU module type and system device numbers for which labels were used.

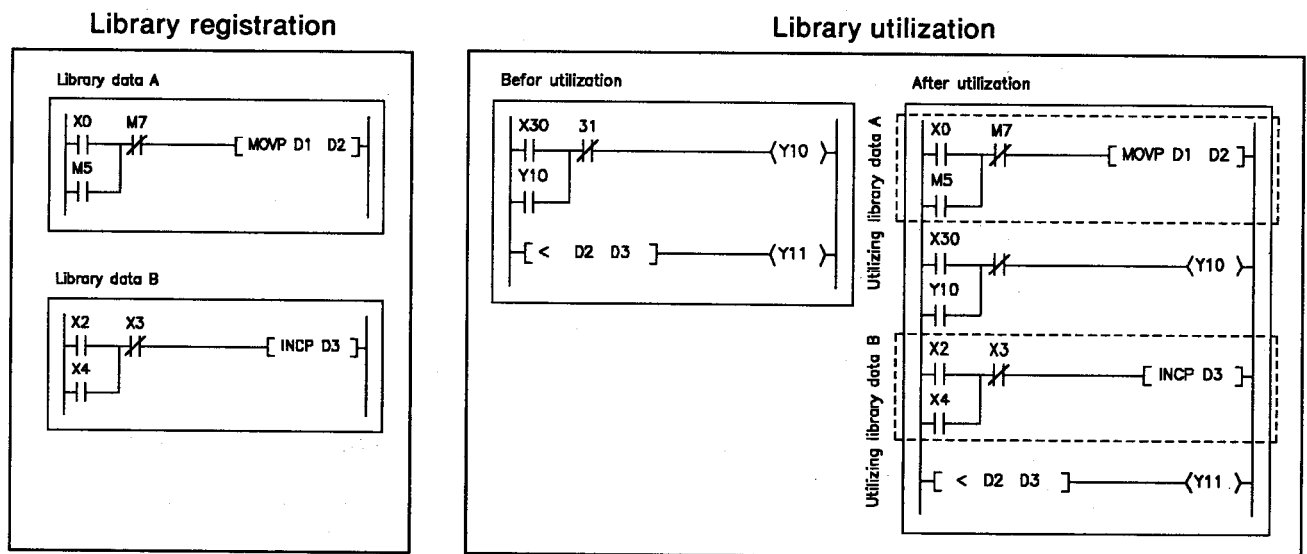


(3) Library registration and utilization

If you assign one part of a program a file name as library data and register it, then when you edit other programs you can utilize that program.

You can copy programs in file units.

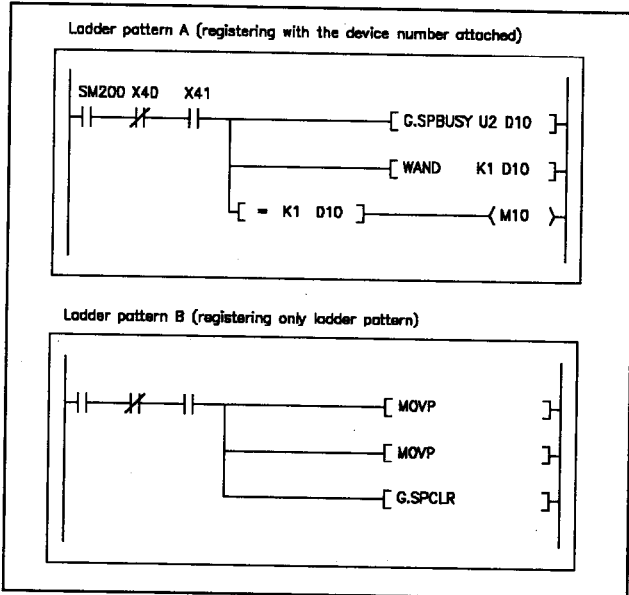
The device number of the program utilizing the library data is the same number as when the library data was registered.



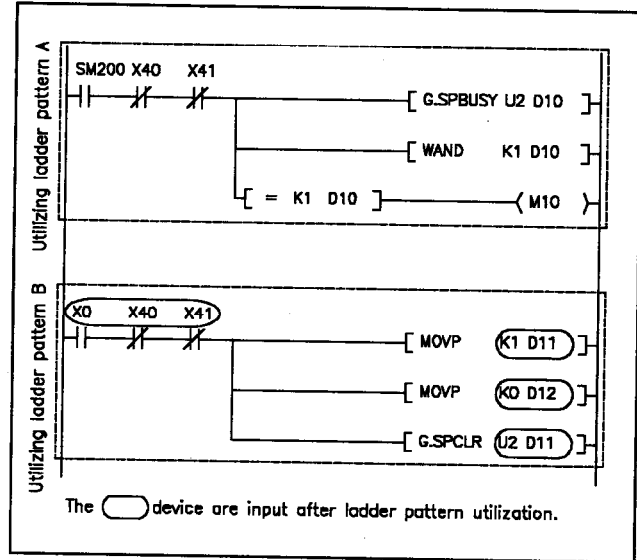
(4) Ladder pattern registration and utilization

If you assign file names to and register ladder patterns which you frequently use, then you can use the ladder pattern during program editing. You can choose to register just the ladder pattern itself, or you can register it with a device number appended.

Ladder pattern registration



Ladder pattern utilization



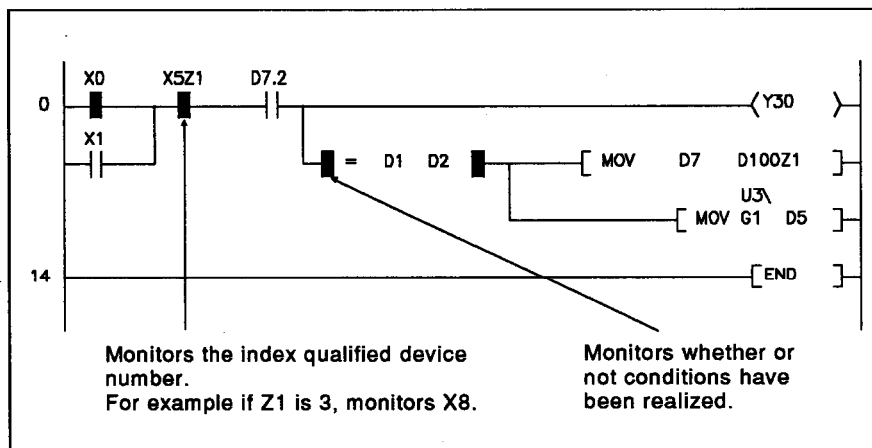
4.3.3 Debugging function

(1) Monitoring

(a) Device monitoring

Index qualified devices, devices with extension designations (I/O No., network No.), and extension file registers, can be monitored.

1) Example of monitoring continuity status



2) Example of monitoring word device data

(16-bit integer)

D0
123

(Integer)

D10
<123.45>

(Text)

D20
"ABC"

(b) Monitoring timing

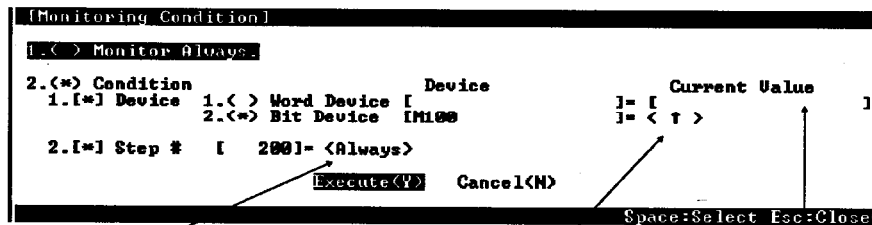
You can set the timing and conditions for the monitoring of the QnACPU by a peripheral device, and monitor in accordance with very precise timing. Monitoring is normally conducted at the END processing of a program.

You can designate the following conditions.

- 1) You can designate the monitoring timing as a step No.
In this case, monitoring is conducted just before executing the designated step No.
- 2) You can designate the monitoring condition as a word device present value, or the rise/fall of a bit device.

The peripheral device performs monitoring when both monitoring conditions (1) and (2) are satisfied. If the monitoring conditions are not satisfied, then the previous monitoring status is held.

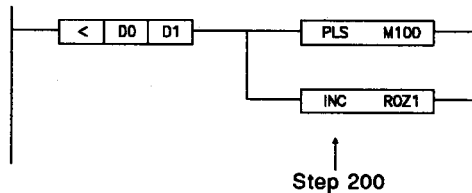
Example: An example of setting monitoring for when M100 rises, in step number 200.



Sets either normal/ON/OFF/rise(↑)/fall(↓) for the continuity status of the designated step.

Sets rise(↑)/fall(↓) for the designated bit device.

Sets the present value of the designated word device.



(c) Measuring execution time

You can monitor the processing time for each program precisely, and make adjustments to the system easily.

The scan time and number of executions for each file is monitored. The execution time for any section in a program file can also be measured while online.

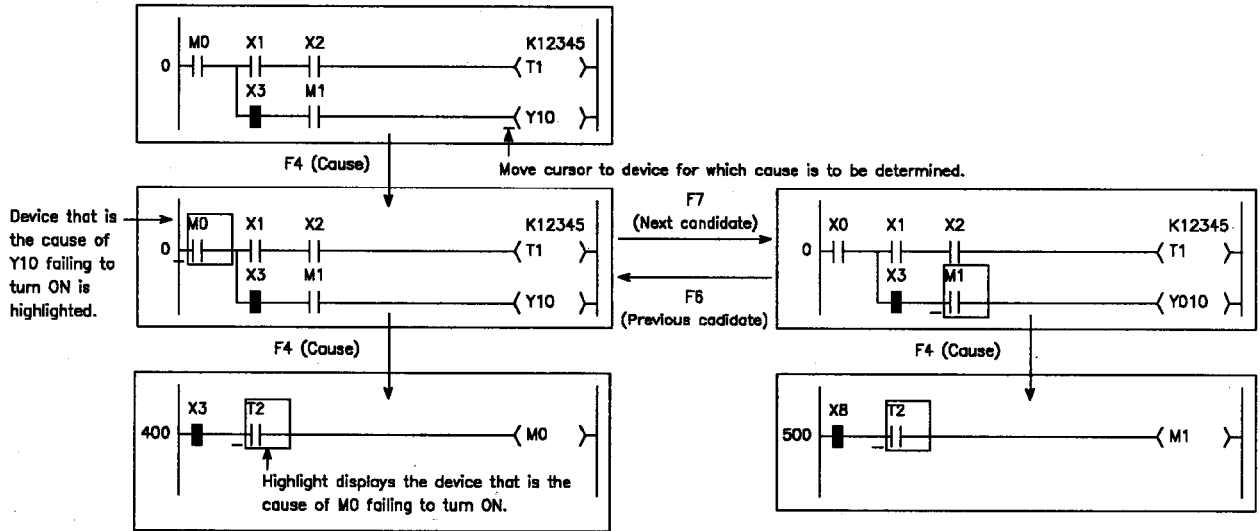
Example: The following diagram shows an example of monitoring a program list.

<Total Scan Time>			<Program Status>				
	Mon Time	Max Scan	#	Program	Exec	Scan Time	Ex Times
Scan	200ms	120.000ms	1	INITIAL	Init	0.100ms	1 x
Init	ms	120.000ms	2	MACHINE	Scan	0.100ms	1400 x
Slow	ms	0.200ms	3	ASSEMBLY	Scan	0.100ms	1400 x
<Time Details / Scan>			4	TRANSFER	Scan	0.100ms	1400 x
Program		0.300ms	5	TEST	Wait	0.000ms	0 x
END Proc Time		119.700ms	6	MONITOR	Slow	0.300ms	57221 x
Slow Prog		110.600ms	7		Wait	0.000ms	0 x
Wait for Con		112.200ms	8		Wait	0.000ms	0 x
			9		Wait	0.000ms	0 x
			10		Wait	0.000ms	0 x
			11		Wait	0.000ms	0 x

(2) Determining ON/OFF causes

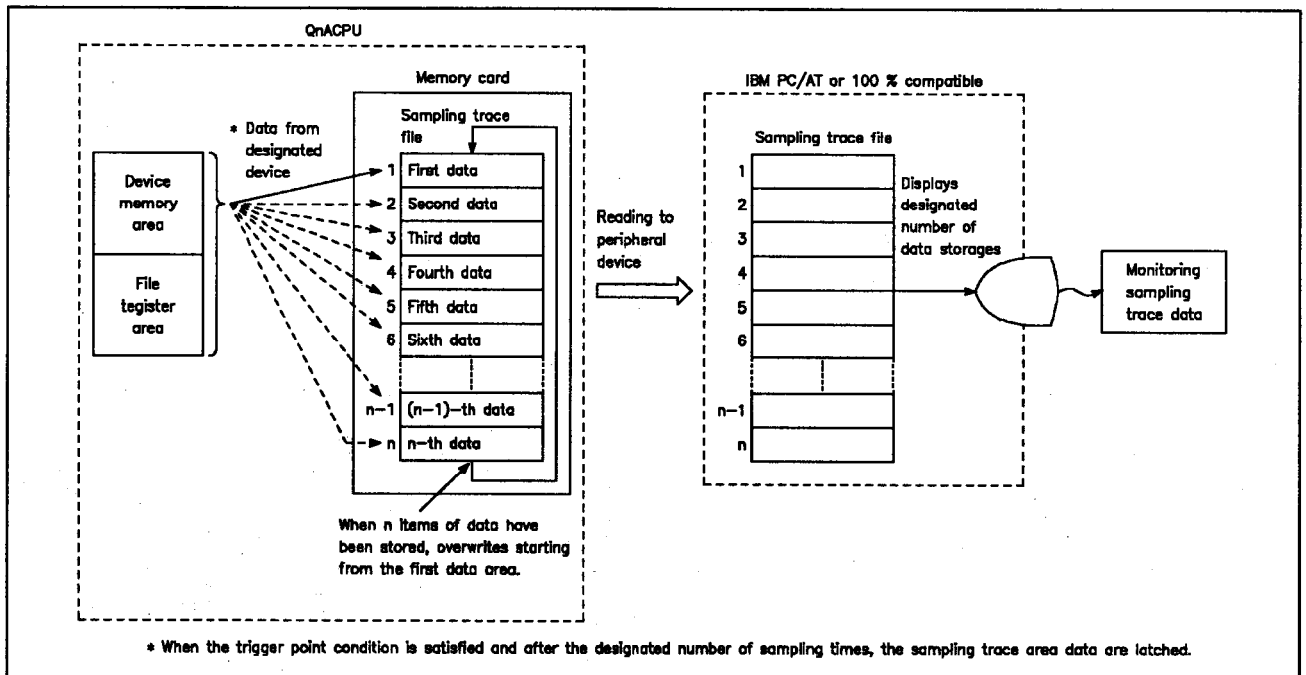
In this operation, by designating a coil that does not turn ON/OFF during ladder monitoring, a search is automatically conducted for the device that is the cause and the ladder turning ON/OFF. Using this function, you can easily conduct ladder searches during debugging.

Example:



(3) Sampling trace

Bit device ON/OFF statuses, and changes in word device contents, can be stored to the memory card in time series. It is also possible to follow-up fast device changes which cannot be confirmed just by peripheral device monitoring.



(a) Execution method

1) Registration

The sampling trace conditions are set at the peripheral device, and written to the QnACPU.

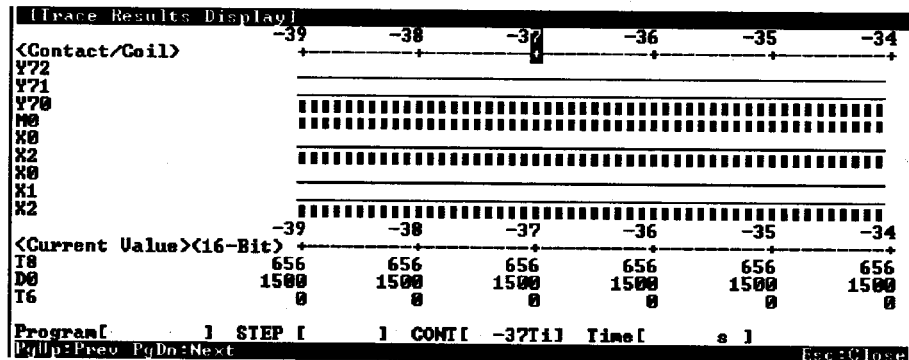
Conditions	Setting Contents
Number of device points	<ul style="list-style-type: none"> • Bit device: : 0 to 50 points • Word device: : 0 to 50 points
Number of traces	<ul style="list-style-type: none"> • Sets the total number, and the number after trigger. • Total number/trigger number : 0 to 8192
Trace point	<ul style="list-style-type: none"> • Selects the point at which the trace starts from one of the following. <ul style="list-style-type: none"> 1) END → At END instruction of each scan. 2) Time → At each designated time. Time can be set from 5 ms to 5 seconds, in 5 ms units. 3) Detailed conditions → Same as for monitoring conditions described in Section 4.3.3 (1) Monitoring.
Trigger point	<ul style="list-style-type: none"> • Selects the trigger point from one of the following. <ul style="list-style-type: none"> 1) At instruction execution → Makes the execution of the STRA instruction the trigger. 2) At trace execution → Makes the trace execution at the peripheral device the trigger. 3) Detailed conditions → Same as for monitoring conditions described in Section 4.3.3 (1) Monitoring.
Trace additional information	<ul style="list-style-type: none"> • Sets the additional information to be added in each trace. • Can select one or more from the following. <ul style="list-style-type: none"> 1) Time → Stores the time when trace executed. 2) Step name → Stores the step No. where the trace was executed. 3) Program name → Stores the program name where the trace was executed.

2) Execution

The sampling trace is started from the peripheral device. Sequential data are collected in the QnACPU according to the trace point conditions. When the trigger point condition is satisfied, the number of data after the trigger is collected and the sampling trace is concluded.

3) Reading and displaying the trace results

You can read the trace data from the QnACPU (or memory card) to a peripheral device, and display the results.



REMARK

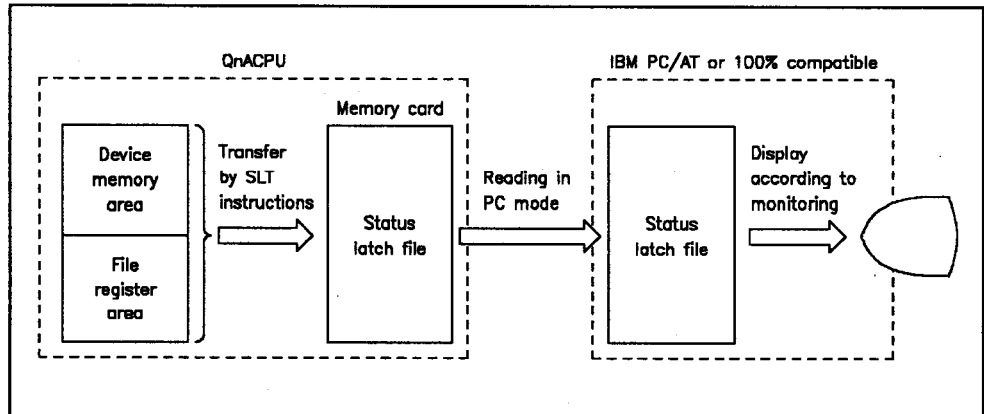
There is also a monitoring trace function that can conduct the same sort of monitoring as the sampling trace function does.

Data collection and data storage are conducted by a peripheral device, so this function can be executed even if there is no memory card in the QnACPU. However, the data collection interval is decided according to the monitoring speed of the peripheral device, and so takes longer than the sampling trace.

(4) Status latch

All devices or just designated devices are stored to the memory card in the designated timing during sequence execution. The stored data are read to the peripheral device, and can be displayed on the ladder monitoring or device monitoring screens.

This function is useful when you are seeking the cause of faults.



(a) Execution method

1) Registration

The status latch conditions are set at a peripheral device, and written to the CPU module.

Condition	Setting Content
Status latch devices	<ul style="list-style-type: none"> • Sets devices for which status latch is conducted. • Sets whether to conduct for all devices, or to designate a device range. Device range up to a maximum total of 1000.
Trigger point	<ul style="list-style-type: none"> • Execute at STL instructions. • Execute at trigger instruction from peripheral device. • Execute when device condition satisfied.

2) Execution

The data of the designated devices is stored to the memory card when the trigger point condition is satisfied.

3) Reading and displaying status latch data

The status latch data is read from the QnACPU (or memory card) to the peripheral device. The peripheral device monitoring destination is switched to status latch data, and the results of ladder monitoring and device monitoring are displayed.

(5) Program trace

This is a function for confirming the execution status when a program consists of multiple files. Also use this function to conduct device data traces when you are conducting a program trace.

The QnACPU traces the program execution status, and stores it to memory card. The trace data can be read and checked at a peripheral device.

You can designate one of the following for trace timing:

- At the execution of branch instructions (CALL, JMP, etc.)
- At the execution of interrupt programs
- At the execution of PTRAs (trace) instructions

(a) Execution method

1) Registration

The program trace conditions are set at a peripheral device, and written to the QnACPU.

Condition	Setting Content
Number of traces	<ul style="list-style-type: none"> • Sets total number and number after trigger. • Total number/trigger number : 0 to 8192
Trace point	<ul style="list-style-type: none"> • Selects point at which the trace is conducted from the following. (When more than one, traces by OR conditions) 1) Branch instructions → At execution of CALL, JMP instructions 2) Interruptions → Each execution of interrupt program 3) Instructions → Execution of PTRAs instructions
Trigger point	<ul style="list-style-type: none"> • Selects the trigger point from one of the following. 1) At instruction execution → Makes the execution of STRA instruction the trigger. 2) At trace execution → Makes the trace execution at the peripheral device the trigger. 3) Detailed conditions → Same as for monitoring conditions described in Section 4.3.3 (1) Monitoring.
Trace device	<ul style="list-style-type: none"> • Bit device : 0 to 50 points • Word device : 0 to 50 points

2) Execution

The program trace is started from a peripheral device. Sequential data are collected in the QnACPU in accordance with the trace point conditions. When the trigger point condition is satisfied, the number of data after the trigger is collected and the program trace is concluded.

3) Reading and displaying trace results

This involves reading the trace data from the QnACPU (or memory card) to the peripheral device, and displaying the results.

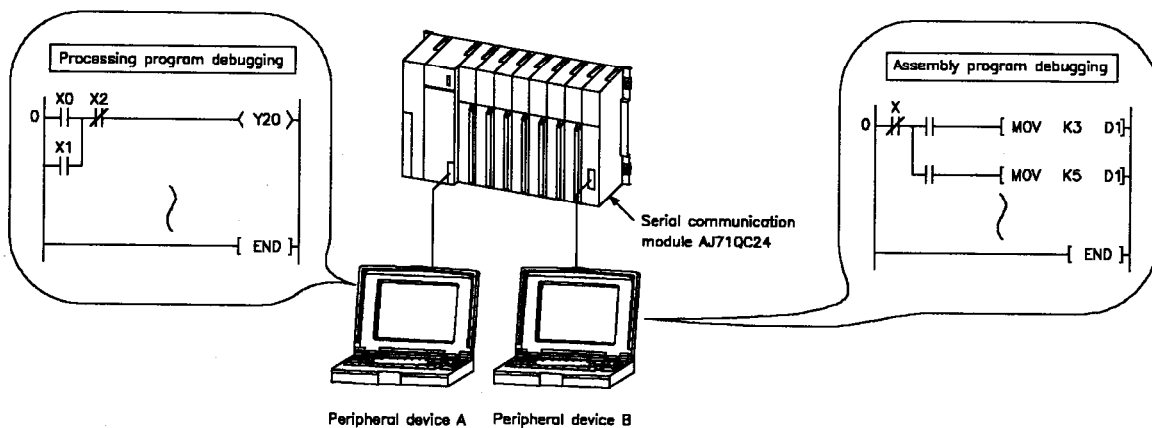
(6) Debugging conducted by a number of people

Monitoring, testing and other operations can be conducted at one QnACPU from a multiple number of peripheral devices.

(a) To access one QnACPU from a multiple number of peripheral devices, you must connect to either a serial communication module (AJ71QC24), or to another station of the MELSECNET/10 network system.

(b) If you designate different file name programs for each peripheral device, then operations can be conducted freely without affecting other peripheral devices, in the same way as if only one peripheral device was connected.

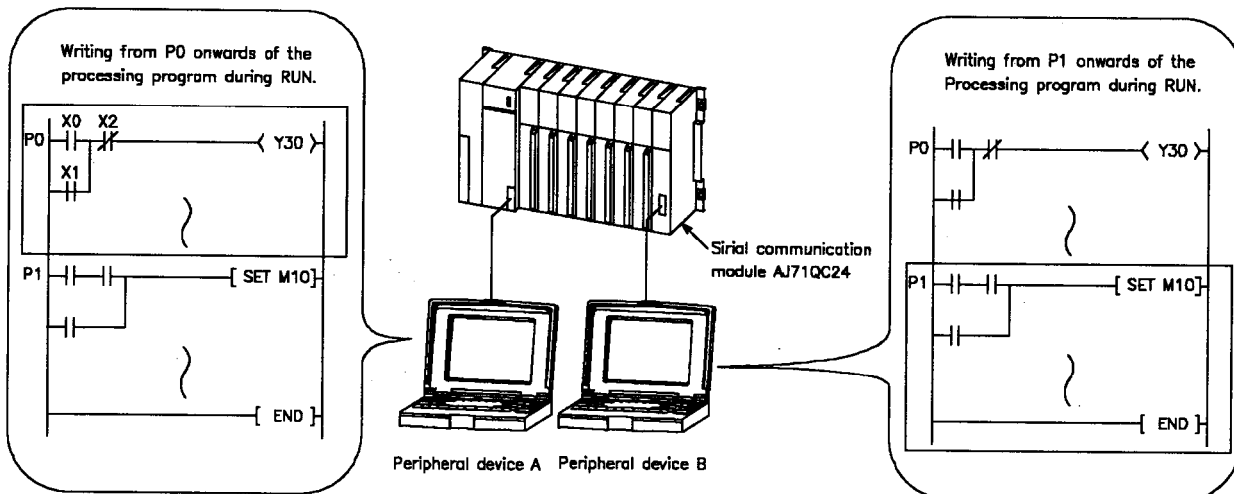
You can conduct test operations, program writing (including write during RUN) and other operations simultaneously from multiple peripheral devices.



(c) It is possible to write programs to just one file from multiple peripheral devices during RUN.

Preset pointers for the programs to be written during RUN from each peripheral device. By doing this, you can conduct safe writing during RUN from each peripheral device.

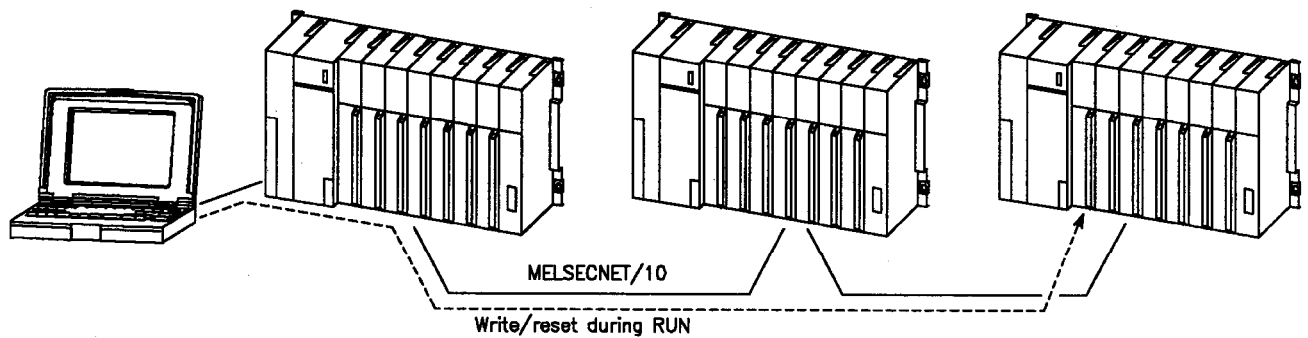
The diagram below is an example of peripheral device A writing from P0 during RUN, and peripheral device B writing from P1 during RUN. The programs enclosed by a are the objects of writing during RUN.



4. GPP FUNCTION SOFTWARE PACKAGE (SW01VD-GPPQ)

MELSEC-QnA

- (d) You can carry out program writing during RUN from a peripheral device connected to another station of the network. Remote RUN/STOP/RESET are also possible, enabling you to remotely reset a QnACPU that has had its operations stopped by a self-diagnosis error.



4.3.4 Document creation function

- (1) This function has enhanced the device comment, statement, and note functions, and has improved the readability of programs.
- Comments 32 characters, can be registered for all devices.
 - Device name . . . 10 characters, can be registered for all devices.
 - Statement 64 characters, can register any number of points provided within the program capacity range.
 - Note 64 characters, can register any number of points provided within the program range.

REMARK

Statements and notes are stored together with the program. However, you can remove the statements and notes from the program and store them in QnACPU. As this only involves removing one step at the position where the statements/notes were created, it hardly effects the sequence program.

- (2) Comments, labels, statements and notes can also be input while inputting the program, making document creation easy.

- (a) An example of inputting device comments when writing instructions:

When the device comment shown below is input following input of the instruction (" \rightarrow X0")

[Comment write] X0 : air pressure normal
└─ Input device comment

- (b) An example of designating a device number and entering a device comment

::X0=air pressure normal
└─ Attach "::" └─ Device number └─ Device comment

- (c) An example of inputting a line statement

: Ready ladder
└─ Attach ":" └─ Line statement

- (d) An example of inputting a pointer statement

P0: Initial setting program
└─ Pointer number └─ Attach ":" └─ Pointer statement

- (e) An example of inputting a note

OUT Y10: Operation ready completion processing
└─ Output instructions └─ Attach ":" └─ Note

- (3) You can printout ladder patterns, macro instructions, library data, etc., making the management of standard programs easier.

- (4) You can also printout device memory data, initial value data and PC diagnosis data.

4. GPP FUNCTION SOFTWARE PACKAGE (SW01VD-GPPQ)

MELSEC-QnA

4.3.5 Utilizing a MELSEC-A series program

(1) Conversion from A to QnA

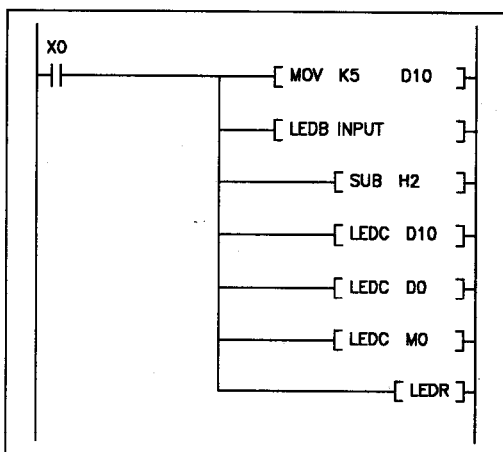
By using the A to QnA conversion function of SW01VD-GPPQ you can utilize programs that were created in the MELSEC-A series in the QnA series.

Parameters, sequence programs + statements + notes, comments, and device memory data can be converted. Conduct the conversion after selecting the items you wish to convert.

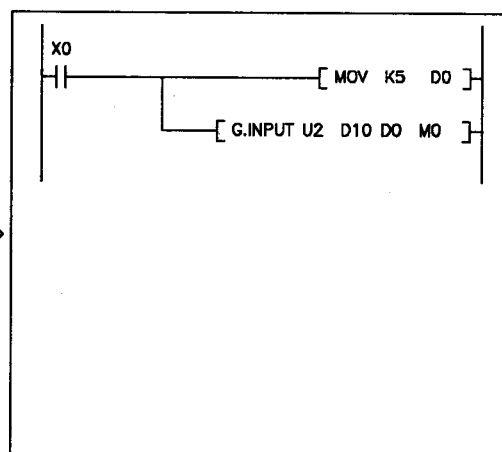
(You need to make partial amendments for special relays and special registers which cannot be used in QnA after conducting the A to QnA conversion.)

The following diagram shows an example of a data receive program in serial communication module no-protocol mode.

Before conversion (MELSEC-A program)



After conversion (MELSEC-QnA program)



A to QnA
conversion

(2) QnA to A conversion

Alternatively, you can use the QnA to A conversion function to create MELSEC-A programs using the QnA series software development environment.

By using this function when creating programs, you can achieve program standardization and modulization, such as registering and utilizing macro instructions, and creating standard programs for the "general-purpose" CPU type.

You can create programs with instructions and device ranges which can be used in the MELSEC-A CPU module, and then converted from QnA to A.

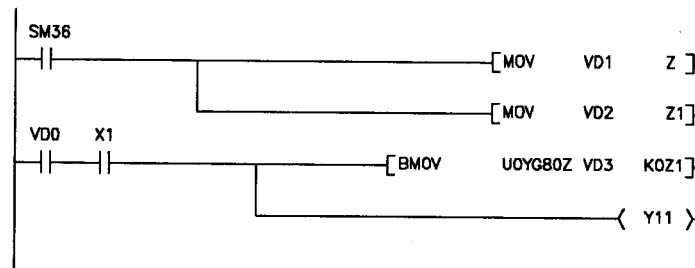
4.4 Powerful Support Tools

4.4.1 SW0IVD-MSPQ/MSDQ-type macro/library software package

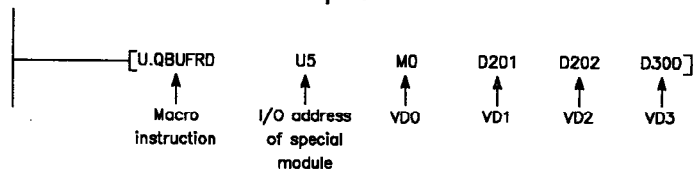
This software package includes standard programs and standard ladders for special function modules, registered as macros. It is useful as a collection of macro and library data for use when creating sequence programs. When creating user sequence programs, use these data as macro instructions.

- (1) SW0IVD-MSPQ is a package of basic sequence programs which access the special function module, in the form of macros and library data.
This package simplifies programming of the special function modules.
- (2) SW0IVD-MSDQ packages standard sequence programs, such as those fault detection and alarm processing, as macros and library data.
You can easily apply this package in programs.
- (3) When using macros, designate the macro names and actual devices which are to be the arguments when creating programs.
For example, the following diagram shows an example of utilizing the macro instruction "QBUFRD" reading data received in the serial communication module no-protocol mode.

Macro instruction: QBUFRD after magnification



Macro instruction description

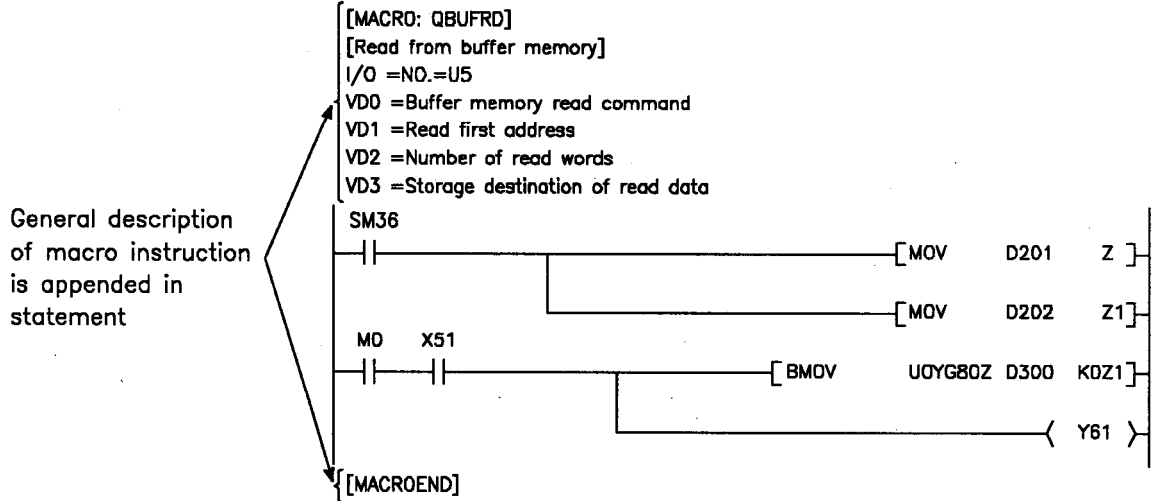


(1)

(1)

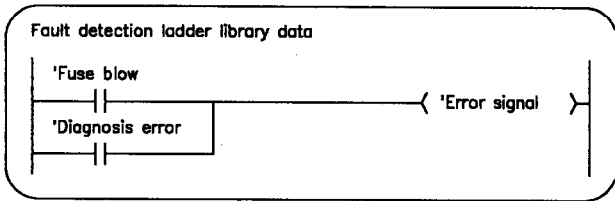


Actual program after macro magnification

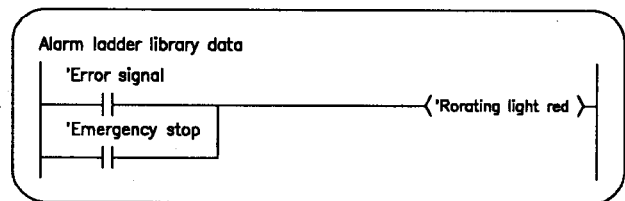


- (4) Library data consists of standard programs created for the "general-purpose" PC type. Programs are generated by defining labels. You can create meaningful programs by combining the different macros and library data. For example, the following diagrams show an example of a program created by combining two library data for a "fault detection ladder" and an "alarm ladder".

Fault detection ladder library data



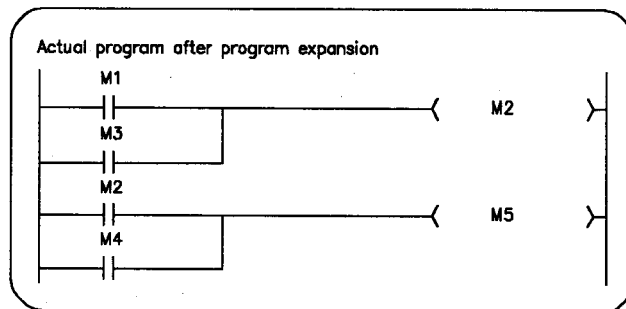
Alarm ladder library data



The above two library data are combined and the labels defined.

Label	Device
Fuse blow	M1
Error signal	M2
Diagnosis error	M3
Emergency stop	M4
Rotating light red	M5

Actual program after program expansion



4. GPP FUNCTION SOFTWARE PACKAGE (SW0IVD-GPPQ)

MELSEC-QnA

- (5) The following table shows the main models of special function module supported and details of their applications.

Supported Model	Applications
AJ71C24	Communication processing in no-protocol, dedicated protocol
AJ71PT32-S3	Communications with batch refresh type remote I/O module Data communications with operation box Data input from bar code reader Data communication by no-protocol mode
AD70	Processing such as home position return, positioning, speed to position control switching, fixed-pitch feed, and JOG operations
AD71(S1)	Basic reading/writing processing JOG operation Positioning address teaching Present position change Positioning start program Manual pulse generator Home position return Stop during positioning, etc.
AD61	Setting processing for using/not using ring counter function Mode register setting Preset value data setting Counter input enabling Set value reading External preset detection 1-phase designated acceleration/deceleration designation Match signal reset Present value reading
A68AD	Used channel designation Averaging processing designation Write data error code reading A/D conversion completed flag reading Average time, average number of times setting Digital output value reading Reset
A68DAV	Digital value setting Processing when installed at remote I/O station
AJ71E71	Initial processing Fixed buffer communications Open processing
AD51	Buffer memory reading/writing processing
AD51H	Buffer memory reading/writing processing
AD51FD	Fault information read processing
AD57	Mode setting processing
AJ71QC24	Communication processing in no-protocol, dedicated protocol

4. GPP FUNCTION SOFTWARE PACKAGE (SW0IVD-GPPQ)

MELSEC-QnA

4.4.2 SW0IVD-CNVQ data conversion software package

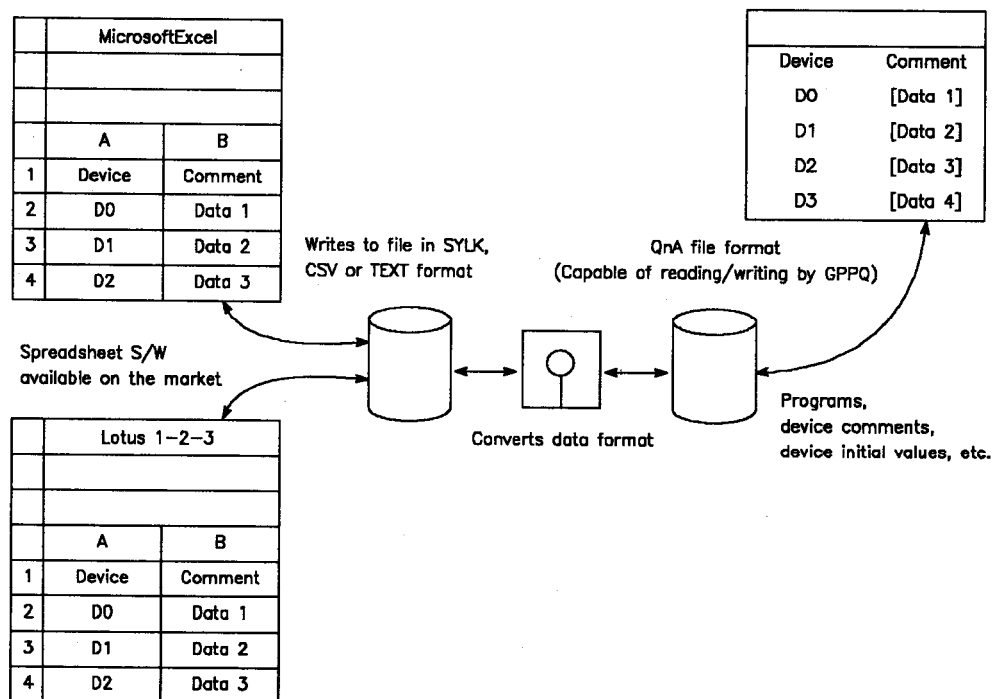
This software package converts numeric data and character data created by OA spreadsheet software or word processor/editors, into data that can be used by MELSEC-QnA series peripheral devices.

Furthermore, this package can also convert data created by QnA peripheral devices to data or data or text format that can be used with spreadsheet software.

By using this software package, you can use the spreadsheet software packages or editors that you are most familiar with, improving the productivity of programmable controller software development.

(1) Data conversion for spreadsheet software

You can convert the tabular data created by spreadsheet software such as EXCEL, Lotus 1-2-3, Multiplan, etc., e.g. device comments and device initial values, into data capable of being used by SW0IVD-GPPQ. Furthermore, using this package you can also convert data created by SW0IVD-GPPQ to a data format capable of being used by spreadsheet software.



(2) Converting text format data

Text format data created by word processors or editors, such as device comments or device initial values, can be converted into data that can be used by SW0IVD-GPPQ.

Furthermore, you can also convert device comments and device initial values created by SW0IVD-GPPQ into text format data.

4. GPP FUNCTION SOFTWARE PACKAGE (SW01VD-GPPQ)

MELSEC-QnA

(3) The following table shows data that can be converted

o : Conversion possible — : Conversion impossible

Data \ Conversion Form	GPPQ to Spreadsheet	Spreadsheet to GPPQ	GPPQ to Text	Text to GPPQ
Ladder data	—	—	o	—
SFC data	—	—	o	—
List data	o	o	o	o
Device comments	o	o	o	o
Device initial values	o	o	o	o
Device labels	o	o	o	o
Device memory data	o	o	o	o
Parameters	o	o *2	o	o *2
Macro/library data	o	o	o	—

*1 Only sampling trace, monitoring trace and status latch can be converted

*2 Excludes network parameters

(4) The following table shows the software on the market and file formats that are capable of conversion.

o : Conversion possible — : Conversion impossible

File Format \ Software on the Market	EXCEL	Lotus 1-2-3	Multiplan	Other Editors
TEXT	o	o	—	o
SYLK	o	—	o	o
CSV	o	o	—	o

4. GPP FUNCTION SOFTWARE PACKAGE (SW0IVD-GPPQ)

MELSEC-QnA

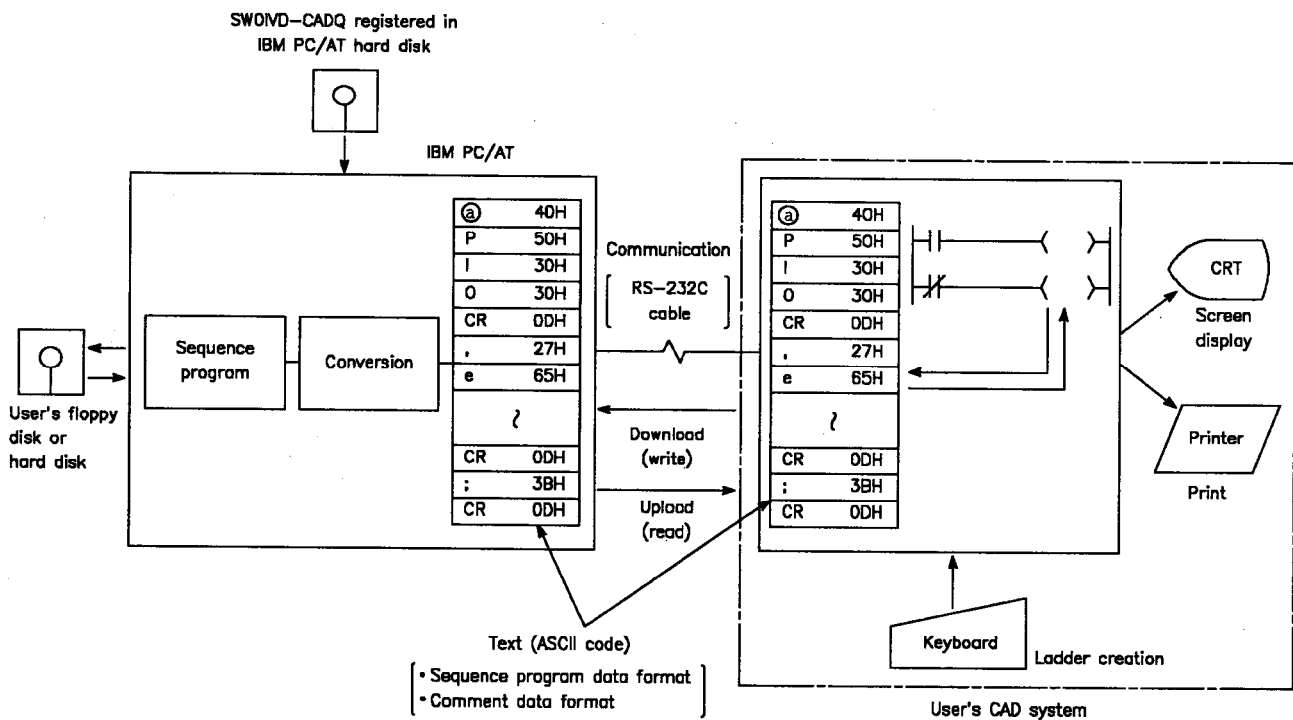
4.4.3 SW0IVD-CADQ-type CAD conversion software package

This software package converts programs or comment data created in CAD systems to data that can be used in SW0IVD-GPPQ. By using this software package you can apply CAD systems to your needs, helping to improve the productivity of programmable controller software development.

(1) Connected to CAD systems and peripheral devices by RS-232C.

ASCII data in the set format is converted into SW0IVD-GPPQ data when it is sent from a CAD system.

The diagram below outlines data communication using sequence ladder symbols.



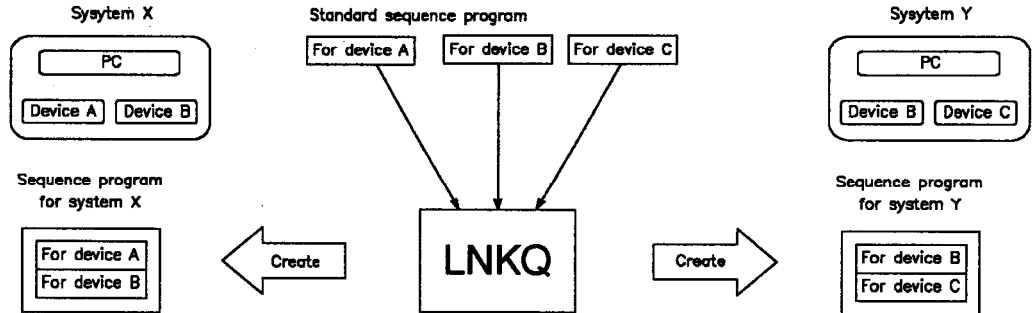
(2) The following data can be converted

Ladder symbol data
Instruction list data
SAP3 symbol data
Comment data
Macro, library data

4. GPP FUNCTION SOFTWARE PACKAGE (SW0IVD-GPPQ)

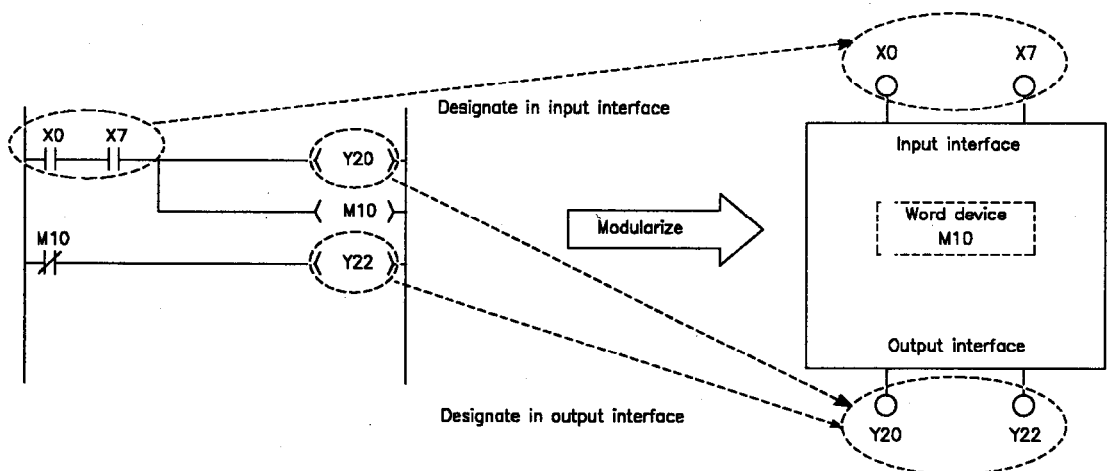
4.4.4 SW0IVD-LNKQ-type ladder sequence program linking software package

This software package allocates devices to standard sequence programs that you have created to make them applicable to actual systems, and links these programs, to create sequence programs. By using control sequence programs with a high level of independence as modules, you can make them very easy to utilize for required applications.



(1) Modularized programs supported

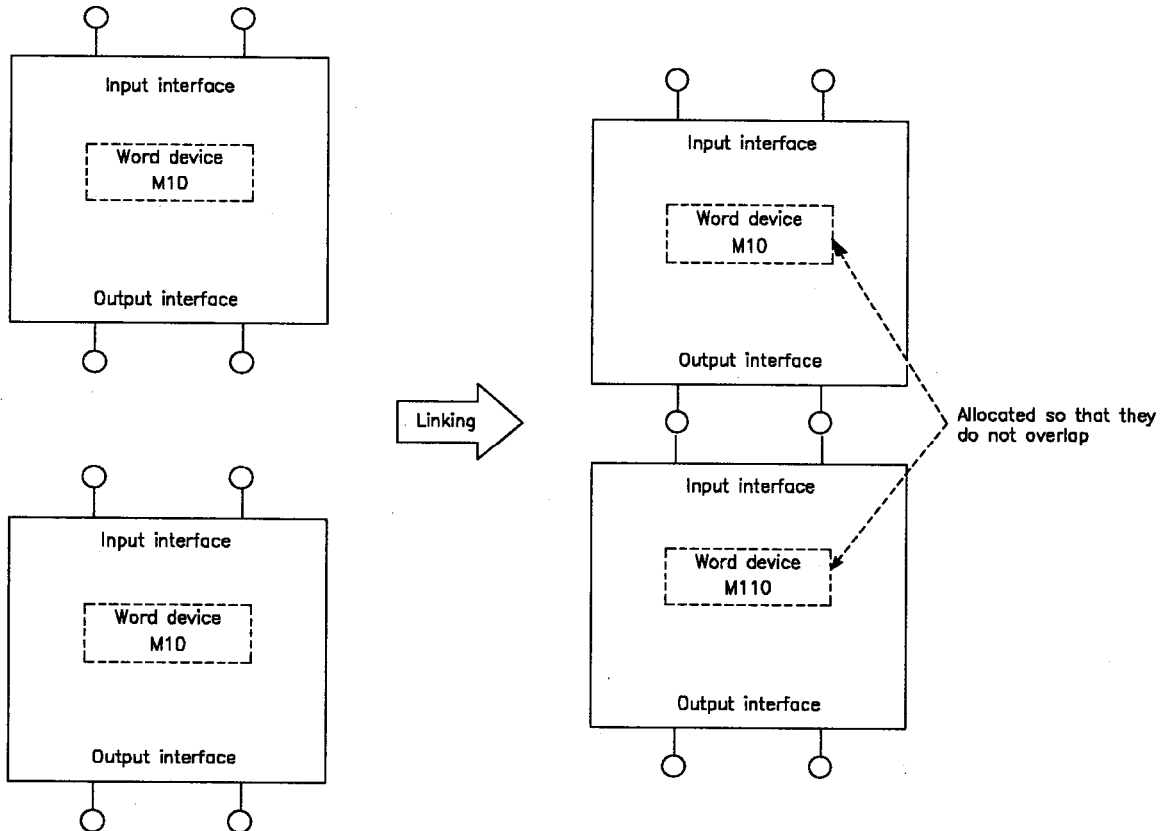
- (a) For each standard program, you can designate devices as the input/output interfaces between programs.
- (b) Data exchange with other generated sequence programs can only be done through devices designated as input/output interfaces. As a result of this, the independence of each standardized program is not lost after linking.
- (c) In debugging, simply by checking the data converted at the input/output interface, modularized programs with bugs can be cut out.



(2) Automatic allocation of device numbers

Devices (or word devices) used in standard programs are automatically allocated so that during program linking they do not overlap with other standard program devices.

You can create standard programs without giving thought to the different CPU types, I/O allocations or device numbers, etc., which differ depending on the system.



(3) Debugging by a number of people possible

During program creation you can either make just one file, or divide the program into multiple files in any combination.

Accordingly, when debugging operations are conducted by a number of people, it is possible to divide the files into separate sections for each designer.

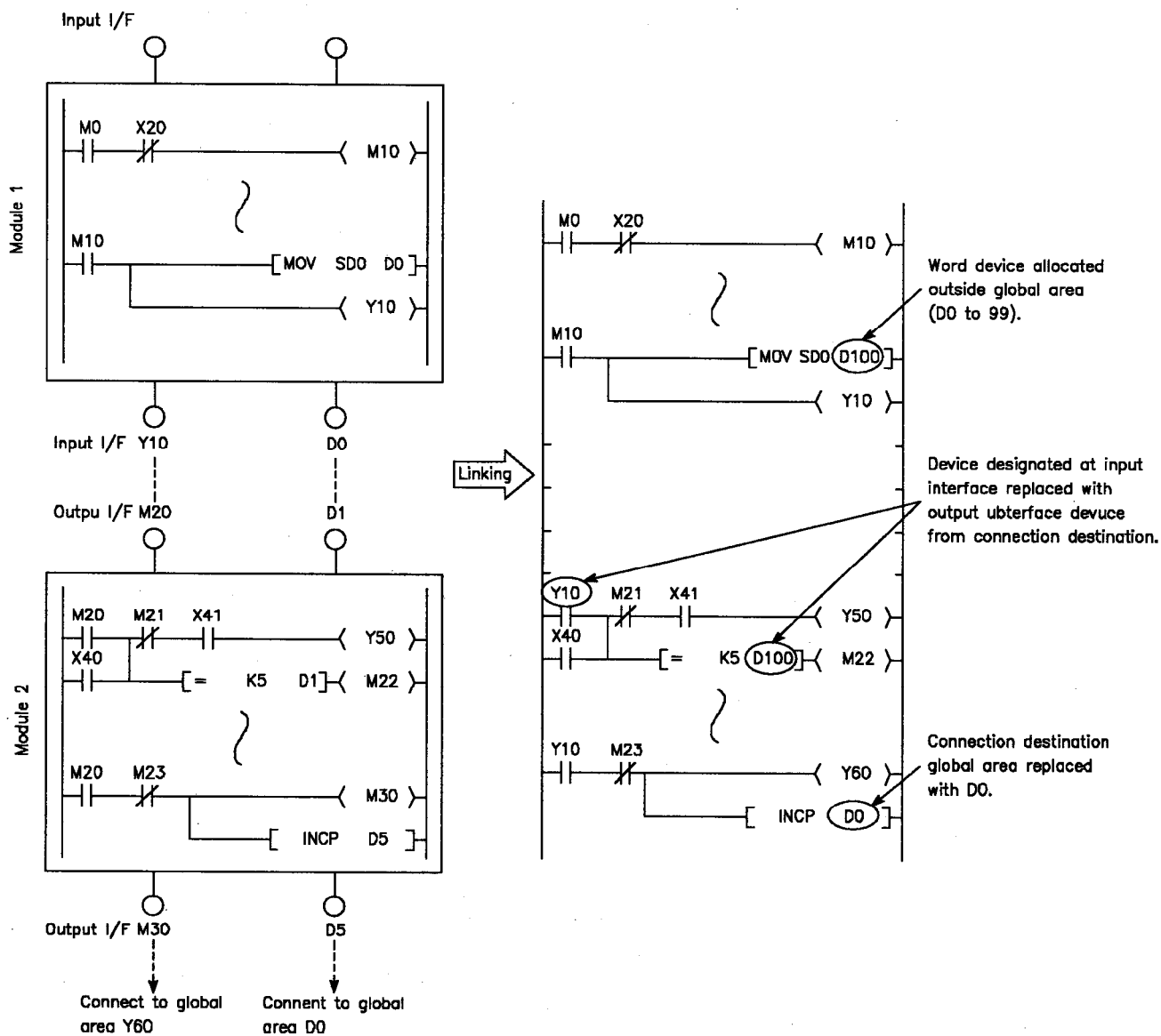
4. GPP FUNCTION SOFTWARE PACKAGE (SW0IVD-GPPQ)

MELSEC-QnA

(4) An example of program generation

The following diagrams show an example of linking module 1 and module 2 of a standard program, using the ladder sequence program linking software package.

(Linking conditions are global area: D0 to 99, I/O section copy: No)



5. PROGRAMMING UNIT (Q6PU)

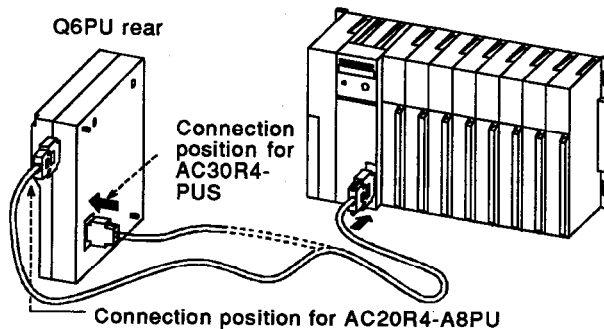
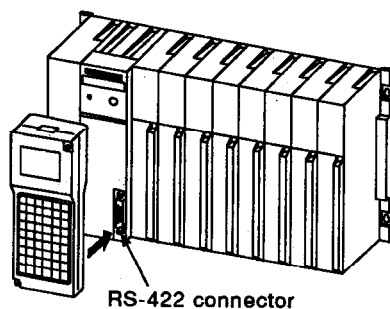
5.1 Performance Specifications

Item	Specification		
Connection target	Q2ACPU, Q2ACPU-S1, Q3ACPU, Q4ACPU		
Power supply, current consumption	Power from connection target (5 VDC 0.55 A)		
Connection method	Add-on	Installed in PC CPU main unit by one-touch method	
	Hand-held	Connected with RS-422 cable	
Display method	Character display by LCD, 4 lines x 20 characters (with cursor) Backlight in display		
Operation method	54 operation keys (Covered by polyester film)		
Key operation confirmation	Buzzer sound		
Indicator service life	Minimum 100,000 hours (Working ambient temperature 15 to 35 °C, ambient humidity not exceeding 65 % RH)		
Backlight service life	Minimum 50,000 hours (Working ambient temperature 25 °C) Backlight switched off if there is no key operation for more than 10 minutes.		
Key service life	1,000,000 times		
External interface	RS-422	Main unit rear panel	Add-on connection, AC30R4-PUS cable connection
		Main unit right side	AC20R4-A8PU cable connection
	Extension interface	Main unit top panel	Not used
External dimension mm (in)	188 (7.4) (height) x 95 (3.74) (width) x 44.5 (1.75) (depth) Depth 37.5 (1.47) for add-on installation to QnACPU		
Weight kg (lb)	0.5 (1.1)		

REMARKS

[Add-on format]
Connect directly to PC CPU.

[Hand-held format]
Connect to PC CPU using AC20R4-A8PU, AC30R4-PUS cable.



5.2 Function List

Mode	Function			Operation Objective	
Write	Program write			Write/add, change program	
	Device change			Change device used in designated step in program	
	Help	Extension characters			Select extension characters (., \, ", \$, @, ;) used in instructions.
		Write	Instruction help	Instruction display/select	Display/select instructions starting with designated characters.
			NOP continuation		Set NOP (no operation) for the designated range in the program.
		Step read			Designate step number and read program.
		Device comment display			Display comment for designated device.
Read	Program read			Designates step number and reads program.	
				Designates instructions being used and reads program.	
				Designates device being used and reads program.	
	Automatic scrolling			Displays the read program while automatically scrolling up to the designated step.	
	Help	Extension characters			Character selection when reading instructions using extension characters.
		Read	Step		The same as the reading function and automatic scrolling function for the above programs.
			Instruction		
			Device		
			Automatic scrolling		
		SFC read	SFC block read		Reads designated block.
			SFC step read		Reads designated step of designated block.
			SFC transition read		Searches for the transition start instruction in the step.
	SFC active step read			Reads active step in designated block.	
Device comment display			Displays comment of designated device.		
Insert	Program insertion			Insert a new program into the existing program.	
	Help	Extension characters			Select extension characters used in instructions.
		Insert	Instruction help	Instruction display/select	Display/select instruction starting with designated character.
			Move		Move designated range in program to designated position.
			Copy		Copy designated range in program to designated position.
		Step read			Designate step number and read program.
	Device comment display			Display comment of designated device	
Delete	Program delete			Delete program of designated step.	
	Help	Delete	Range designation	Delete designated range in program.	
			NOP batch	Batch delete NOP instructions included in program up to END instruction. (Does not delete NOPLF instructions.)	
	Device comment display			Display comment of designated device.	

5. PROGRAMMING UNIT (Q6PU)

MELSEC-QnA

(from previous page)

Mode	Function		Operation Objective		
Monitoring	List monitor		Read program of designated step, and display the instruction continuity status, ON/OFF status of contact, and present value.		
	Monitor search		Searches for and displays OUT/SET/RST instructions using device of designated contact. (Continuous monitoring).		
	Device monitor		Displays the ON/OFF statuses of bit devices, and the present values of word devices (includes set values for T-C).		
	Help	Display change		Displays the numeric values being displayed in a designated format, or ASCII characters corresponding to the numeric values.	
		Monitor	List monitor	Read	Reads the program of the designated step and monitors.
				Search	Conducts the same operations as the monitor search function above.
		Device monitor		Monitor	Conducts the same operations as the above device monitoring function for the designated device.
				SFC block monitor	Reads and monitors the designated block.
				SFC step monitor	Reads and monitors the designated step of the designated block.
			SFC active step monitor	Monitors the active step number of the designated block.	
Extension characters		Character selection for monitoring instructions using extension characters.			
Device comment display		Displays comment of the designated device.			
Test	Test in list/device monitoring		Conducts SET(ON)/RST(OFF) for the bit device. Changes the present value of the word device.		
	Help	Display change		Displays the numeric values being displayed in a designated format, or ASCII characters corresponding to the numeric values.	
		Test	List monitor test	Reads the program of the designated step, and conducts same operation as test in the above list monitoring.	
			Device monitor test	Conducts same operation as in the above list monitoring for the designated device.	
	Extension characters		Character selection for testing instructions using extension characters.		
Device comment display		Displays comment of designated device.			
Parameter	Parameter clear		Conducts all clear for only the PC CPU and memory card parameters.		
	Parameter setting *1		Sets parameters such as timer timing, common pointer, and latch range.		
	Applicable memory		Selects the write destination memory for the set parameters.		
	Entry code setting		Sets the entry code. Also changes the entry code.		

*1 You can set the timer timing setting, common pointer setting, latch range setting, local device setting, WDT setting, program setting and boot file settings parameters at the Q6PU. Settings such as I/O allocations and MELSEC/10 network parameters cannot be made.

5. PROGRAMMING UNIT (Q6PU)

MELSEC-QnA

(from previous page)

Mode	Function		Operation Objective		
Other	T-C set value change		Change set value of designated device (T-C).		
	PC check	Error step read	Display the content of error occurring in QnACPU, and the step number of the error.		
		Annunciator display	Displays the number of annunciators (F) that are ON, and the annunciator numbers.		
		Program check	Instruction check	Confirms the instruction code, etc.	
			Duplicated coil check	Confirms for duplicated coils in the program.	
		Link communication test	Conducts MELSEC/10 link communication test.		
	PC system	Monitor	Link monitor	Displays MELSECNET(I)/B/10 link status.	
			Buffer register monitor	Monitors the buffer memory content of the designated address for the I/O module at the designated input/output number.	
			Clock monitor	Monitors and sets the QnACPU clock (SD210 to SD212).	
			Terminal	Displays messages designated by MSG instructions, and conducts key operations on execution of PKEY instructions.	
		All clear	PC memory format	Conducts all clear of QnACPU memory, returning to the initial status.	
			Program all clear	Clears the currently selected programs.	
			Device memory all clear	Clears all device memory except for SM, SD and R.	
		Switching	PC No. setting	Switches the target PC CPU for each Q6PU mode operation.	
			Program select	Selects the file to be the editing object at the Q6PU.	
			Device comment select	Selects the device comment file to be displayed at the Q6PU.	
		Other	Remote RUN/STOP	Forcibly switches the QnACPU operating status (RUN/STOP)	
			Link start/stop	Conducts link start/stop on MELSECNET/10.	
			PC memory sort	Condenses the files dotted around the memory area, and creates a continuous free area.	
			Program delete	Deletes designated program files in designated memory.	
			Program copy	Designates the copy source and copy destination drives, and copies program files.	
			Program size change	Changes the program capacity of programs currently being edited.	
		PU setting	Program mode select		Sets whether or not to conduct program write during RUN, and whether or not to use only Q6PU monitor mode and test mode.
			Continuity status display		Sets the display items when using the list monitor function (the continuity status for each instruction, and the ON/OFF status for each bit device).
	Buzzer setting		Sets buzzer ON/OFF during key operations.		
	Entry code input		Conducted when an entry code is registered.		
	Communication waiting time setting		Sets the communication time check time when communicating.		

6. SFC

6.1 Performance Specifications Related to SFC Programs

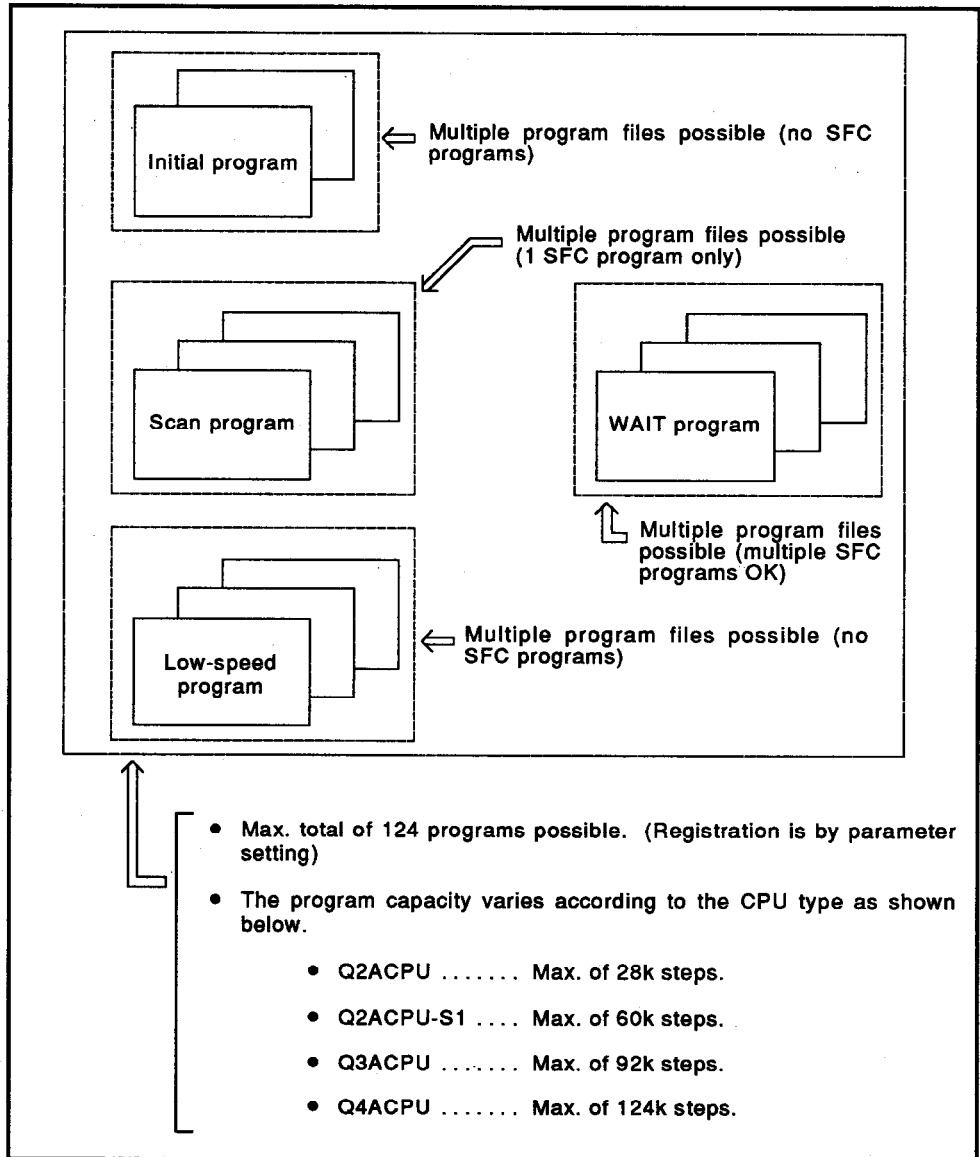
Performance specifications related to SFC programs are shown in the table below.

Performance Specifications Related to SFC Programs

Item		Q2ACPU	Q2ACPU-S1	Q3ACPU	Q4ACPU	
SFC program	Capacity	Max. 28k steps	Max. 60k steps	Max. 92k steps	Max. 124k steps	
	Number of files	1 file (number of scannable files)				
	Number of blocks	Max. of 320 blocks (0 to 319)				
	Number of SFC steps	Max. of 8192 steps for all blocks (512 steps per block)				
	Number of branches	Max. of 32				
	Number of concurrently active steps	Max. of 1280 steps for all blocks (256 steps per block) (including HOLD steps)				
	Number of operation output sequence steps	Max. of 4k steps per block; no per step restrictions				
	Number of transition conditions sequence steps	Max. of 4k steps per block; no per transition condition restrictions				
STEP-RUN function	Break	All-blocks break	Batch break setting for all blocks			
		Designated block break	Max. of 64-block designations			
		Designated step break	Max. of 64-step designations			
	Continue		Number of cycles	1 to 255 times		
		All blocks continue	All blocks designation			
		Designated block continue	1 block designation			
		Designated step continue	1 point designation at specified step			
	Forced execution	1 step continue from designated step	1 point designation at specified step			
		Forced block execution	1 block designation			
		Forced step execution	1 point designation at specified step			
		Forced 1 step execution for designated block	1 point designation at specified step			
		Forced block end	1 block designation			
Forced step end	1 point designation at specified step					
Step trace function	Trace memory capacity	Max. of 48k bytes for all blocks; 1 to 48k bytes per block (1k byte units)				
	Trace memory capacity after trigger	From 128 bytes to capacity setting of block				
	Block designation	Max. of 12 blocks				
	Trigger step	1 step per block				
	Execution conditions	Per scan or per designated time				
Step transition watchdog timer function		Equipped with 10 timers				

REMARK

The relationship between the CPU memory's program capacity and the number of files is shown below.

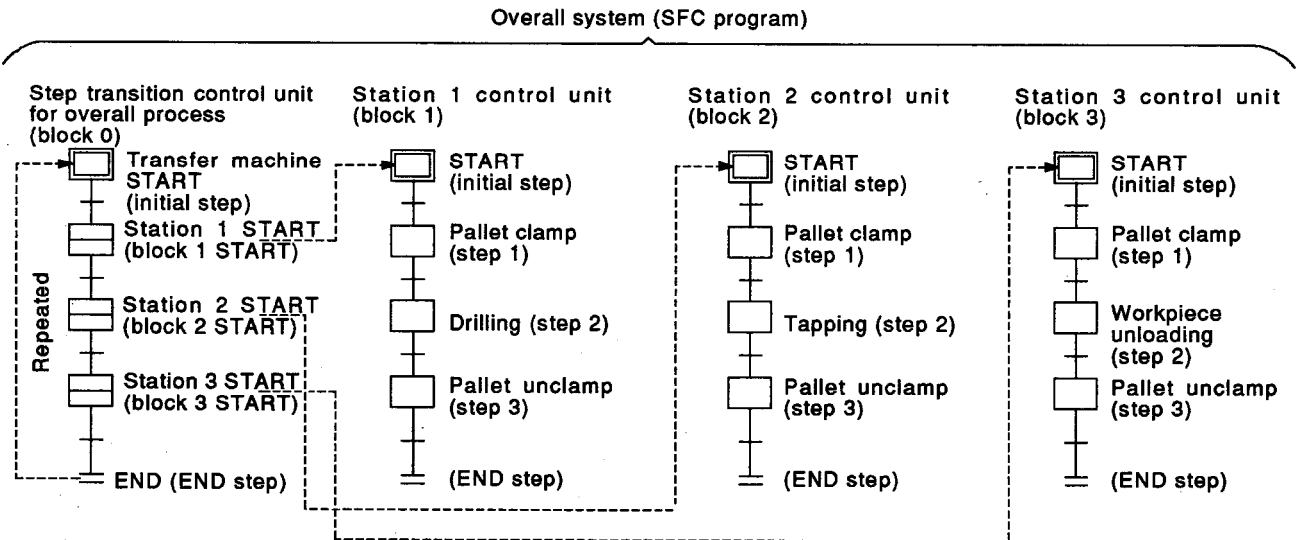
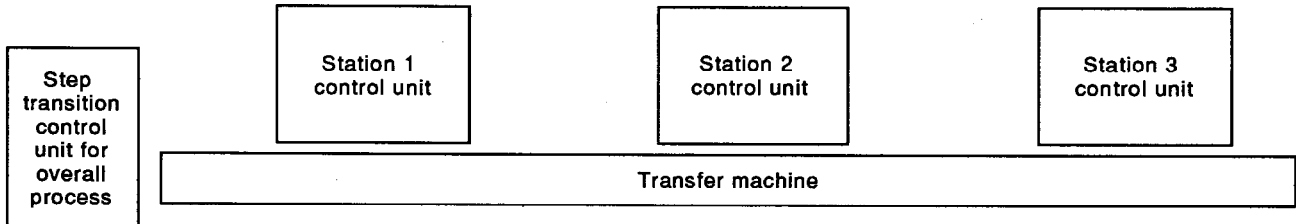


- The SFC program can execute only 1 file. To execute an SFC program which is in the wait status, switch the SFC program being scanned to the wait status, then scan the program in question.

6.2 SFC Features

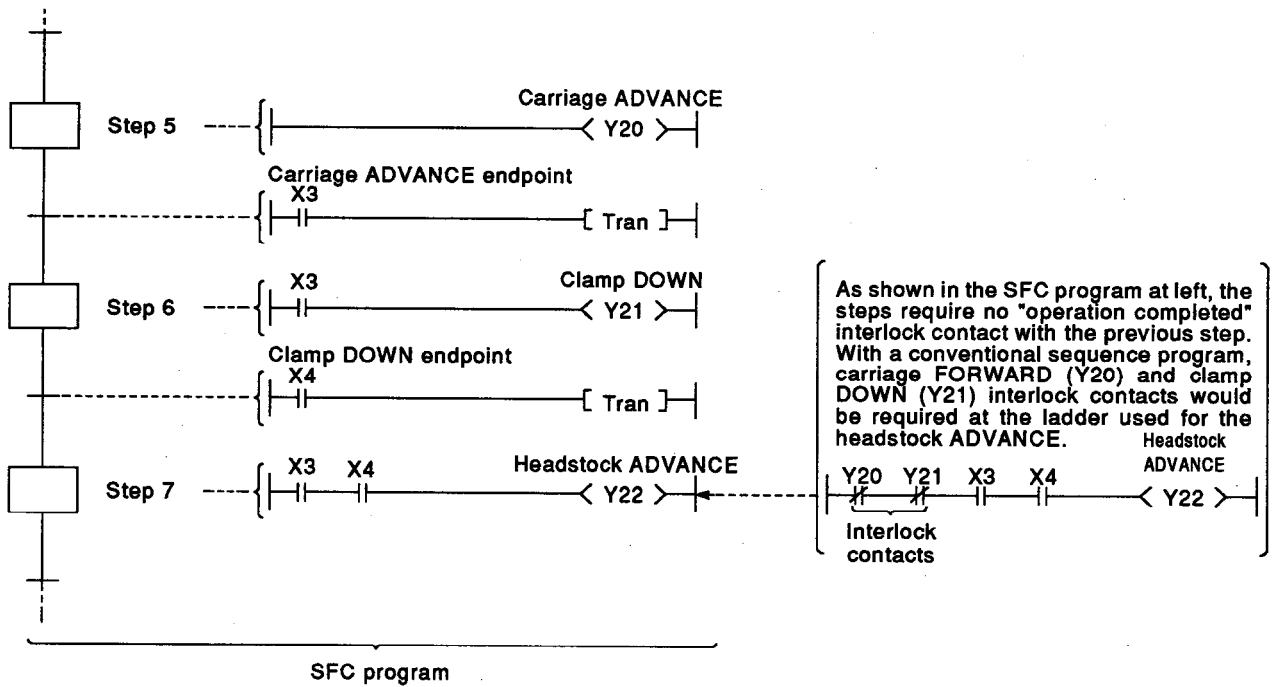
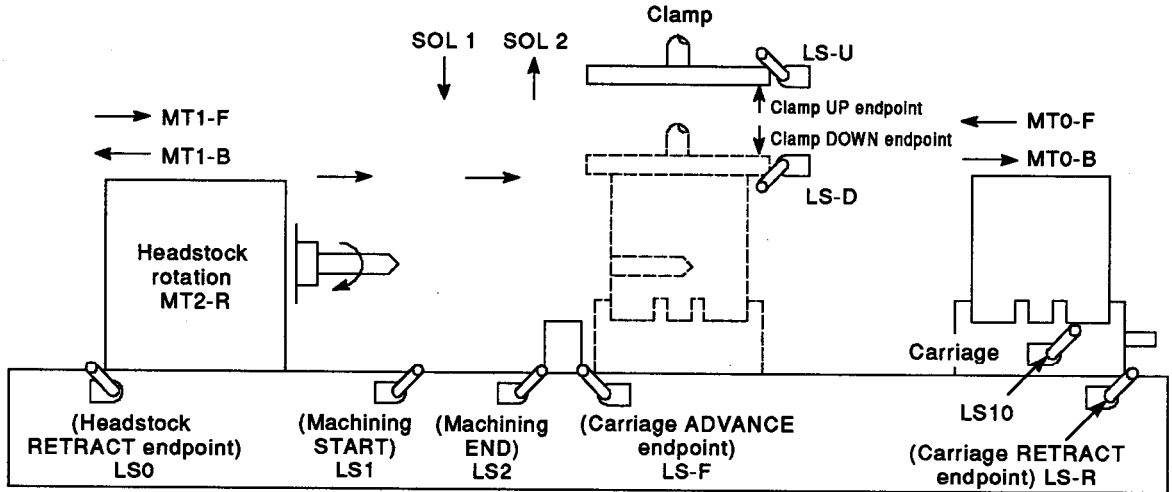
(1) Easy to design and maintain systems

Because control of the overall system and each station, as well as the machines themselves, corresponds on a one-to-one basis with the blocks and steps of the SFC program, systems can be designed and maintained even by those with relatively little sequence program experience. Moreover, programs designed by other programmers using this format are much easier to decode than sequence programs.



(2) Requires no complex interlock circuitry

Interlock circuits are used only in the operation output programs for each step. Because interlocks between steps are not required, it is not necessary to consider interlocks with regard to the overall system.

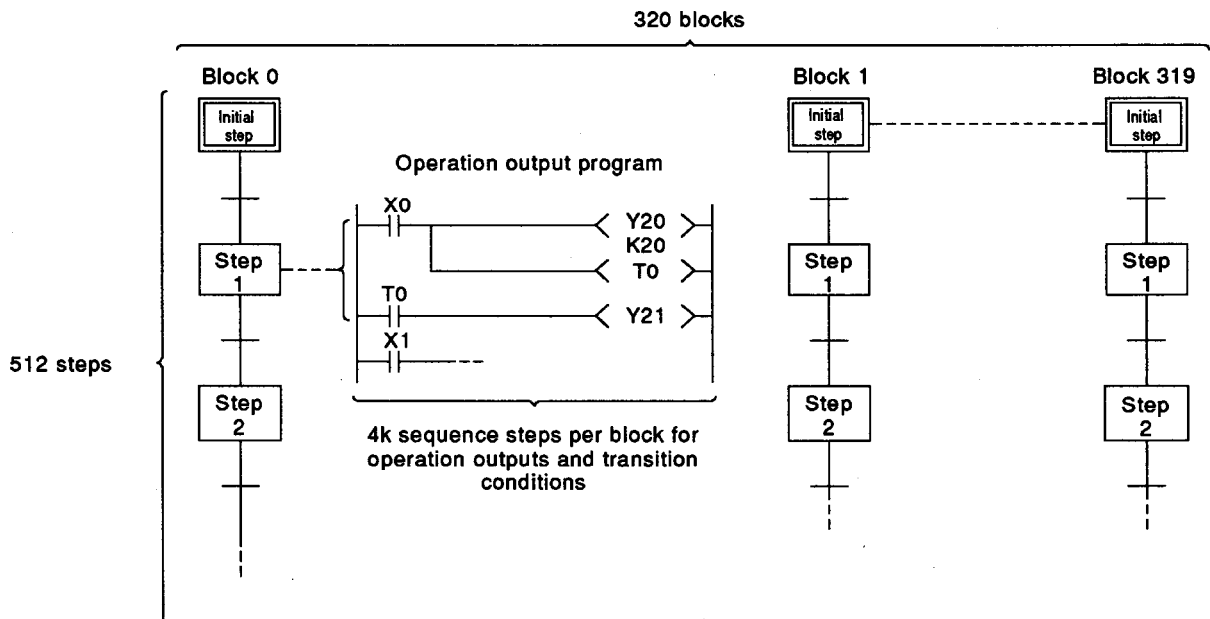


As shown in the SFC program at left, the steps require no "operation completed" interlock contact with the previous step. With a conventional sequence program, carriage FORWARD (Y20) and clamp DOWN (Y21) interlock contacts would be required at the ladder used for the headstock ADVANCE.

- (3) Block and step configurations can easily be changed for new control applications

A total of 320 blocks can be used in an SFC program, with 512 steps in each block. A total of 4k sequence steps can be created in each block of the ladder diagram programs for operation outputs and transition conditions.

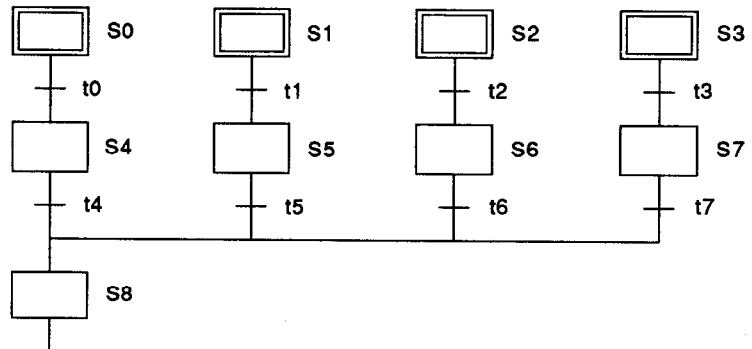
Reduced tact times, as well as easier debugging and trial run operations are possible by dividing the blocks and steps so as to obtain the optimum configuration for system-of-units used for machine operation.



(4) Creation of multiple initial steps is possible

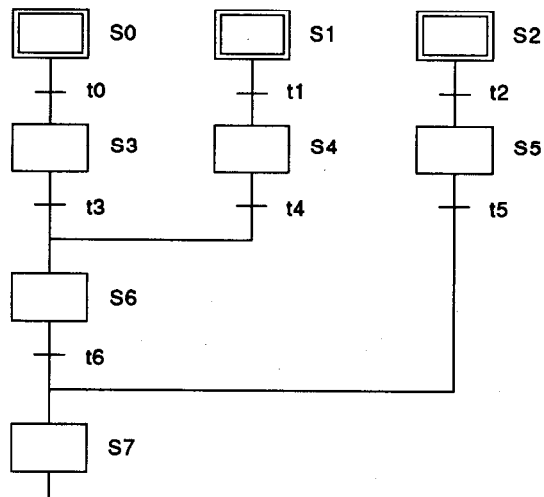
Multiple processes can easily be executed and combined. Initial steps are linked using a "selection coupling" format. When multiple initial steps (S0 to S3) are active, the step where the transition condition (t4 to t7) immediately prior to the selected coupling is satisfied becomes inactive, and a transition to the next step occurs. Moreover, when the transition condition immediately prior to an active step is satisfied, the next step is executed in accordance with the parameter settings.

- Wait : Transition to the next step occurs after waiting for the next step to become inactive.
- Transfer : Transition to the next step occurs even if the next step is active.
- Pause : An error occurs if the next step is active.



REMARK

Linked steps can also be changed at each initial step.



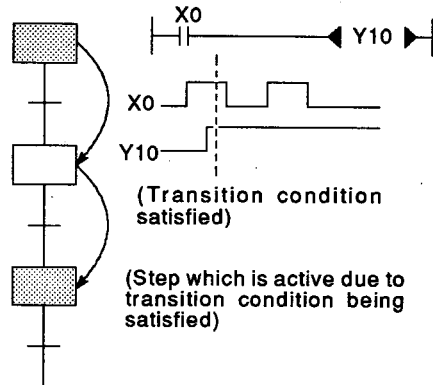
(5) Program design is easy due to a wealth of step attributes

A variety of step attributes can be assigned to each step according to the control operation, greatly simplifying the program design procedure.

For steps without step attributes, when the transition condition is satisfied, the coil in the step currently being executed is turned OFF and the step is deactivated.

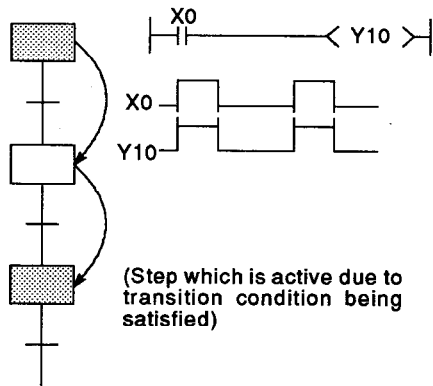
(a) Types of HOLD steps, and their operations

1) Coil HOLD step ([SC])



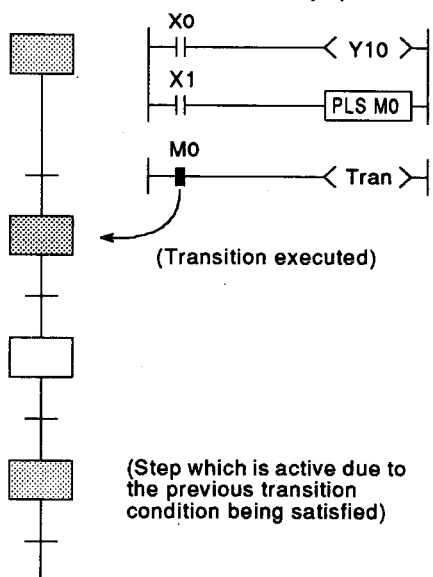
- When the transition condition is satisfied, the coil output status is maintained.
- Convenient for maintaining an output until the block in question is completed (hydraulic motor output, pass confirmation signal, etc.).

2) Operation HOLD step (no transition check)([SE])



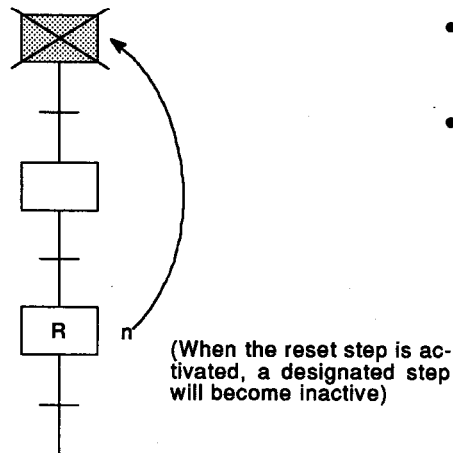
- Operation output processing continues even after a step transition occurs, and coil output (Y10) ON/OFF switching occurs in accordance with the interlock condition (X0) ON/OFF status.
- Transition will not occur if the transition condition is satisfied again.
- Convenient for repeating the same operation (cylinder advance/retract, etc.) while the relevant block is active.

3) Operation HOLD step (with transition check)([ST])



- Operation output processing continues even after a step transition occurs, and coil output (Y10) ON/OFF switching occurs in accordance with the interlock condition (X0) ON/OFF status.
- When the transition condition is again satisfied, the transition is executed, and the next step is activated.
- Operation output processing is executed at the reactivated next step. When the transition condition is satisfied, transition occurs, and the step is deactivated.
- Convenient for outputs where there is an interlock with the next operation, for example where machining is started on completion of a repeated operation (workpiece transport, etc.).

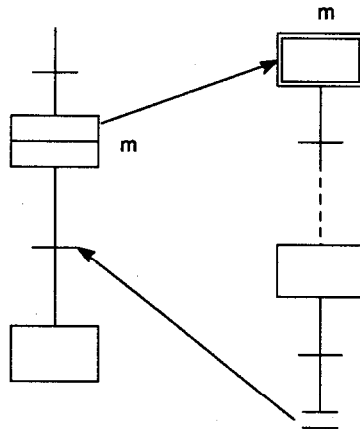
(b) Reset step([R])



- A reset request can be designated for the HOLD step, deactivating the step in question.
- Convenient when a HOLD status becomes unnecessary for machine control, or on selective branching to a manual ladder occurs after an error detection, etc.

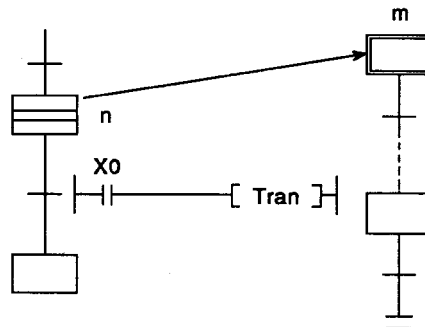
(c) Types of block START steps, and their operations

1) Block START step (with END check)(≡ m)



- In the same manner as for a subroutine CALL-RET, a START source block transition will not occur until the end of the START destination block is reached.
- Convenient for starting the same block several times, or to use several blocks together, etc.
- A convenient way to return to the START source block and proceed to the next process block when a given process is completed in a processing line, for example.

2) Block START step (Without END check)(≡ m)



- Even if the START destination block is active, a START source block transition will occur if the transition conditions for the block START step are satisfied. At such times, processing of the START destination block will be continued to the block END.
- The START destination block can be controlled independently and asynchronously with the START source block.

- (6) A given function can be controlled in a variety of ways according to the application in question

Block functions such as START, END, temporary stop, restart, and forced activation and ending of specified steps can be controlled by SFC diagram symbols, SFC control instructions, or by SFC information registers.

- Control by SFC diagram symbols:
Convenient for control of automatic operations with easy sequential control.
- Control by SFC instructions:
Enables requests from program files other than the SFC, and is convenient for error processing, for example after emergency stops, and interrupt control.
- Control by SFC information registers:
Enables control of SFC peripheral devices, and is convenient for partial operations such as debugging or trial runs.

Functions which can be controlled by these 3 methods are shown below.

Function	Control Method		
	SFC Diagram	SFC Control Instructions	SFC Information Registers
Block START (with END wait)	□m	—	—
Block START (without END wait)	▣m	SET BLm	Block START/END bit ON
Block END	⊥	RST BLm	Block START/END bit OFF
Block STOP	—	PAUSE BLm	Block STOP/RESTART bit ON
Restart stopped block	—	RSTART BLm	Block STOP/RESTART bit OFF
Forced step activation	—	SET Sn SCHG Kn	—
Forced step END	Ⓜn	RST Sn SCHG Kn	—

- (a) In cases where the same function can be executed by a number of methods, the first control method which has been designated by the request output to the block or step in question will be the effective control method.
- (b) Functions controlled by a given control method can be canceled by another control method.

Example: For block START

An active block which was started by the SFC diagram (□m) method can be ended (forced end) by an SFC control instruction (RST BLm), or by switching the SFC information register's block START/END bit OFF.

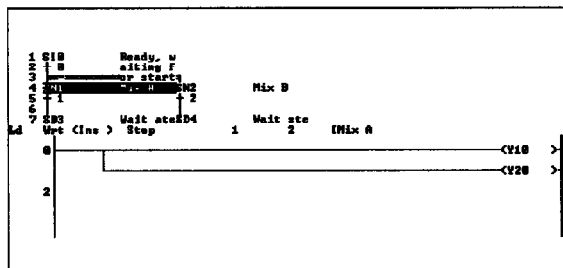
(7) A sophisticated edit function simplifies editing operations

A same-screen SFC diagram, operation output, and transition condition ladder display features a zoom function which can split the screen 4 ways (right/left/upper/lower) to simplify program cut-and-paste operations.

Moreover, advanced program edit functions such as the SFC diagram or device search function, etc., make program creation and editing operations quick and easy.

(8) Displays with comments for easy understanding

Comments (two-byte one-byte characters) can be entered at each step and transition condition item. Up to 24 characters (8 characters x 3 lines = 24 characters) can be entered when one-byte characters are used.

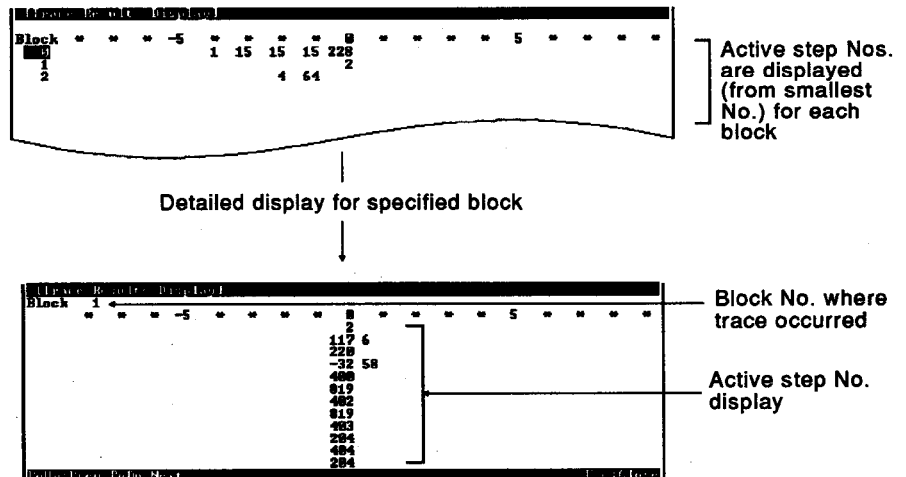


(9) An automatic scrolling functions enables quick identification of mechanical system trouble spots

Active (execution) blocks and steps, as well as the execution of operation output/transition condition ladders can be monitored by a peripheral device (with automatic scrolling function). This monitor function enables even those with little knowledge of sequence programs to easily identify trouble spots.

(10) Convenient trace function

Blocks can be synchronized and traced, enabling the user to check the operation timing of multiple blocks. Moreover, the trace results display screen can be switched to display the trace result details for each block.


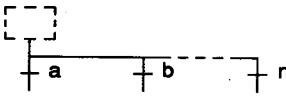

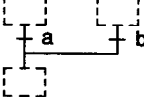
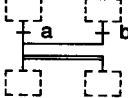
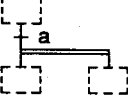
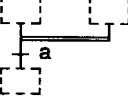
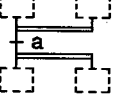
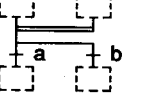
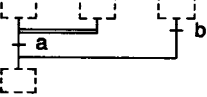
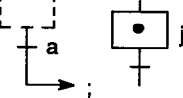



- (11) Program editing is possible using the Q6PU programming unit

Step programs and transition condition programs can easily be revised using the Q6PU. The SFC diagram can be shown in a list format by converting with SW0IVD-GPPQ. The Q6PU unit uses a list format program for program creation/modification operations.

6.3 List of SFC Diagram Symbols

Class	Name		SFC Diagram Symbol	Quantity
Step	Initial step	When step No. is "0"	□ 0	1 of these steps per block
	Dummy initial step		■ 0	
	Coil HOLD initial step		SC 0	
	Operation HOLD step (without transition check) initial step		SE 0	
	Operation HOLD step (with transition check) initial step		ST 0	
	Reset initial step		R 0Sn	
	Initial step		When initial step No. is other than "0"	
	Dummy initial step	⊗ j		
	Coil HOLD initial step	SC j		
	Operation HOLD step (without transition check) initial step	SE j		
	Operation HOLD step (with transition check) initial step	ST j		
	Reset initial step	R jSn		
	Step	Steps other than "initial" step		□ i
	Dummy step		⊗ i	
	Coil HOLD step		SC i	
	Operation HOLD step (without transition check)		SE i	
	Operation HOLD step (with transition check)		ST i	
	Reset step		R iSn	
	Block START step (with END check)		IBLm	
	Block START step (without END check)		iBLm	

Class	Name	SFC Diagram Symbol	Quantity
Transition	Serial transition		
	Selective branching		
	Selective branching - parallel branching		
	Selection coupling		
	Selection coupling - parallel branching		
	Parallel branching		
	Parallel coupling		
	Parallel coupling - parallel branching		
	Parallel coupling - selective branching		
	Parallel coupling - selective coupling		
Jump			
Block END	Block END		

7. MELSECNET/10 NETWORK SYSTEM

7.1 Performance Specifications

7.1.1 Performance specifications for a PC-to-PC network

Performance Specifications for a PC-to-PC Network

Item		Optical Loop System (AJ71QLP21(S))		Coaxial Bus System (AJ71QBR11)	
Maximum number of link points per network	X/Y	8192 points			
	B	8192 points			
	W	8192 points			
Maximum number of link points per station		$\left\{ \frac{Y+B}{8} + (2 \times W) \right\} \leq 2000$ bytes			
Communication speed		10 Mbps (corresponds to 20 Mbps in multiplex transmission)		10 Mbps	
Communication method		Token ring method		Token bus method	
Synchronization method		Frame synchronous method			
Sign format		NRZI sign (Non Return to Zero Inverted)		Manchester sign	
Transmission channel type		Duplex loop		Single bus	
Transmission format		Conforms to HDLC (frame format)			
Maximum network number		239 (total with remote I/O networks)			
Maximum group number		9			
Number of stations connected in 1 network		64 stations (Control stations: 1 Normal stations: 63)		32 stations (Control stations: 1 Normal stations: 31)	
Overall distance of 1 network (station-to-station)		30 km (98425 ft)		3C-2V	300 m (984.25 ft) (station-to-station 300 m (984.25 ft)*1)
		$\left[\begin{array}{l} \text{SI cable H type station-to-station:} \\ \quad 300 \text{ m (984.25 ft)} \\ \text{SI cable L type station-to-station:} \\ \quad 500 \text{ m (1640 ft)} \\ \text{QSI cable station-to-station:} \\ \quad 1 \text{ km (3281 ft)} \end{array} \right]$		5C-2V	500 m (1640 ft) (station-to-station 500 m (1640 ft)*1)
				Can be extended up to 2.5 km (8202 ft) by using a repeater unit (A6BR10, A6BR10-DC)	
Error control system		Retry by CRC ($X^{16} + X^{12} + X^5 + 1$) and overtime			
RAS function		<ul style="list-style-type: none"> • Loop-back at error detection and cable break (only optical loop systems) • Host station link circuit check diagnosis function • System down prevention by control station shift • Error detection by special relay, special register • Network monitor, various diagnostic functions 			
Transient transmission		<ul style="list-style-type: none"> • N:N communication (monitoring, program upload/download, etc.) • ZNRD/ZNWR, SEND/RECV, READ/WRITE, REQ instructions 			
Connection cable		SI-200/250 *2	QSI-185/230 *2	Compatible with 3C-2V, 5C-2V cable	
Application connector		2-core optical connector plug CA7003		BNC connector compatible with 3C-2V, 5C-2V cable	
Cable transmission loss		Maximum 12dB/km	Maximum 5.5dB/km	Conforms to JIS C3501	
Current consumption (5 VDC)		0.65 A		0.8 A	
External power supply (24 VDC)		0.2A (only AJ71QLP21S)			
Weight kg (lb)		0.45 (0.99)			
Occupying number of inputs/outputs		32			

*1 There is a limit on the station-to-station cable length for the coaxial bus system, depending on the number of connected stations.

Please refer to the MELSECNET/10 Reference Manual when preparing the cable.

*2 Fiber-optic cable is handled at your nearest Mitsubishi representative.

7. MELSECNET/10 NETWORK SYSTEM

MELSEC-QnA

7.1.2 Performance Specifications for a Remote I/O Network

Performance Specifications for a Remote I/O Network

Item	Optical Loop System		Coaxial Bus System	
	AJ71QLP21(S)	AJ72QLP25	AJ71QBR11	AJ72QBR15
Maximum number of link points per network	X/Y	8192 points		
	B	8192 points		
	W	8192 points		
Maximum number of link points per station	Master station/sub-master station → remote I/O station $\left\{ \frac{Y+B}{8} + (2 \times W) \right\} \leq 1600$ bytes		Remote I/O station → Master station/sub-master station $\left\{ \frac{Y+B}{8} + (2 \times W) \right\} \leq 1600$ bytes	
	Master station → sub-master station $\left\{ \frac{Y+B}{8} + (2 \times W) \right\} \leq 2000$ bytes			
Maximum number input/output points per remote station	—	X+Y ≤ 2048 points	—	X+Y ≤ 2048 points
Communication speed	10 Mbps (corresponds to 20 Mbps in multiplex transmission)		10 Mbps	
Communication method	Token ring method		Token bus method	
Synchronization method	Frame synchronous method			
Sign format	NRZI sign (Non Return to Zero Inverted)		Manchester sign	
Transmission channel type	Duplex loop		Single bus	
Transmission format	Conforms to HDLC (frame format)			
Maximum network number	239 (total with PC-to-PC network)			
Number of stations connected in 1 network	65 stations (Master stations: 1 Remote I/O stations: 64)		33 stations (Master stations: 1 Remote I/O stations: 32)	
Overall distance of 1 network (station-to-station)	30 km (98425 ft)	$\left[\begin{array}{l} \text{SI cable H type station-to-station:} \\ 300 \text{ m (984.25 ft)} \\ \text{SI cable L type station-to-station:} \\ 500 \text{ m (1640 ft)} \\ \text{QSI cable station-to-station:} \\ 1 \text{ km (3281 ft)} \end{array} \right]$	3C-2V	300 m (984.25 ft) (station-to-station 300 m (984.25 ft)*1)
			5C-2V	500 m (1640 ft) (station-to-station 500 m (1640 ft)*1)
		Can be extended up to 2.5 km (8202 ft) by using a repeater unit (A6BR10, A6BR10-DC)		
Error control system	Retry by CRC($X^{16} + X^{12} + X^5 + 1$) and overtime			
RAS function	<ul style="list-style-type: none"> • Loop-back at error detection and cable break (only optical loop systems) • Host station link circuit check diagnosis function • Error detection by special relay, special register • Network monitor, various diagnostic functions 			
Transient transmission	<ul style="list-style-type: none"> • Monitoring, program upload/download by peripheral device • Intelligent special function module can be used • ZNTO/ZNFR instructions 			
Connection cable	SI-200/250 *2	QSI-185/230 *2	Compatible with 3C-2V, 5C-2V cable	
Application connector	2-core optical connector plug CA7003		BNC connector compatible with 3C-2V, 5C-2V cable	
Cable transmission loss	Maximum 12dB/km	Maximum 5.5dB/km	Conforms to JIS C3501	
Current consumption (5 VDC)	0.65 A	0.8 A	0.8 A	0.9 A
Weight kg (lb)	0.45 (0.99)	0.53 (1.17)	0.45 (0.99)	0.6 (1.32)
Occupying number of inputs/outputs	32	—	32	—

*1 There is a limit on the station-to-station cable length for the coaxial bus system, depending on the number of connected stations.

Please refer to the MELSECNET/10 Reference Manual when preparing the cable.

*2 Fiber-optic cable is handled at your nearest Mitsubishi representative.

7.2 Functions

7.2.1 Direct access

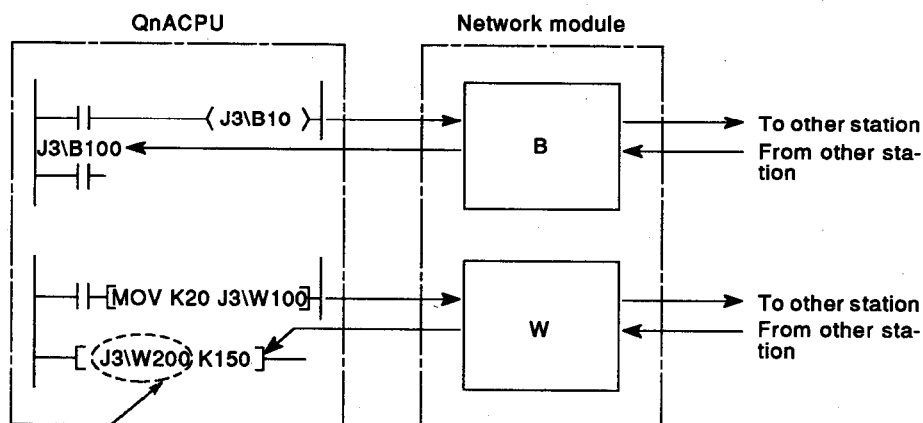
This function allows direct reading/writing of the network module link devices (B, W, X, Y, SB, SW) by the sequence program, and reading/writing of link devices that are not set in the range for link refresh (reading/writing of link devices between the QnACPU and network modules) in the network refresh parameters.

(1) Reduce the link refresh time

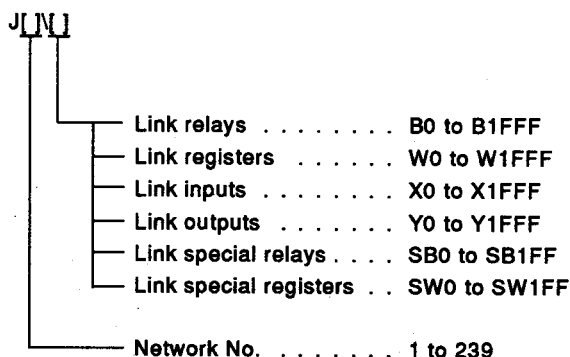
You can reduce the link refresh time by using direct access for link devices which are used infrequently at the host station, and also by removing these link devices from the link refresh range.

(2) Reduce transmission delay time for link devices

Link refresh is conducted in QnACPU END processing, but by using direct access you directly read from/write to the network module when executing instructions, and so can reduce the time from executing direct access instructions until executing link refresh (END instruction).



Direct accessing device designated according to "J[]\ []".

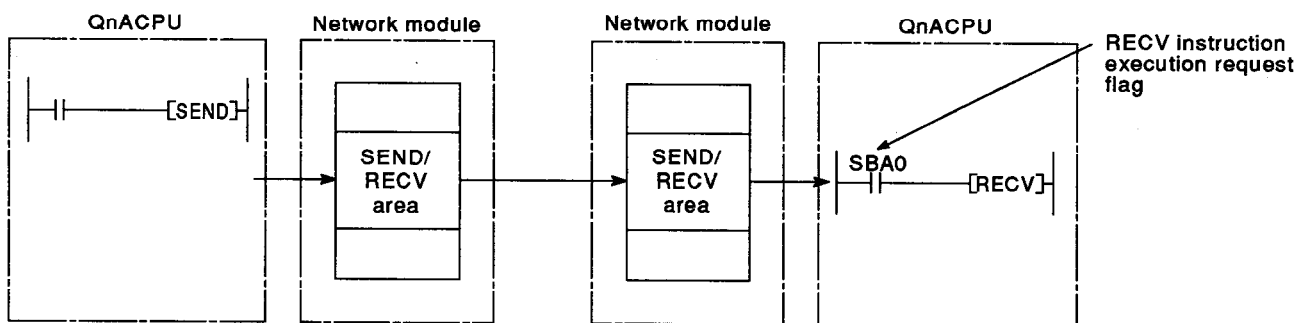


7.2.2 Improved transient transmission

(1) SEND/RECV instructions Data send/receive

Up to a maximum of 480 words of data can be communicated at one time from the sending station (station executing SEND instruction) to the receiving station (station executing RECV instruction).

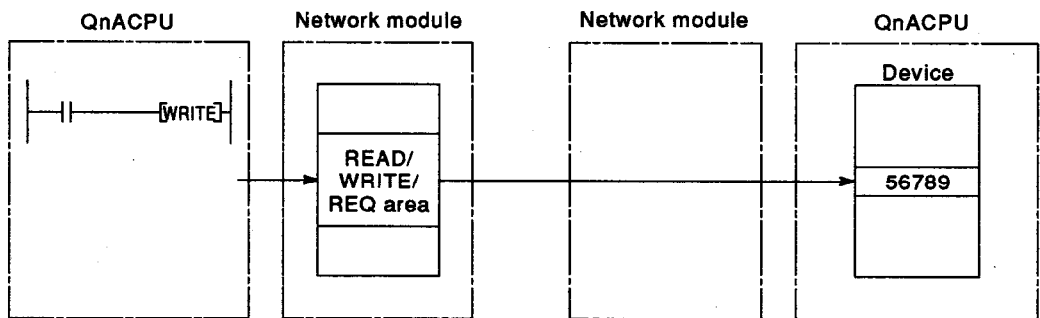
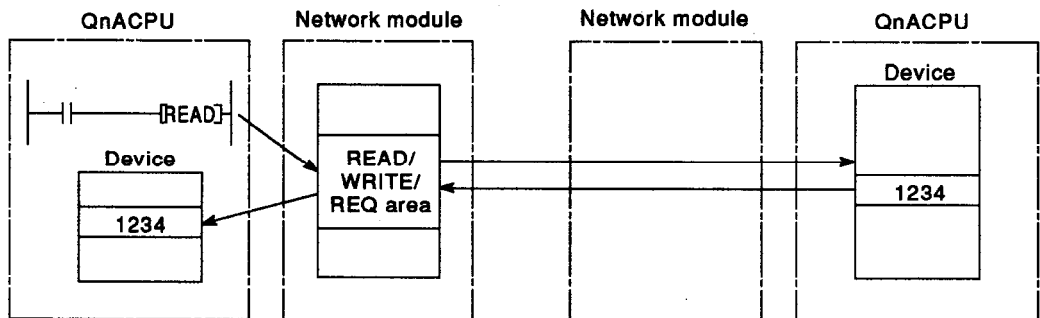
- (a) The SEND instruction is completed when the sending station stores send data in the designated network.
- (b) The receiving station reads receive data in accordance with the RECV instruction when the reading request SB turns ON.



(2) READ/WRITE instructions . . . Reading/writing other station word devices

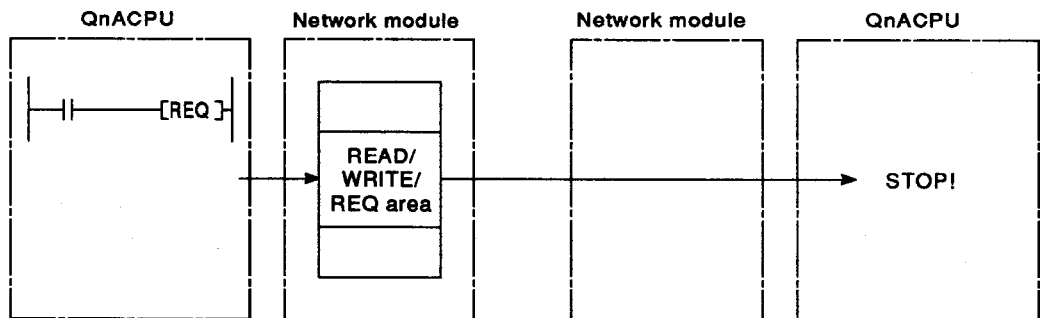
Up to a maximum of 480 words of other station QnACPU word device data can be read/written in one communication.

- (a) Sequence programs for READ/WRITE in the request destination QnACPU are not required.
Other station word device reading/writing can only be done by executing READ/WRITE instructions at the request source.
- (b) The area that can be used by READ/WRITE instructions has 8 channels, and can conduct up to 8 readings/writings of other station word devices at the same time.



(3) REQ instruction Other station transient request

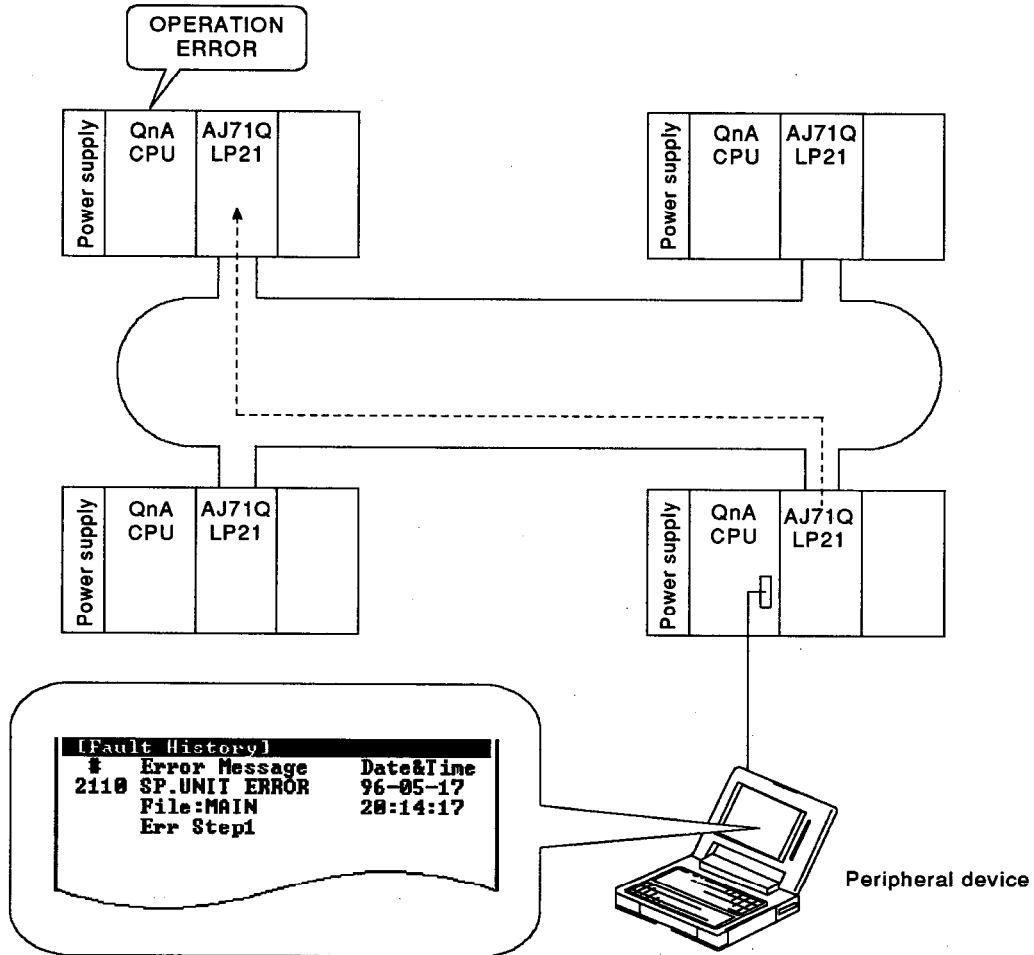
"Remote RUN/STOP" and "Clock data reading/writing" can be performed with respect to other station QnACPU's from the sequence program.



7.2.3 Transient transmission possible even during PC CPU error

Even if an error that stops the PC CPU occurs while the system is operating, network module transient transmission continues, so it is possible to confirm whether the applicable station QnACPU is down.

Even if an error occurs in the control station PC CPU, because the control station is not shifted a temporary downing of the data link is avoided.



The following table shows the cyclic transmission/transient transmission status in each PC CPU status.

PC CPU Status	Cyclic Transmission	Transient Transmission
Continue (eg. Low battery)	Continue	Possible
Stop (e.g. WDT error)		
Control not possible (e.g. RAM error)	Stop	Error returned

7. MELSECNET/10 NETWORK SYSTEM

MELSEC-QnA

7.2.4 Improved convenience of default parameters

Because the QnACPU has default values for the network refresh parameters (automatically allocated by the CPU), you can reduce to a minimum the parameter settings to be made from a peripheral device. For normal stations, network refresh parameter settings are not required if the refresh range is within the range in the following table.

Number of Modules	QnACPU	Priority 1	Priority 2	Priority 3	Priority 4
1					
2					
3					
4					

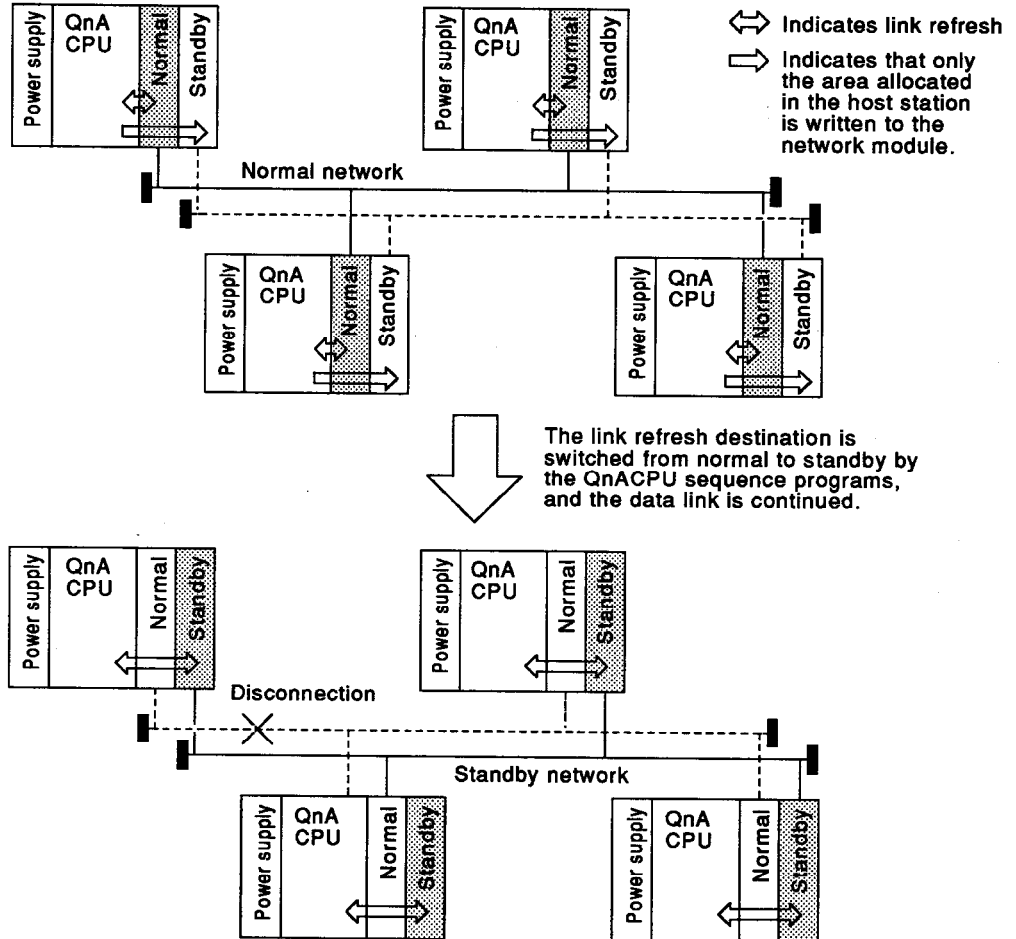
- Priority 1: MELSECNET (II)
 Priority 2: First MELSECNET/10 module
 Priority 3: Second MELSECNET/10 module
 Priority 4: Third MELSECNET/10 module
 Priority 5: Fourth MELSECNET/10 module

7. MELSECNET/10 NETWORK SYSTEM

MELSEC-QnA

7.2.5 Network duplication (PC-to-PC network)

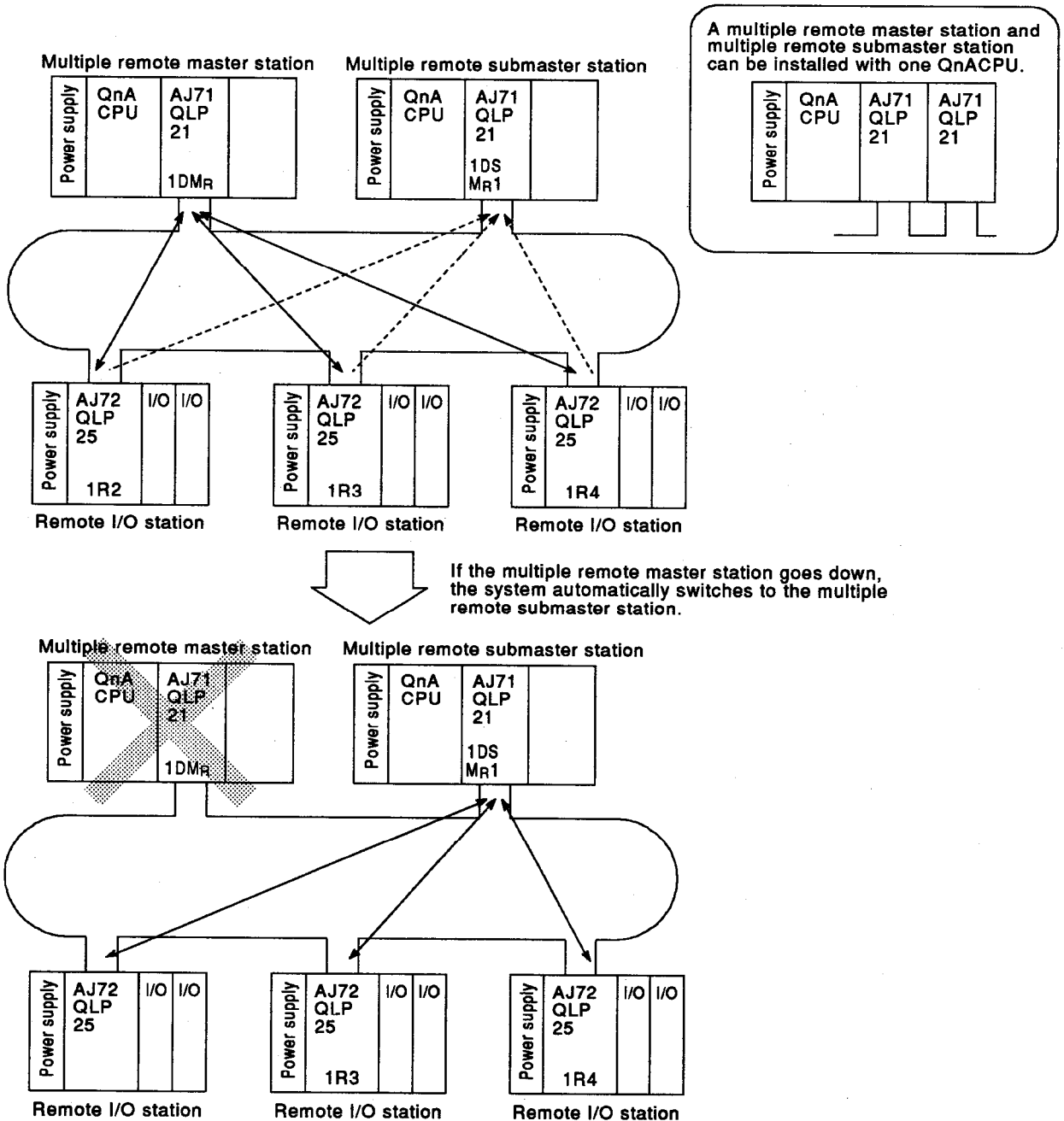
Two network modules are mounted in each PC CPU, so if an error occurs in the normal network because of a module fault, then the system switches to the standby network and continues the data link. (Switching of the link refresh destination between normal and standby is conducted by the sequence program.)



7.2.6 Multiple master stations (remote I/O network)

By providing both multiple remote master stations and multiple remote submaster stations, even if the multiple remote master station goes down, then the multiple remote submaster station can take its place and continue the data link.

- (1) The multiple remote submaster station receives remote I/O station input signals and link register information allocated for special function module use even when the multiple remote master station is operating normally. Accordingly, even after switching to the multiple remote submaster station, control is continued.
- (2) Only set network parameters for the multiple remote master stations. Network parameters are not required for the multiple remote submaster stations.



7. MELSECNET/10 NETWORK SYSTEM

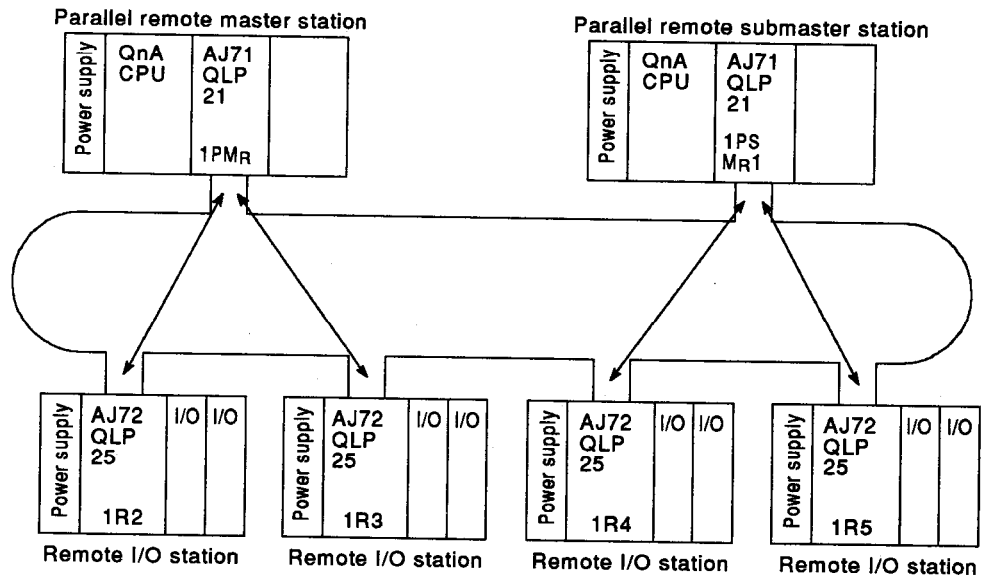
MELSEC-QnA

7.2.7 Parallel master stations (remote I/O network)

As you can connect two remote I/O systems by sharing data link cables, the wiring costs can be reduced.

Network parameters can only be set in parallel remote master stations. Network parameters are not required for parallel remote submaster stations.

In the example system in the following diagram, a parallel remote master station (station 1PM_R) controls remote I/O stations 1R₂ and 1R₃, and a parallel remote submaster station (station 1PS_M) controls stations 1R₄ and 1R₅.

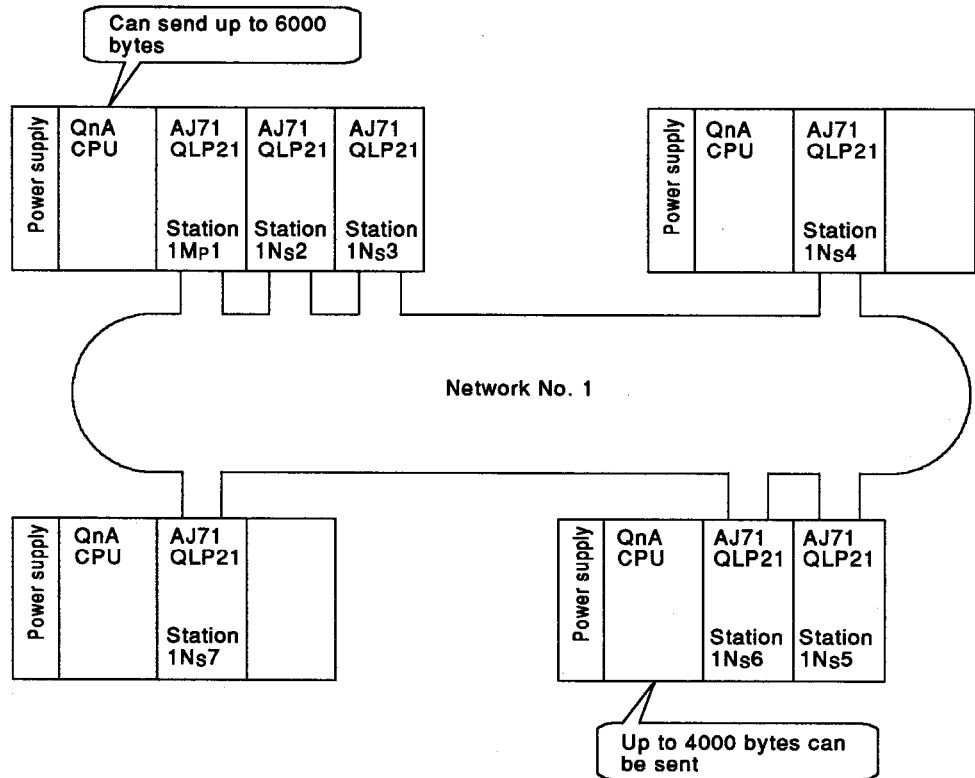


7. MELSECNET/10 NETWORK SYSTEM

MELSEC-QnA

7.2.8 Increasing the number of transmission points by installing multiple modules with the same network No. (PC-to-PC network)

By installing a multiple number of network modules with the same network number in one QnACPU, more than 2000 bytes can be sent.
You can install up to four modules, so a maximum of 8000 bytes (2000 bytes x 4 modules) can be sent.



7. MELSECNET/10 NETWORK SYSTEM

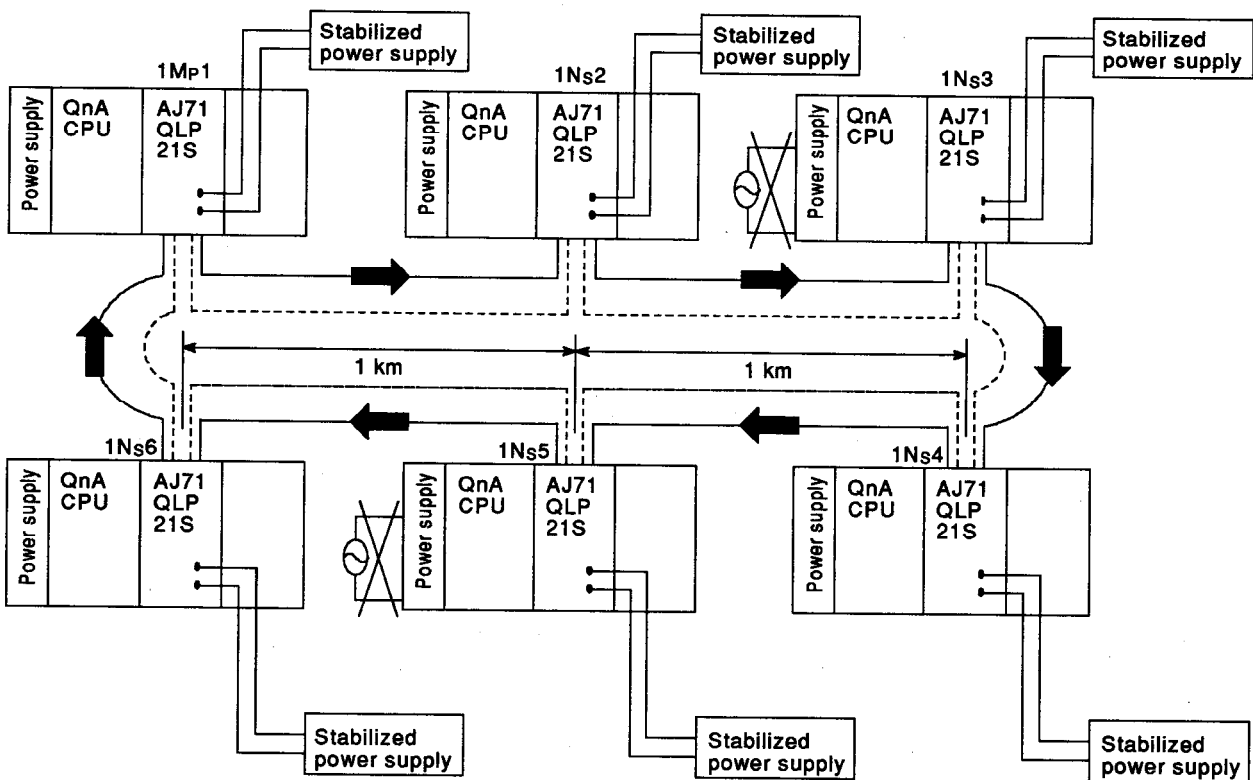
MELSEC-QnA

7.2.9 Preventing loopback by the external power supply (PC-to-PC network: Optical loop system)

By supplying an external power source (24 VDC) to the network module, even if the PC power supply goes OFF the data link circuit is unaffected.

- (1) A normal station which is caught between stations whose PC power supply is OFF can still continue data link
- (2) When the PC power supply goes OFF, even if the station-to-station distance between normal stations exceeds 1 km (3281 ft) (for QSI cables), the data link is continued normally.

For example, in the system below, should the power supply module for stations 1Ns3 and 1Ns5 go OFF, provided an external power supply is supplied to the applicable station network module, then station 1Ns4 continues the data link as normal.

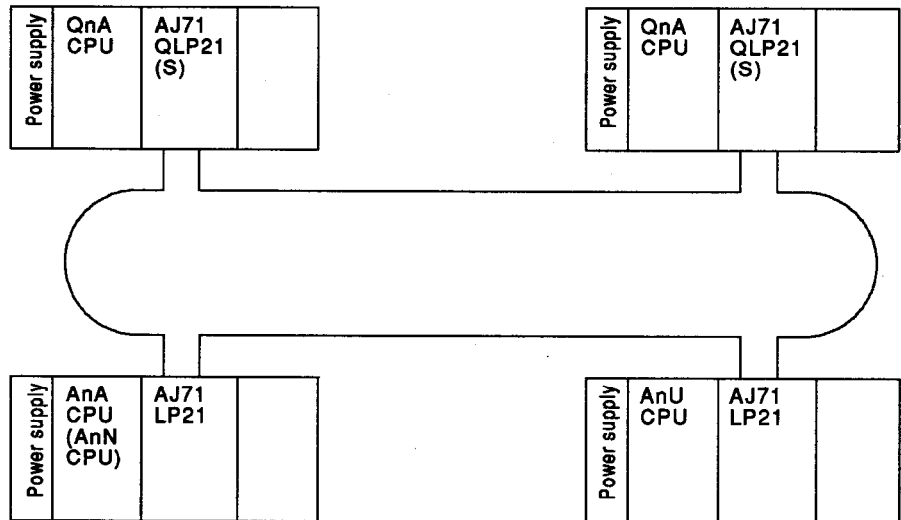


The AJ71QLP21S network module can have an external power supply.

7.3 Compatibility with MELSECNET/10 for AnU

7.3.1 PC-to-PC network

(1) System configuration



(2) Combining CPU modules and network modules

AJ71QLP21(S)/AJ71BR11 (MELSECNET/10 network module for use with QnACPU) cannot be installed with AnUCPU/AnACPU/AnNCPUs. Furthermore, AJ71LP21(S)/AJ71BR11 (network module for use with AnUCPU) cannot be installed with QnACPU.

o: Installation possible x: Installation not possible

CPU Module \ Network Module	AJ71QLP21(S) AJ71QBR11	AJ71LP21 AJ71BR11
	QnACPU	o
AnUCPU AnACPU AnNCPUs	x	o

(3) Combining different control station CPU module types

QnACPU and AnUCPU can be set as control stations. Regardless of which of these two CPU modules you set as the control station, QnACPU/AnUCPU/AnACPU/AnNCPU can be connected as normal stations.

o: Installation possible

Normal Station \ Control Station	QnACPU	AnUCPU	AnACPU AnNCPU
QnACPU	o	o	o
AnUCPU	o	o	o

* AnACPU and AnNCPU cannot be set as control stations.

(4) Combining transient transmission instructions

The following table shows the transient instructions that can be executed by request source and request destination CPU types.

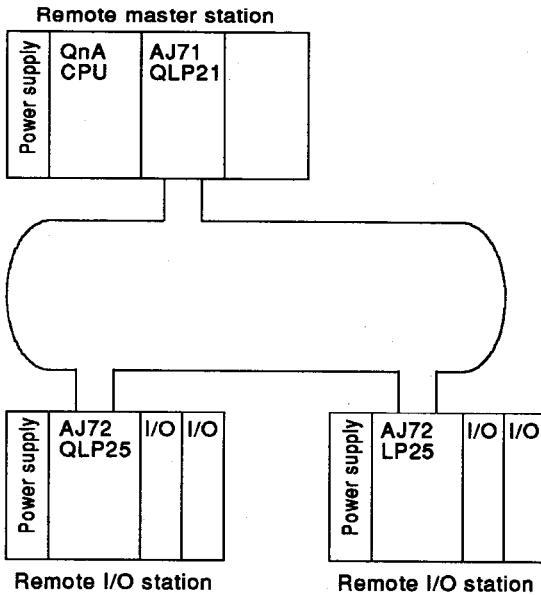
Request Destination \ Request Source	QnACPU	AnUCPU	AnACPU AnNCPU
QnACPU	ZNRD/ZNWR SEND/RECV READ/WRITE REQ	ZNRD/ZNWR	ZNRD/ZNWR
AnUCPU	ZNRD/ZNWR	ZNRD/ZNWR	ZNRD/ZNWR

* There are no instructions to access MELSECNET/10 other stations from AnACPU and AnNCPU.

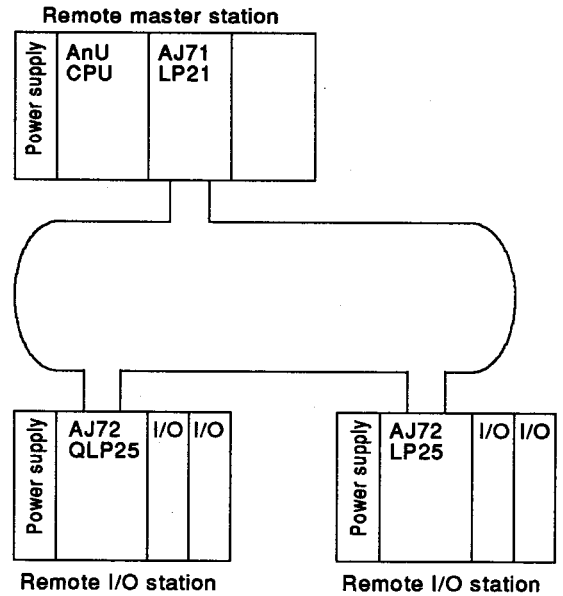
7.3.2 Remote I/O network

(1) System configuration

[When the remote master station is QnACPU]



[When the remote master station is AnUCPU]



(2) Combining CPU modules and network modules

AJ72QLP25/AJ72QBR15 (MELSECNET/10 remote I/O module for use with QnACPU) can be used whether the remote master station is QnACPU or AnUCPU.

Furthermore, AJ71LP25/AJ72BR15 (MELSECNET/10 remote I/O module for use with AnUCPU) can be used whether the remote master station is QnACPU or AnUCPU.

o: Connection possible

Remote Master Station \ Remote I/O Station	AJ72QLP25 AJ72QBR15	AJ72LP25 AJ72BR15
	QnACPU	o
AnUCPU	o	o

(3) Multiple master/parallel master function

The multiple master/parallel master function is only available when the remote master station is QnACPU. AJ72LP25/AJ72BR15 can also be connected to a system operating using the parallel master function, however only the parallel remote master station can execute control. Control from the parallel remote submaster station is not possible.

o: Control possible x: Control not possible

Remote Master Station \ Remote I/O Station	QnACPU			
	Control from Multiple Remote Master Station	Control from Multiple Remote Submaster Station	Control from Parallel Remote Master Station	Control from Parallel Remote Submaster Station
For use with QnA (AJ72QLP25, AJ72QBR15)	o	o	o	o
For use with AnU (AJ72LP25, AJ72BR15)	o	o	o	x

* The multiple master/parallel master function cannot be used when AnUCPU is the remote master station.

8. SERIAL COMMUNICATION MODULE (AJ71QC24)

8.1 Transmission Specifications

The following table shows the AJ71QC24 transmission specifications.

Item	Specification		
	AJ71QC24	AJ71QC24-R2	AJ71QC24-R4
Interface (H/W specification)			
CH1	Conforming to RS-232C (Full-duplex/half-duplex communication)	Conforming to RS-232C (Full-duplex/half-duplex communication)	Conforming to RS-422 (Full-duplex/half-duplex communication)
CH2	Conforming to RS-422/485 (Full-duplex/half-duplex communication)	Conforming to RS-232C (Full-duplex/half-duplex communication)	Conforming to RS-422/485 (Full-duplex/half-duplex communication)
Data communication function communication method	_____		
Dedicated protocol	Half-duplex communication (full-duplex/half-duplex communication when using the on-demand function)		
No-protocol protocol	Full-duplex/half-duplex communication		
Bidirectional protocol	Full-duplex/half-duplex communication		
Synchronization method	Start-stop synchronization (asynchronous)		
Transmission speed (bps)	300, 600, 1200, 2400, 4800, 9600, 19200 *1		
Data format (number of bits)	* Data bit - stop bit format selected by switch.		
Start bit	1		
Data bit	7/8		
Parity bit	1 (yes)/0 (no) * When yes, select odd/even parity. *1		
Stop bit	1/2		
Access cycle	_____		
Dedicated protocol	Processes one request at END processing of installed PC CPU. (Can change to this multiple processing in the PC CPU parameters) Processes one request per link scan when installed at remote station. (The number of scans/number of link scans needed for processing differs depending on the request content.)		
No-protocol protocol	Transmission is executed at each request to send, and reception is possible at all times.		
Bidirectional protocol			
Error detection	_____		
Parity check	Yes/no	* For all the protocols, when "yes" is set, select odd/even parity by DIP switch. *2	
Sum check code	Yes/no	* Select whether for dedicated protocol/bidirectional protocol by switch. * Select whether for no-protocol protocol in the user registration frame.	

(To next page)

8. SERIAL COMMUNICATION MODULE (AJ71QC24)

MELSEC-QnA

(From previous page)

Item		Specification											
Transmission control		RS-232C	RS-422	RS-422/485	DTR/DSR signal control and DC code control are selected by the user.								
	DTR/DSR signal control	Yes	Yes	No									
	RS/CS signal control	Yes	No	No									
	CD signal control	Yes	No	No									
	DC code control	Yes	Yes	Yes									
Number of writes to EEPROM		Maximum 100,000 times for the same area											
Line connection (External device: PC)		AJ71QC24	QJ71QC24-R2	AJ71QC24-R4									
		For independent operation for each interface											
C H 1	Dedicated protocol	1:1	1:1	1:1									
	No-protocol protocol												
	Bidirectional protocol												
C H 2	Dedicated protocol	1:1, 1:n, m:n (Total of n, m+n is a maximum of 32)	1:1	1:1, 1:n, m:n (Total of n, m+n is a maximum of 32)									
	No-protocol protocol												
	Bidirectional protocol												
For operation interlocked between interfaces													
C H 1 • 2	Dedicated protocol	1:n, m:n (Total of n, m+n is a maximum of 32)	Interlocked operation not possible	1:n, m:n (Total of n, m+n is a maximum of 32)									
	No-protocol protocol	1:n (n is a maximum of 32)		1:n (n is a maximum of 32)									
	Bidirectional protocol	Data communication not possible		Data communication not possible									
Transmission distance (overall distance)		<table border="1"> <thead> <tr> <th>Interface</th> <th>Target Interface Setting Transmission Speed (bps)</th> </tr> </thead> <tbody> <tr> <td>RS-232C</td> <td>Maximum 15 m (49.21 ft)</td> </tr> <tr> <td>RS-422</td> <td>Maximum 1200 m (3937 ft)</td> </tr> <tr> <td>RS-42/485</td> <td>Maximum 1200 m (3937 ft)</td> </tr> </tbody> </table>				Interface	Target Interface Setting Transmission Speed (bps)	RS-232C	Maximum 15 m (49.21 ft)	RS-422	Maximum 1200 m (3937 ft)	RS-42/485	Maximum 1200 m (3937 ft)
Interface	Target Interface Setting Transmission Speed (bps)												
RS-232C	Maximum 15 m (49.21 ft)												
RS-422	Maximum 1200 m (3937 ft)												
RS-42/485	Maximum 1200 m (3937 ft)												
Number of occupied inputs/outputs		32 (1 slot occupied)											

*1: The setting range for transmission speed is within 19200 bps for the total of the two interface transmission speeds.

*2: The parity bit complies with horizontal parity.
When using a parity bit, select odd parity/even parity by switch setting.

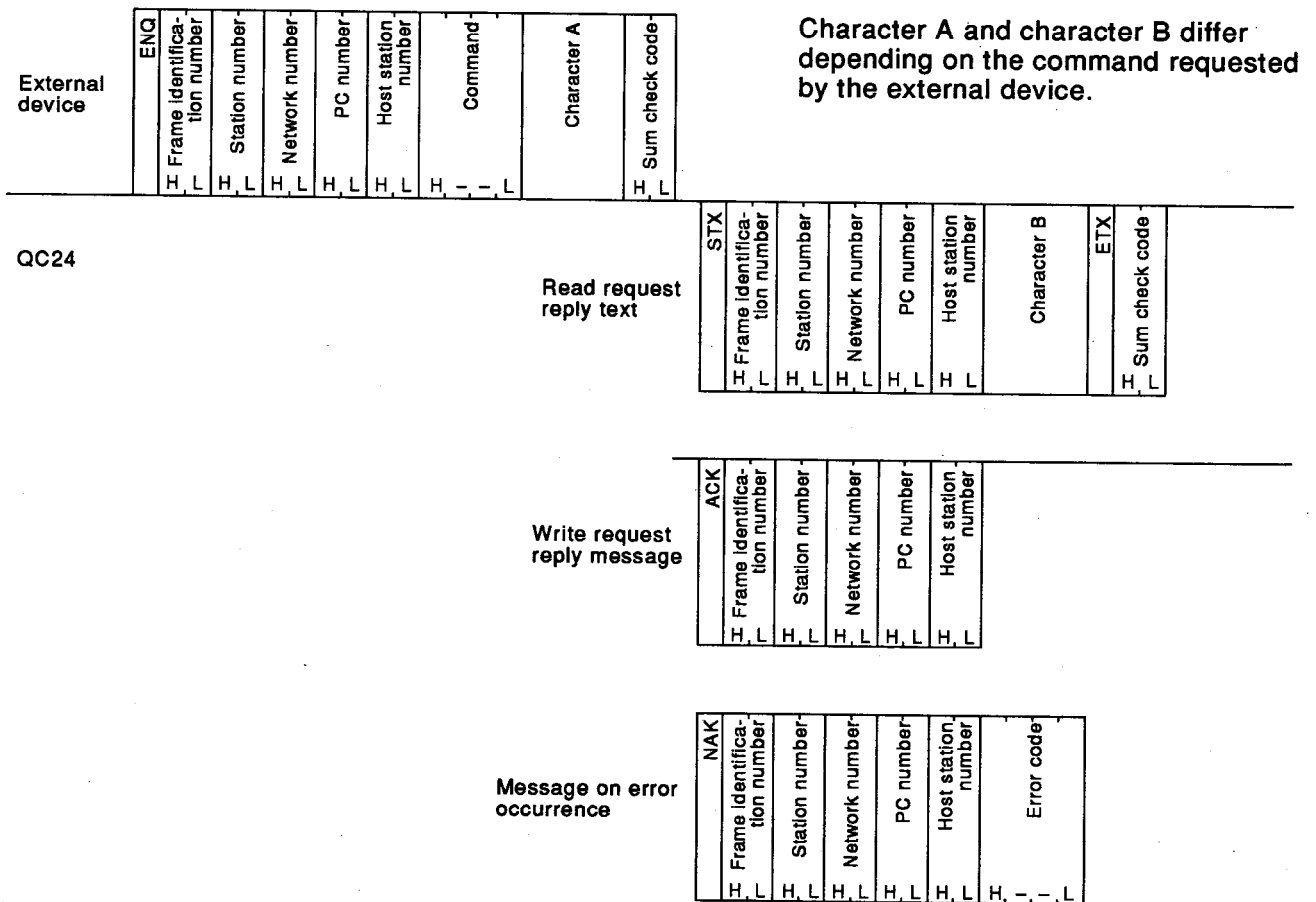
8.2 Features

8.2.1 Dedicated protocol function

Using this function you can transmit command messages in the communication protocol fixed with AJ71QC24, and can read from/write to the QnACPU device memory.

A sequence program for conducting data communications is not required. The following diagram shows the message format when communicating in QnA frame format 1 between an external device and AJ71QC24.

<Example of using QnA frame protocol 1>



(1) Different frame types and applications

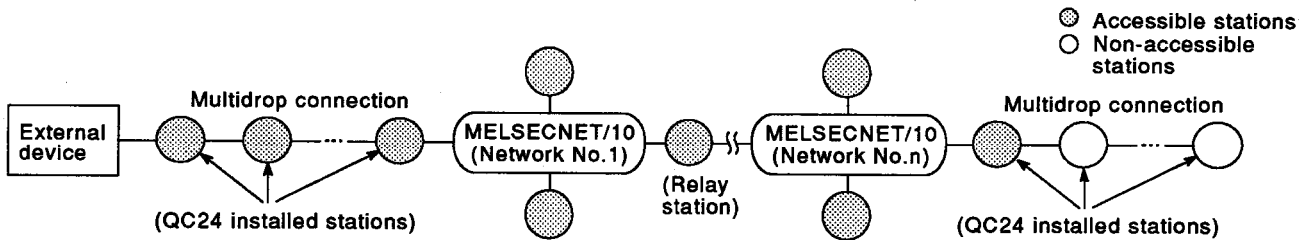
There are three types of frames in dedicated protocol: QnA frames, QnA extension frames, and A-series compatible frames.

(a) QnA frame

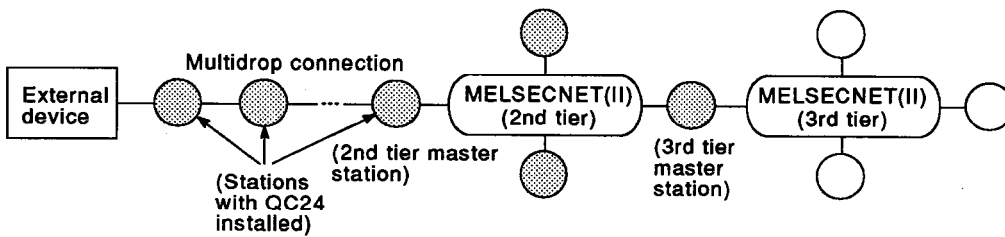
The QnA frame can use all the AJ71QC24 functions.

1) Range for accessing other stations via MELSECNET/10

Using the MELSECNET/10 routing function, other stations of other network Nos. can be accessed. However, stations connected in a multidrop system cannot be accessed via MELSECNET/10.



2) Accessible range for other stations via MELSECNET(II)

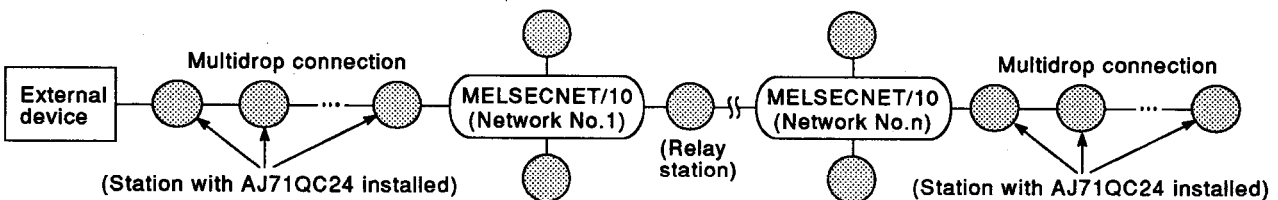


(b) QnA extension frame

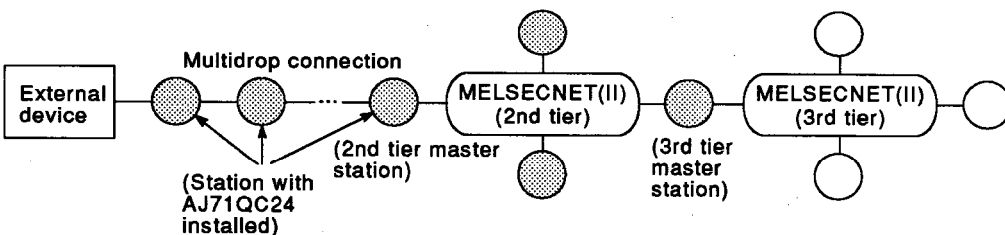
The QnA extension frame can use all of the AJ71QC24 functions. Furthermore, it can connect the GPP function peripheral device to AJ71QC24 by protocol 5.

1) Range for accessing other stations via MELSECNET/10

The QnA extension frame can access stations connected in a multidrop system via MELSECNET/10, which cannot be accessed using the QnA frame.



2) Range for accessing other stations via MELSECNET(II)

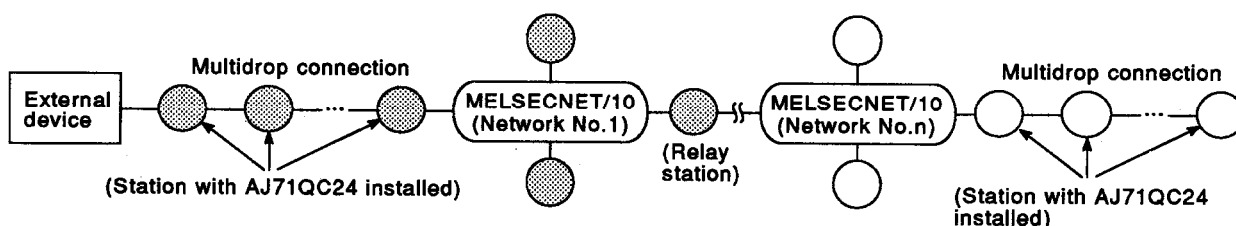


(c) A-series compatible frame

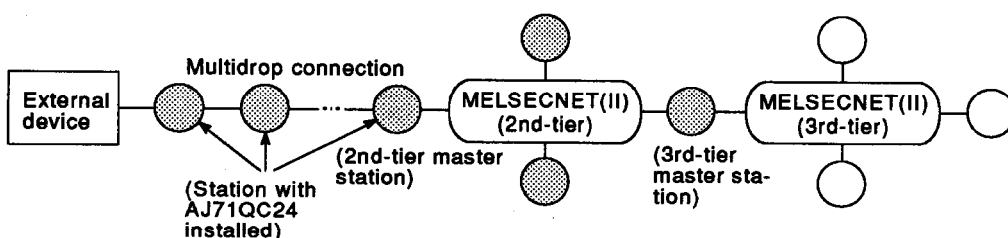
This control protocol is compatible with the dedicated protocol of MELSEC-A series computer link modules (AJ71C24-S[], AJ71UC24, etc.), and allows use of software created for the MELSEC-A series at an external device without alteration. However, you can only read/write with respect to QnACPU within the AnUCPU device range. It is not possible to read from/write to the additional devices available with QnACPU, or perform other functions (reading/writing of programs, remote RUN/STOP, etc.).

1) Range for accessing other stations via MELSECNET/10

When using this frame it is not possible to access other network Nos. through the MELSECNET/10 routing function.



2) Range for accessing other stations via MELSECNET(II)



8. SERIAL COMMUNICATION MODULE (AJ71QC24)

MELSEC-QnA

(2) QnA frame/QnA extension frame commands

The following table shows the commands and functions when accessing PCs from external devices using the QnA frame and QnA extension frame.

Function		Command	Processing Details	Number of Points that can be Processed in One Communication	
Device memory	Batch read	Bit units	0401	Reads bit devices (X, Y, M, etc.) in 1 point units.	3952
		Word units	0401	Reads bit devices (X, Y, M, etc.) in 16 point units. Reads word devices (D, R, T, C, etc.) in 1 point units.	480 words (7680) 480
	Batch write	Bit units	1401	Writes to bit devices (X, Y, M, etc.) in 1 point units.	3952
		Word units	1401	Writes to bit devices (X, Y, M, etc.) in 16 point units. Writes to word devices (D, R, T, C, etc.) in 1 point units.	480 words (7680) 480
	Random read	Word units	0403	Randomly designates and reads bit devices (X, Y, M, etc.) and device numbers in 16 point and 32 points units.	96
				Randomly designates and reads word devices (D, R, T, C, etc.) and device numbers in 1 point and 2 point units.	
	Test (random write)	Bit units	1402	Randomly designates and sets/resets bit devices (X, Y, M, etc.) and device numbers in 1 point units.	96
		Word units	1402	Randomly designates and sets/resets bit devices (X, Y, M, etc.) and device numbers in 16 point units. Randomly designates and writes word devices (D, R, T, C, etc.) and device numbers in 1 point and 2 point units. However, only in 1 point units for other than the QnACPU.	96
	Monitor data registration	Word units	0801	Registers the monitoring bit devices (X, Y, M, etc.) in 16 point units.	176
				Registers the monitoring word devices (D, R, T, C, etc.) in 1, 2 point units. However, only in 1 point units for other than the QnACPU.	
	Monitors	Word units	0802	Monitors devices that registered the monitoring data.	(Registered number)
	Buffer *1 memory	Batch read	0613	Reads the buffer memory data of QC24 connected to external device.	480 words (960)
Batch write		1613	Writes data to buffer memory of QC24 connected to external device.		
PC CPU	Remote RUN	1001	Requests remote RUN with respect to PC CPU.	(1 station)	
	Remote STOP	1002	Requests remote STOP with respect to PC CPU.	(1 station)	
	Remote PAUSE	1003	Requests remote PAUSE with respect to PC CPU.	(1 station)	
	Remote latch clear	1005	Requests remote latch clear with respect to PC CPU when PC CPU is in STOP status.	(1 station)	
	Remote RESET	1006	Requests remote RESET of the PC CPU to cancel a STOP status caused by an error.	(1 station)	
Drive memory	Read memory use status	0205	Reads the cluster status of the drive.	(256 clusters)	
	Memory sort	1207	Rearranges the drive memory contents, increasing the contiguous free area. (Adjustment of file storage positions).	(1 station)	

8. SERIAL COMMUNICATION MODULE (AJ71QC24)

MELSEC-QnA

Function		Command	Processing Details	Number of Points that can be Processed in One Communication	
File	Read file information list	No comment	0201	Reads the file list (file name, last edit date, file size).	(36)
		Comment	0202	Reads the file list (comments attached to file, file name, last edit date, file size).	(16)
		File No. use status	0204	Reads the use status of the file No.	(256)
	Change file information	Change last edit date	1204	Changes the last edit date of the file.	(1)
		Change file name, size	1204	Changes the file name, file size.	(1)
		Batch change	1204	Changes the file name, file size, last edit date.	(1)
	File search		0203	Reads the presence or absence of designated files, file No. file size.	(1)
	Read file content		0206	Reads the file content.	960 bytes
	New registration (register file name)		1202	Reserves the file area of the designated file name.	(File size)
	Write file content	Any data	1203	Write designated data (n bytes) to file.	960 bytes
		Same data (FILL)	1203	Write n bytes of designated data (1 word) to file.	(File size)
	File lock register/cancel		0808	Registers the file lock while accessing a designated file so that the file contents cannot be changed from another station. Alternatively, cancels the registration.	(1)
	File copy		1206	Writes the content of an existing file to a newly registered file. (Copies).	480 bytes
	File delete		1205	Delete file.	(1)
User registration frame ^{*3}	Reads registered data		0610	Reads the registered data of the designated frame No.	80 bytes
	Register data		1610	Registers the first frame/last frame data at data communication in user format message format. (Write).	
	Deletes registered data		1610	Deletes the registered data of designated frame No.	(1)
Global ^{*1}		1618	Turns ON/OFF the global signal (X1A/X1B) for QnACPU where QC24 is installed.	(1 station/all stations)	
On-demand		2101	Sends request to send from PC CPU, and sends data to external device. Can send data corresponding to the largest area size of the contiguous free area of the user free area in the QC24 buffer memory. (Possible when system configuration is 1:1)	(See left)	
Transmission sequence initialization ^{*1} (Only used in binary mode)		1615	Aborts the current processing request, and sets the QC24 to wait for command reception.	(1 station)	
Mode switching ^{*1}		1612	Switches the designated interface operation mode and transmission specifications.	(1 station)	
LED display, error code initializing ^{*1}		1617	Lights up the display error LED, and conducts error code initialization.	(1 station)	
Loopback test ^{*1}		0619	Confirms whether the data communication between QC24 and the external device is being conducted normally. (For checking the connection status and the communication status)	960 bytes (Communication possible only for connected stations)	

*1: Commands can only be executed for a QC24 connected to an external device (including multidrop connected stations), or the QnACPU station where this QC24 is installed.

Commands cannot be executed for other station PC CPUs via data link systems or network systems.

8. SERIAL COMMUNICATION MODULE (AJ71QC24)

MELSEC-QnA

- (3) A-series compatible frame commands, accessible devices, and reference manuals

The following table shows the commands and functions when accessing the PC from an external device using the A-series compatible frame.

(a) Commands that can be used

Function		Command		Processing Details	Number of Points that can be Processed in One Communication		
		Signal	ASCII Code				
Device memory	Batch read	Bit units	BR JR	42H, 52H 4AH, 52H	Reads bit devices (X, Y, M, etc.) in 1 point units.	256	
		Word units	WR QR	57H, 52H 51H, 52H	Reads bit devices (X, Y, M, etc.) in 16 point units. Reads word devices (D, R, T, C, etc.) in 1 point units.	32 words (512) 64	
	Batch write	Bit units	BW JW	42H, 57H 4AH, 57H	Writes to bit devices (X, Y, M, etc.) in 1 point units.	160	
		Word units	WW QW	57H, 57H 51H, 57H	Writes to bit devices (X, Y, M, etc.) in 16 point units. Writes to word devices (D, R, T, C, etc.) in 1 point units.	10 words (160) 64	
	Test (random write)	Bit units	BT JT	42H, 54H 4AH, 54H	Sets/resets bit devices (X, Y, M) in 1 point units, and randomly designates and sets/resets the devices, device numbers.	20	
		Word units	WT QT	57H, 54H 51H, 54H	Sets/resets bit devices (X, Y, M) in 16 point units, and randomly designates and sets/resets the device, device number. Writes to word devices (D, R, T, C, etc.) in 1 point units, and randomly designates and writes to devices, device numbers.	10 words (160) 10	
	Registering monitor data	Bit units	BM JM	42H, 4DH 4AH, 4DH	Registers the monitoring bit devices (X, Y, M, etc.) in 1 point units.	40	
		Word units	WM QM	57H, 4DH 51H, 4DH	Registers the monitoring bit devices (X, Y, M, etc.) in 16 point units. Registers the monitoring word devices (D, R, T, C, etc.) in 1 point units.	20 words (320) 20	
	Monitoring	Bit units	MB MJ	4DH, 42H 4DH, 4AH	Monitors the devices that registered the monitoring data.	—	
		Word units	MN MQ	4DH, 4EH 4DH, 51H			
	Extension file register	Batch read		ER	45H, 52H	Reads extension file registers (R) in 1 point units.	64
		Batch write		EW	45H, 57H	Writes to extension file registers (R) in 1 point units.	64
		Test (random write)		ET	45H, 54H	Randomly designates and writes block No. device numbers to extension file registers (R) in 1 point units.	10
		Register monitoring data		EM	45H, 4DH	Registers monitored extension file registers (R) in 1 point units.	20
Monitor		ME	4DH, 45H	Conducts monitoring of the extension file register (R) that registered monitored data.	—		
Direct read		Word unit	NR	4EH, 52H	Reads extension file registers in 1 point units without considering the block No. by designating serial device numbers.	64	
Direct write		Word unit	NW	4EH, 57H	Writes in 1 point units the extension file register without considering the block No. by designating serial device numbers.	64	

8. SERIAL COMMUNICATION MODULE (AJ71QC24)

MELSEC-QnA

(b) Accessible devices

QnACPU Accessible Devices (when Accessed with A-Series Compatible Frame Commands)							
Category	Device	Device Number (Designated Range)	Decimal/Hexadecimal Expression	Category	Device	Device Number (Designated Range)	Decimal/Hexadecimal Expression
Internal user devices	Input relays	X0 to X1FFF	Hexadecimal	Internal user devices	Timers	Contact	TS0 to TS2047
	Output relays	Y0 to Y1FFF				Coil	TC0 to TC2047
	Internal relays	M0 to M8191	Present value			TN0 to TN2047	
	Latch relays	L0 to L8191	Decimal		Counters	Contact	CS0 to CS1023
	Step relays	S0 to S8191				Coil	CC0 to CC1023
	Link relays	B0 to B1FFF				Present value	CN0 to CN1023
	Annunciators	F0 to F2047	Decimal	Internal system devices	File registers		R0 to R32767
	Data registers	D0 to D8191			Special relays *1		M9000 to M9255
	Link registers	W0 to W1FFF	Hexadecimal		Special registers *2		D9000 to D9255

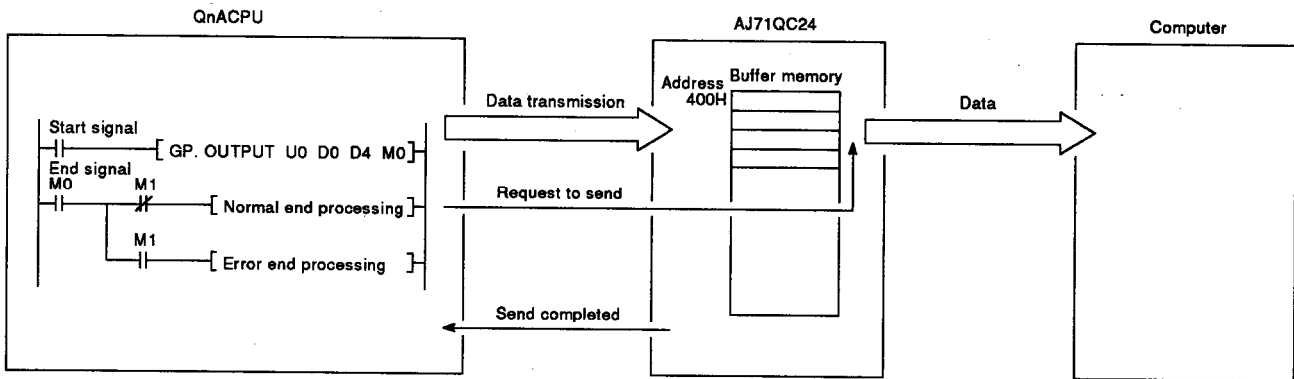
*1: At setting M9000 to M9255 accesses SM1000 to SM1255.

*2: At setting M9000 to M9255 accesses SD1000 to SD1255.

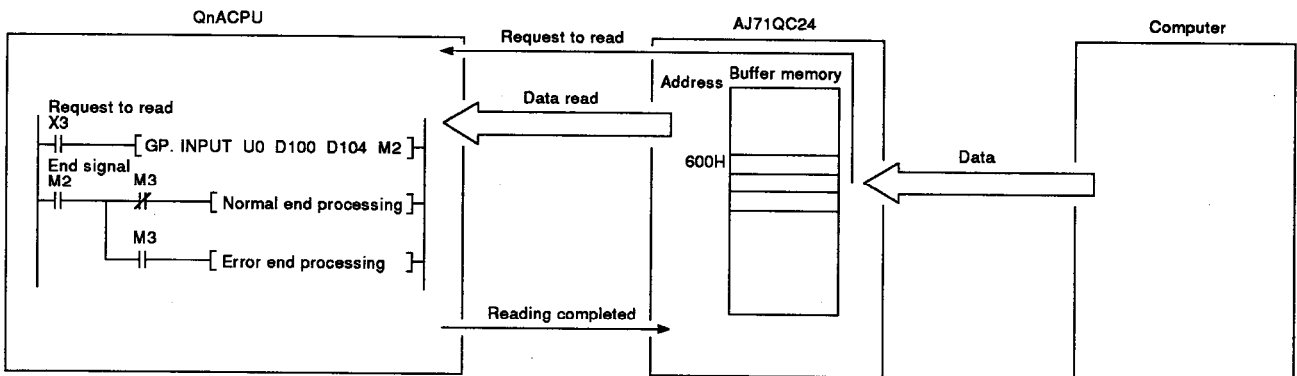
8.2.2 No-protocol protocol function

Data communication between the QnACPU and the external device is conducted in accordance with the communication protocol you select. A sequence program is required for data communication.

(1) An example of a transmission program in no-protocol protocol



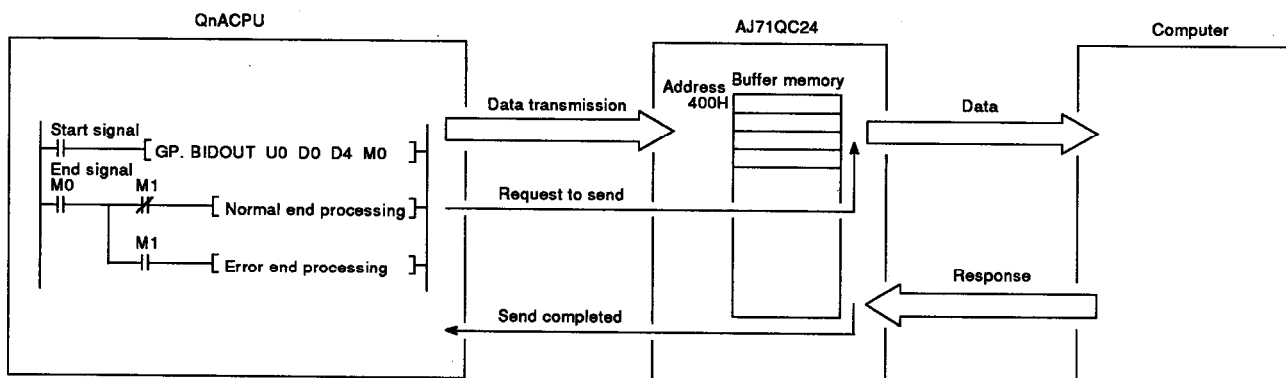
(2) An example of a receive program in no-protocol protocol



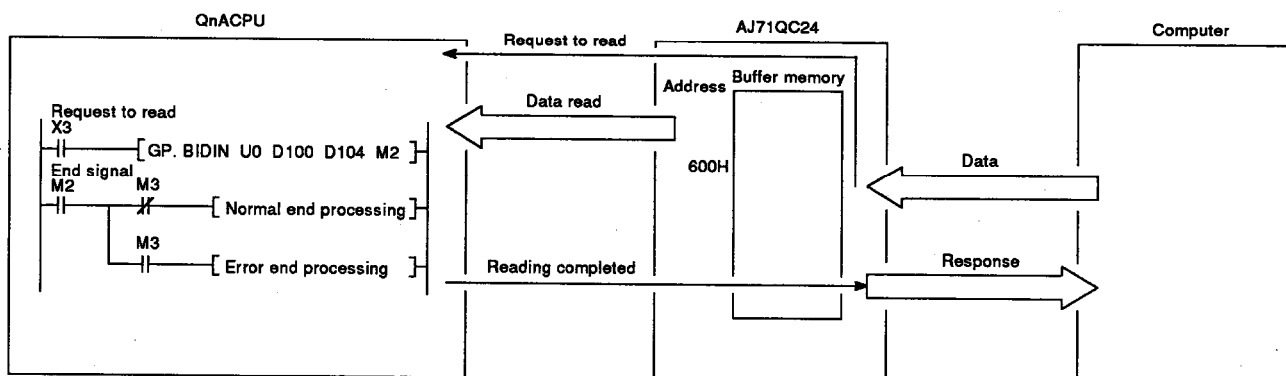
8.2.3 Bidirectional protocol function

Data communication between the QnACPU and the external device is conducted in the communication protocol you select. A sequence program is required for data communication. The bidirectional protocol can confirm whether or not data is received correctly at the reception side.

(1) An example of a transmission program in bidirectional protocol



(2) An example of a receive program in bidirectional protocol



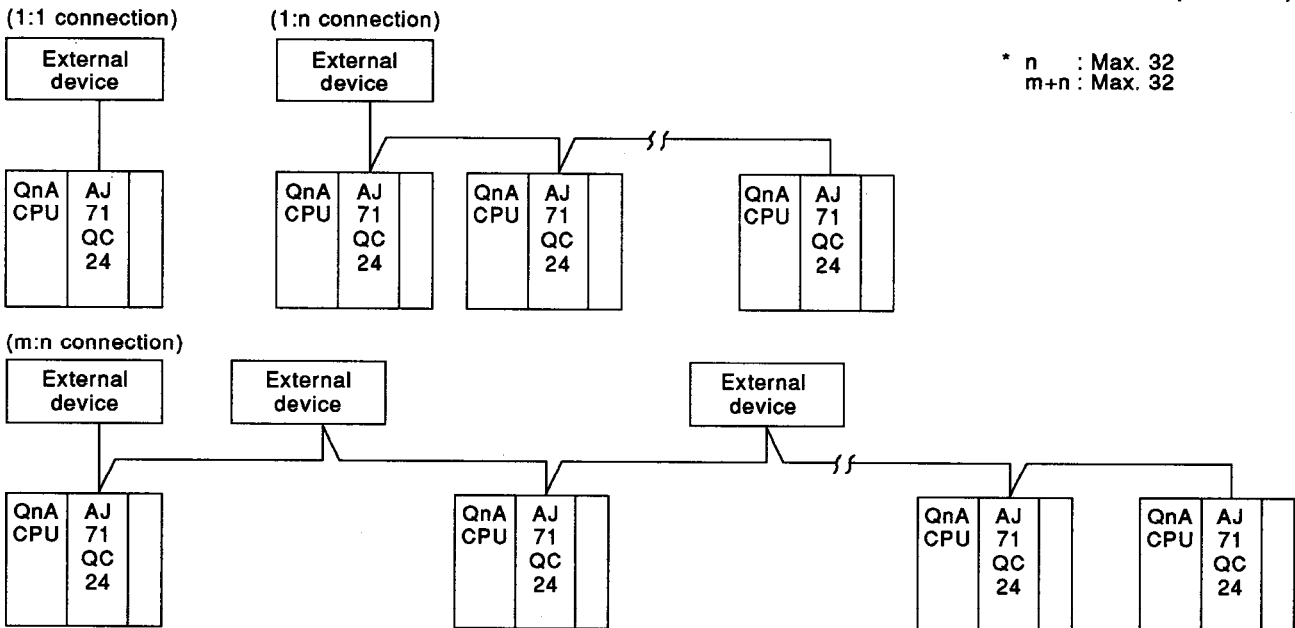
8. SERIAL COMMUNICATION MODULE (AJ71QC24)

MELSEC-QnA

8.2.4 Choose the model that suits your intended application

The serial communication module has two interfaces conforming to either RS-232C, RS-422, or RS-422/485. Three types of modules are provided with different combinations of these interfaces.

- AJ71QC24..... CH1: RS-232C (1:1 connection possible)
CH2: RS-422/485 (1:1, 1:n, m:n connections possible)
- AJ71QC24-R2..... CH1: RS-232C (1:1 connection possible)
CH2: RS-232C (1:1 connection possible)
- AJ71QC24-R4..... CH1: RS-422 (1:1 connection possible)
CH2: RS-422/485 (1:1, 1:n, m:n connections possible)



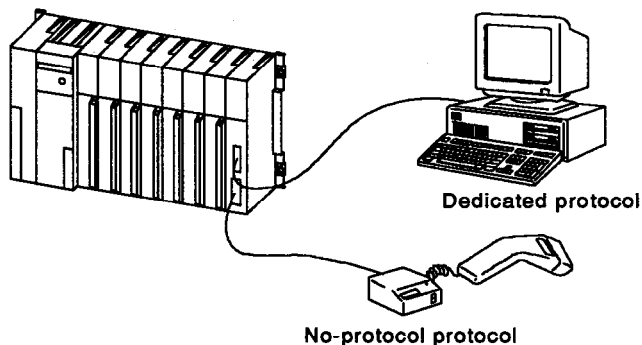
(1) Various data communication system constructions possible

The two AJ71QC24 interfaces can operate both independently and together.

(a) When operating independently

When operating independently, the communication function and transmission specifications (transmission speed, data format, transmission control, etc.) can be set independently for each interface. For example, dedicated protocol can be set for both CH1 and CH2.

<Connection example>

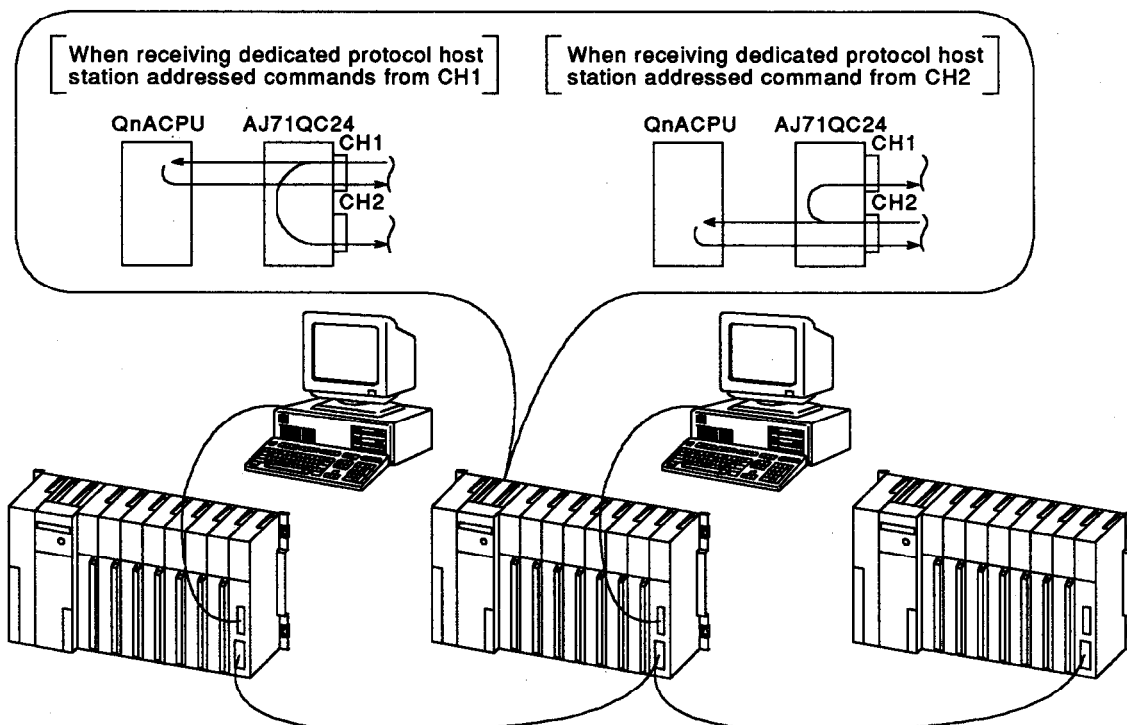


(b) When operating together

When the interfaces operate together, the data received from each interface is processed while sending the received data to the other interface.

When AJ71QC24 and the external device are made to work together in 1:n, m:n connections, AJ71QC24 can replace the RS-232C/RS-422 converter, and can connect AJ71QC24 and the external device through RS-232C.

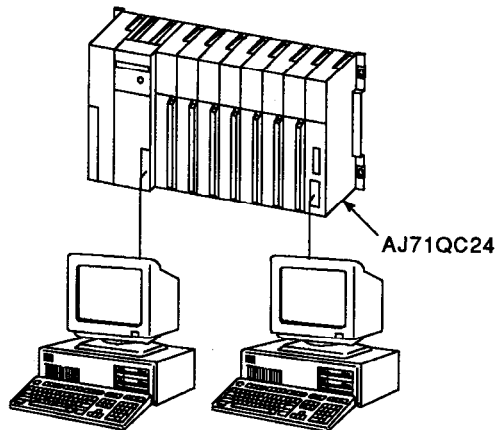
<Connection example>



8.2.5 Other useful functions

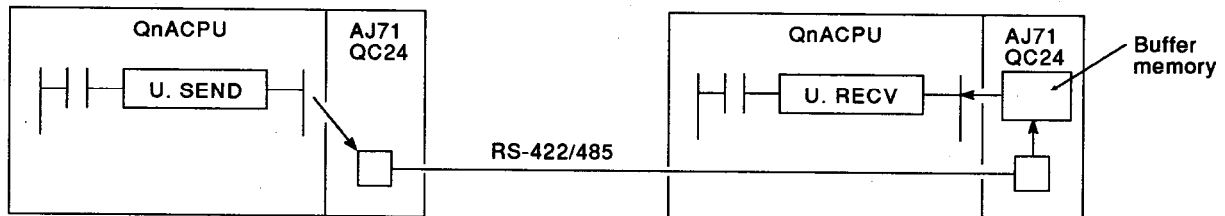
(1) Accessing QnACPU from the GPP function

- (a) By setting the CH1-side RS-232C/RS-422 interface of an AJ71QC24 to QnA extension frame control protocol 5 (binary mode), peripheral devices running GPP function software for use with QnACPU can be connected.
- (b) This allows the same operations as when a peripheral device running GPP function software for use with QnACPU is connected to QnACPU.



(2) Accessing other stations from QnACPU via AJ71QC24

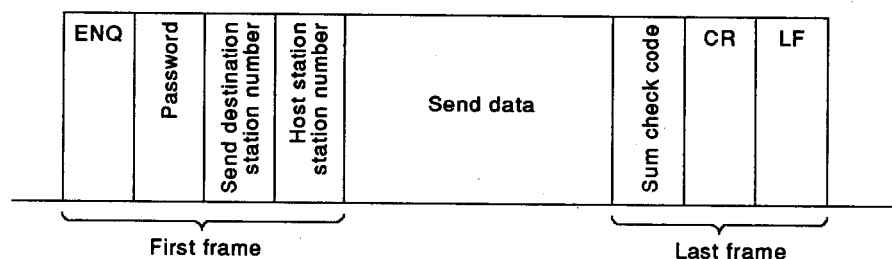
Using sequence program communication instructions (SEND/RECV, READ/WRITE, REQ) you can access QnACPUs at other stations through a AJ71QC24 in a multidrop connection.



(3) Data communication matching the external device message format specifications

Communication can be carried out while adding data in the first and last frames of data sent and received in the dedicated protocol on-demand function and in the no-protocol protocol.

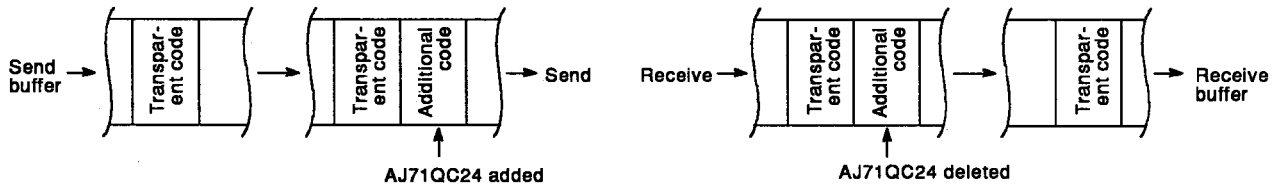
Apart from the fixed data contained in the frames, the host station AJ71QC24 station number can be attached to the first frame, and the sum check code can be attached to the last frame.



(4) Setting transparent code

When communicating data in binary code in the no-protocol protocol or bidirectional protocol, in order to be able to handle 1 byte of external device transmission control data as user data the following codes can be preset in AJ71QC24.

- Transparent code 1 byte of transmission control data
- Additional code 1 byte of send/receive data appended immediately preceding the transparent code.



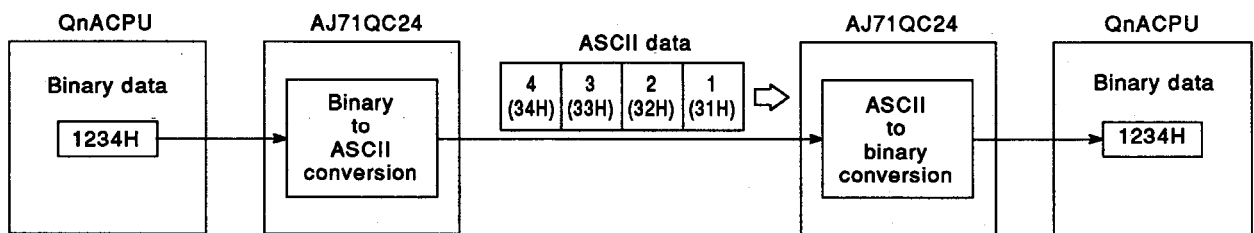
(5) ASCII/binary code selection for communication data

This function allows selection of ASCII/binary code for communicating data matching the specifications of the external device.

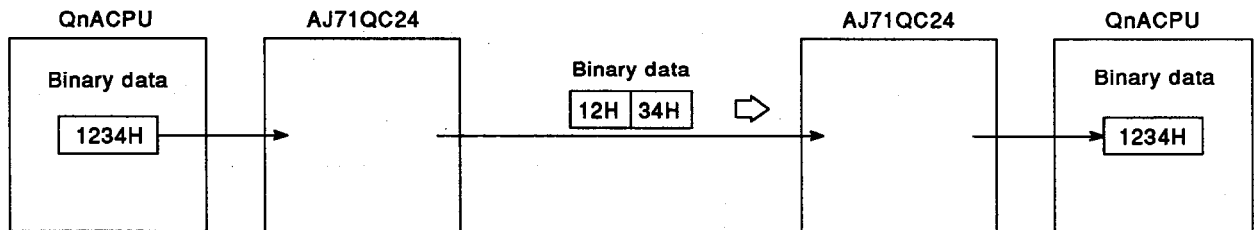
- (a) Any function of dedicated protocol, no-protocol protocol and bidirectional protocol can be selected.
- (b) Even if the communication data are designated as ASCII data, the data handled by the QnACPU are binary data.

The data received by the AJ71QC24 from the external device are converted from ASCII to binary before being transferred to the QnACPU. Furthermore, the data transferred by QnACPU to AJ71QC24 are converted from binary to ASCII code and sent to the external device.

Example 1: When communication data are converted to ASCII data



Example 2: When communication data are converted to binary data



(6) Changing the data communication function/transmission specifications

Data communication functions and transmission specifications can be changed in the sequence program after AJ71QC24 start-up. Furthermore, they also be changed from an external device communicating in dedicated protocol.

At AJ71QC24 start-up, communication is conducted by the data communication function set with the AJ71QC24 switches and the transmission specifications.

(7) Registering user-registered frames and system settings in EEPROM

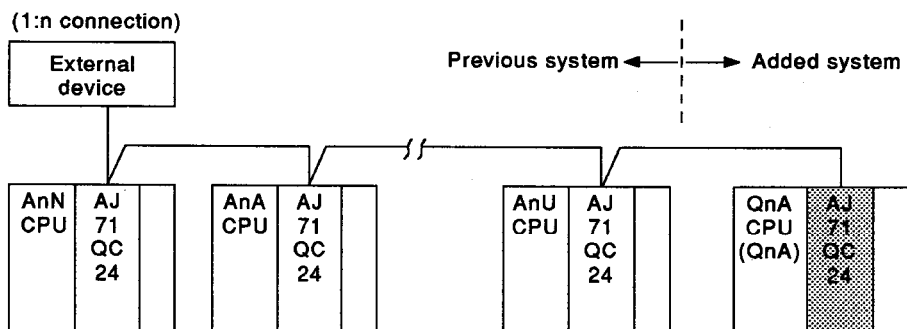
You can reduce sequence programs by registering user-registered frames and system settings in the built-in EEPROM of the AJ71QC24. Up to 200 user-registered frames can be registered.

8.2.6 Incorporation into your existing system

(1) AJ71QC24 can be connected to a multidrop system in which computer link modules (AJ71C24-S[], AJ71UC24, etc.) and external devices are connected in a 1:n or m:n correspondence using a RS-422/485 interface.

(2) AJ71QC24 has an A-series compatible frame, compatible with MELSEC-A series computer link module (AJ71C24-S[], AJ71UC24, etc.) dedicated protocol. (See Section 8.2.1)

External device programs can be utilized by using A-series compatible frames.



IMPORTANT

Design the configuration of a system to provide an external protective or safety inter locking circuit for the PCs.

Under no circumstances will Mitsubishi Electric be liable or responsible for any consequential damage that may arise as a result of the installation or use of this equipment.

All examples and diagrams shown in this manual are intended only as an aid to understanding the text, not to guarantee operation. Mitsubishi Electric will accept no responsibility for actual use of the product based on these illustrative examples.

Owing to the very great variety in possible applications of this equipment, you must satisfy yourself as to its suitability for your specific application.



MODEL	QNA-GAISETSUSYO-E
MODEL CODE	13JF45
IB(NA)66605-A(9607)MEE	

 **MITSUBISHI ELECTRIC CORPORATION**

HEAD OFFICE : MITSUBISHI DENKI BLDG MARUNOUCHI TOKYO 100-0005 TELEX : J24532 CABLE MELCO TOKYO
NAGOYA WORKS : 1-14 , YADA-MINAMI 5 , HIGASHI-KU , NAGOYA , JAPAN

When exported from Japan, this manual does not require application to the Ministry of International Trade and Industry for service transaction permission.

Specifications subject to change without notice.