User's
Manual

Sequence CPU Instruction Manual - Functions (for F3SP22-0S, F3SP28-3N/3S, F3SP38-6N/6S, F3SP53-4H/4S, F3SP58-6H/6S, F3SP59-7S)

IM 34M06P13-01E

vigilantplant.

YOKOGAWA
Yokogawa Electric Corporation

IM 34M06P13-01E
4th Edition

Blank Page

Applicable Product

- **Range-free Multi-controller FA-M3**
  - Model Name: F3SP22, F3SP28, F3SP38, F3SP53, F3SP58, F3SP59
  - Name: Sequence CPU Modules

The document number and document model code for this manual are given below. Refer to the document number in all communications, including when purchasing additional copies of this manual.

Document No.	:	IM 34M06P13-01E
Document Model Code	:	DOCIM

Media No. IM 34M06P13-01E (CD) 4th Edition : Jan. 31, 2012 (YHQ)

Important

■ About This Manual

- This Manual should be passed on to the end user.
- Before using the controller, read this manual thoroughly to have a clear understanding of the controller.
- This manual explains the functions of this product, but there is no guarantee that they will suit the particular purpose of the user.
- Under absolutely no circumstances may the contents of this manual be transcribed or copied, in part or in whole, without permission.
- The contents of this manual are subject to change without prior notice.
- Every effort has been made to ensure accuracy in the preparation of this manual. However, should any errors or omissions come to the attention of the user, please contact the nearest Yokogawa Electric representative or sales office.

■ Safety Precautions when Using/Maintaining the Product

- The following safety symbols are used on the product as well as in this manual.

Danger. This symbol on the product indicates that the operator must follow the instructions laid out in this user's manual to avoid the risk of personnel injuries, fatalities, or damage to the instrument. Where indicated by this symbol, the manual describes what special care the operator must exercise to prevent electrical shock or other dangers that may result in injury or the loss of life.

Protective Ground Terminal. Before using the instrument, be sure to ground this terminal.

Function Ground Terminal. Before using the instrument, be sure to ground this terminal.

Alternating current. Indicates alternating current.

Direct current. Indicates direct current.

The following symbols are used only in the user's manual.

WARNING

Indicates a "Warning".

Draws attention to information essential to prevent hardware damage, software damage or system failure.

CAUTION

Indicates a "Caution"

Draws attention to information essential to the understanding of operation and functions.

TIP

Indicates a "TIP"

Gives information that complements the present topic.

SEE ALSO

Indicates a "SEE ALSO" reference.

Identifies a source to which to refer.

- For the protection and safe use of the product and the system controlled by it, be sure to follow the instructions and precautions on safety stated in this manual whenever handling the product. Take special note that if you handle the product in a manner other than prescribed in these instructions, the protection feature of the product may be damaged or impaired. In such cases, Yokogawa cannot guarantee the quality, performance, function and safety of the product.
- When installing protection and/or safety circuits such as lightning protection devices and equipment for the product and control system as well as designing or installing separate protection and/or safety circuits for fool-proof design and fail-safe design of processes and lines using the product and the system controlled by it, the user should implement it using devices and equipment, additional to this product.
- If component parts or consumable are to be replaced, be sure to use parts specified by the company.
- This product is not designed or manufactured to be used in critical applications which directly affect or threaten human lives and safety — such as nuclear power equipment, devices using radioactivity, railway facilities, aviation equipment, shipboard equipment, aviation facilities or medical equipment. If so used, it is the user's responsibility to include in the system additional equipment and devices that ensure personnel safety.
- Do not attempt to modify the product.

■ Exemption from Responsibility

- Yokogawa Electric Corporation (hereinafter simply referred to as Yokogawa Electric) makes no warranties regarding the product except those stated in the WARRANTY that is provided separately.
- Yokogawa Electric assumes no liability to any party for any loss or damage, direct or indirect, caused by the use or any unpredictable defect of the product.

■ Software Supplied by the Company

- Yokogawa Electric makes no other warranties expressed or implied except as provided in its warranty clause for software supplied by the company.
- Use the software with one computer only. You must purchase another copy of the software for use with each additional computer.
- Copying the software for any purposes other than backup is strictly prohibited.
- Store the original media that contain the software in a safe place.
- Reverse engineering, such as decompiling of the software, is strictly prohibited.
- Under absolutely no circumstances may the software supplied by Yokogawa Electric be transferred, exchanged, or sublet or leased, in part or as a whole, for use by any third party without prior permission by Yokogawa Electric.

■ General Requirements for Using the FA-M3 Controller

● Avoid installing the FA-M3 controller in the following locations:

- Where the instrument will be exposed to direct sunlight, or where the operating temperature exceeds the range 0°C to 55°C (32°F to 131°F).
- Where the relative humidity is outside the range 10 to 90%, or where sudden temperature changes may occur and cause condensation.
- Where corrosive or flammable gases are present.
- Where the instrument will be exposed to direct mechanical vibration or shock.
- Where the instrument may be exposed to extreme levels of radioactivity.

● Use the correct types of wire for external wiring:

- Use copper wire with temperature ratings greater than 75°C.

● Securely tighten screws:

- Securely tighten module mounting screws and terminal screws to avoid problems such as faulty operation.
- Tighten terminal block screws with the correct tightening torque as given in this manual.

● Securely lock connecting cables:

- Securely lock the connectors of cables, and check them thoroughly before turning on the power.

● Interlock with emergency-stop circuitry using external relays:

- Equipment incorporating the FA-M3 controller must be furnished with emergency-stop circuitry that uses external relays. This circuitry should be set up to interlock correctly with controller status (stop/run).

● Ground for low impedance:

- For safety reasons, connect the [FG] grounding terminal to a Japanese Industrial Standards (JIS) Class D Ground*1 (Japanese Industrial Standards (JIS) Class 3 Ground). For compliance to CE Marking, use braided or other wires that can ensure low impedance even at high frequencies for grounding.

*1 Japanese Industrial Standard (JIS) Class D Ground means grounding resistance of 100 Ω max.

● Configure and route cables with noise control considerations:

- Perform installation and wiring that segregates system parts that may likely become noise sources and system parts that are susceptible to noise. Segregation can be achieved by measures such as segregating by distance, installing a filter or segregating the grounding system.

● Configure for CE Marking Conformance:

- For compliance to CE Marking, perform installation and cable routing according to the description on compliance to CE Marking in the “Hardware Manual” (IM 34M06C11-01E).

- **Keep spare parts on hand:**
  - Stock up on maintenance parts including spare modules, in advance.
  - Preventive maintenance (replacement of the module or its battery) is required for using the module beyond 10 years. For enquiries on battery replacement service (for purchase), contact your nearest Yokogawa Electric representative or sales office. (The module has a built-in lithium battery. Lithium batteries may exhibit decreased voltage, and in rare cases, leakage problems after 10 years.)

- **Discharge static electricity before operating the system:**
  - Because static charge can accumulate in dry conditions, first touch grounded metal to discharge any static electricity before touching the system.

- **Never use solvents such as paint thinner for cleaning:**
  - Gently clean the surfaces of the FA-M3 controller with a cloth that has been soaked in water or a neutral detergent and wringed.
  - Do not use volatile solvents such as benzine or paint thinner or chemicals for cleaning, as they may cause deformity, discoloration, or malfunctioning.

- **Avoid storing the FA-M3 controller in places with high temperature or humidity:**
  - Since the CPU module has a built-in battery, avoid storage in places with high temperature or humidity.
  - Since the service life of the battery is drastically reduced by exposure to high temperatures, take special care (storage temperature should be from –20°C to 75°C).
  - There is a built-in lithium battery in a CPU module and temperature control module which serves as backup power supply for programs, device information and configuration information. The service life of this battery is more than 10 years in standby mode at room temperature. Take note that the service life of the battery may be shortened when installed or stored at locations of extreme low or high temperatures. Therefore, we recommend that modules with built-in batteries be stored at room temperature.

- **Always turn off the power before installing or removing modules:**
  - Failing to turn off the power supply when installing or removing modules, may result in damage.

- **Do not touch components in the module:**
  - In some modules you can remove the right-side cover and install ROM packs or change switch settings. While doing this, do not touch any components on the printed-circuit board, otherwise components may be damaged and modules may fail to work.

- **Do not use unused terminals:**
  - Do not connect wires to unused terminals on a terminal block or in a connector. Doing so may adversely affect the functions of the module.

■ Waste Electrical and Electronic Equipment

Waste Electrical and Electronic Equipment (WEEE), Directive 2002/96/EC
(This directive is only valid in the EU.)

This product complies with the WEEE Directive (2002/96/EC) marking requirement. The following marking indicates that you must not discard this electrical/electronic product in domestic household waste.

Product Category
With reference to the equipment types in the WEEE directive Annex 1, this product is classified as a “Monitoring and Control instrumentation” product.
Do not dispose in domestic household waste.
When disposing products in the EU, contact your local Yokogawa Europe B. V. office.

■ How to Discard Batteries

The following description on DIRECTIVE 2006/66/EC (hereinafter referred to as the EU new directive on batteries) is valid only in the European Union.

Some models of this product contain batteries that cannot be removed by the user. Make sure to dispose of the batteries along with the product.

Do not dispose in domestic household waste.
When disposing products in the EU, contact your local Yokogawa Europe B. V. office.

Battery type: Lithium battery

Note: The symbol above means that the battery must be collected separately as specified in Annex II of the EU new directive on batteries.

Introduction

■ Overview of the Manual

This manual describes the sequencing functions of sequence CPU modules (For F3SP22, F3SP28-3N/3S, F3SP38-6N/6S, F3SP53-4H/4S, F3SP58-6H/6S, F3SP59-7S) designed for use with the Range-free Multi-controller FA-M3.

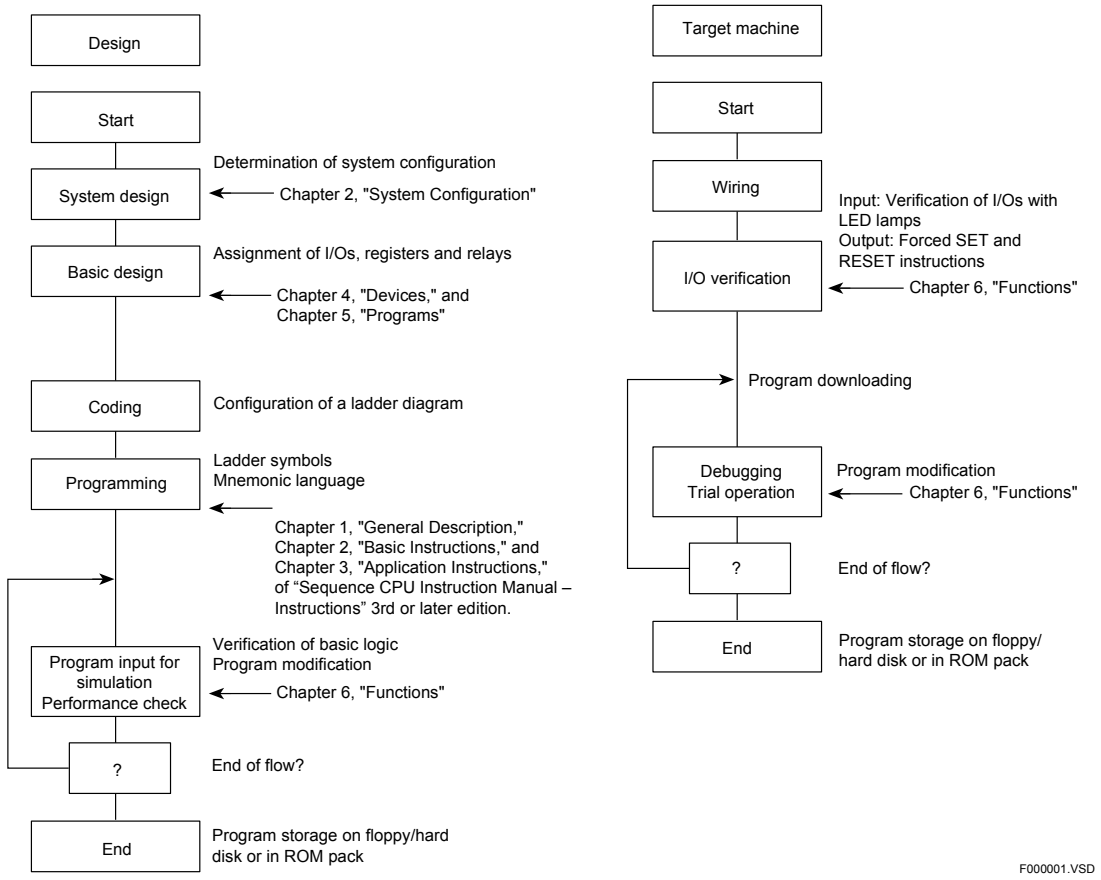
■ How to Read the Manual

If you are a first-time reader, first go through this paragraph, “How to Read the Manual,” and proceed to Chapter 1, then Chapter 3.

For efficiency, read only the relevant remaining chapters according to your flow of work from system design to system operation.

The chart below shows the regular workflow, from system design to system operation, as well as chapters you should refer to in each step.

● Work Flow from System Design to System Operation, and Relevant Chapters



■ Other User's Manuals

Be sure to read each of the following manuals, in addition to this manual.

- **For information on the instructions used with sequence CPUs, refer to:**
  - Sequence CPU Instruction Manual – Instructions (IM 34M06P12-03E)

- **For information on the commands and responses of personal computer link functions**
  - Personal Computer Link Commands (IM 34M06P41-01E).

- **When creating programs using ladder language, refer to:**
  - FA-M3 Programming Tool WideField3 (IM 34M06Q16-☐☐E)

  or
  - FA-M3 Programming Tool WideField2 (IM 34M06Q15-01E)
  - FA-M3 Programming Tool WideField (IM 34M06Q14-01E); and
  - FA-M3 Programming Tool WideField - Application (IM 34M06Q14-02E).

- **For information on the specifications*, configuration*, installation, wiring, trial operation, maintenance and inspection of the FA-M3, as well as information on the system-wide limitation of module installation, refer to:**
  - Hardware Manual (IM 34M06C11-01E).

*: For information on the specifications of products other than the power supply module, base module, I/O module, cable and terminal block unit, refer to their respective user's manuals.

Read the following user's manuals, as required.

- **For information on the functions of F3SP21, F3SP25 and F3SP35 sequence CPU modules, refer to:**
  - Sequence CPU - Functions (for F3SP21, F3SP25 and F3SP35) (IM 34M06P12-02E).

- **For information on the functions of fiber-optic FA-bus modules, refer to:**
  - Fiber-optic FA-bus Module and Fiber-optic FA-bus Type 2 Module, FA-bus Type 2 Module (IM 34M06H45-01E).

- **For information on the functions of FA link H and fiber-optic FA link H modules, refer to:**
  - FA Link H Module, Fiber-optic FA Link H Module (IM 34M06H43-01E).

- **For information on the FL-net functions, refer to:**
  - FL-net (OPCN-2) Interface Module (IM 34M06H32-02E).

- **For information on the functions of BASIC CPU modules, refer to:**
  - BASIC CPU Modules and YM-BASIC/FA Programming Language (IM 34M06Q22-01E).

Copyrights and Trademarks

■ Copyrights

■ Trademarks

FA-M3

Sequence CPU Instruction Manual - Functions

(for F3SP22-0S, F3SP28-3N/3S, F3SP38-6N/6S, F3SP53-4H/4S, F3SP58-6H/6S, F3SP59-7S)

IM 34M06P13-01E 4th Edition

CONTENTS

1. Specifications and Basic Configuration

This chapter explains the CPU module specifications and the basic configuration of the Range-free Multi-controller FA-M3.

1.1 Overview

This section describes the overview, features and main functions of the sequence CPU module.

■ Overview

Models F3SP28-3N, F3SP38-6N, F3SP53-4H, F3SP58-6H, F3SP22-0S, F3SP28-3S, F3SP38-6S, F3SP53-4S, F3SP58-6S and F3SP59-7S are CPU modules with built-in memory for use with the Range-free Multi-controller FA-M3.

In addition to high-speed operation and large memory capacity, these modules have many more features that help increase development and maintenance efficiency.

■ Features

● High-speed Operation

- 20K steps/1 ms, with shortest scan interval of 200 μs
- High-speed I-P-R-S, which means:
  - High-speed **I**nstruction
  - High-speed **P**erformance
  - High-speed **R**esponse
  - High-speed **S**can

● Sensor Control Functions

In addition to normal scanning, each CPU module has an independent, multiple constant scan function, permitting fast scanning. Fast response is also achievable with a single CPU.

You can execute a block of your program at a high-speed constant scan (200 μs-minimum), separately from normal scanning. This feature enables you to eliminate the effects of a fault diagnosis program or MMI program, as well as ensure stable control program operation.

● Object Ladder

The FA-M3 Programming Tool WideField3, an object-oriented ladder language development tool, is available with the CPU module. This tool not only increases software development productivity over and above structured programming, but also simplifies program maintenance.

● Function for Storing Comments [F3SP□□-□S]

Circuit comments, subcomments, and tag name definitions (including I/O comments) can be stored in the sequence CPU or the ROM pack. This function allows you to debug a program using tag names, even during unscheduled maintenance.

● Other Features

- A compact body allows for reduced panel enclosure size.
- Large-capacity programs and large device sizes are supported to cope with advanced, complex control applications.
- Index modification and structured ladder language simplifies program design and maintenance.
- The device size and operating method can be flexibly configured to suit your application needs.
- A rich set of functions are provided to facilitate program debugging and maintenance. For example, a forced SET/RESET function independent of program operation results.
- A carefully designed self-diagnosis function supplements a highly reliable design.
- Macro instruction functions allow you to create and register new instructions.
- The sampling trace functions acquire and displays the states of multiple devices for a maximum of 1024 scans.
- The programming tool connection port supports the personal computer link functions and thus enables connection to a higher-level computer or a monitor without the need for a personal computer link module.
- The log function records errors encountered in a program, as well as messages created and registered in advance.
- F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 or F3SP59 modules can be mounted in slots 2 to 4 of the main unit, for use as add-on CPU modules for sequence processes added to the main CPU module (F3SP21, F3SP22, F3SP25, F3SP28, F3SP35, F3SP38, F3SP53, F3SP58 F3SP59, F3SP66, F3SP67, F3SP71, or F3SP76).
- A ROM pack can be attached so that you can perform ROM-based operation and store programs.
- Program protection functions ensure security.
- The partial download functions allow downloading of specified blocks only, which increases debugging efficiency especially in collaborative program development. [F3SP□□-□S]
- Indirect specification via devices enables large volume data handling and creation of efficient programs. [F3SP□□-□S]
- Structure macros simplify passing of data to macros and updating of these data structures. [F3SP□□-□S]

■ Functions

- Sensor control
- Configuration (setup of parameters, including device size, range of devices to be latched in case of power failure, and external output to be retained in case of sequence stop)
- Constant scan (at an interval of 1 to 190 ms, in 0.1 ms increments)
- Sampling trace
- Debugging (forced SET/RESET instructions, online editing, etc.)
- Error logging, user logging
- Clock (year, month, day, hour, minute, second, and day of week)
- Support for programming tool connection port with the personal computer link functions
- Program protection
- Program/data storage in ROM pack
- Circuit/sub-comment, tag name definitions storage in ROM pack [F3SP□□-□S]
- Circuit/sub-comment storage function and tag name definition storage function [F3SP□□-□S]
- Partial download functions [F3SP□□-□S]

See Section 1.2, “Specifications,” for more information.

1.2 Specifications

This section describes the basic specifications of the FA-M3 sequence CPU module for each CPU type. For functional specifications, see Section 1.2.1, "List of Specifications." For the types and number of devices, see Section 1.2.2, "Device List." For configuration setup ranges, see Section 1.2.3, "Configuration."

For the names and functions of the components of the sequence CPU module, see Section 1.2.4, "Components and Their Functions." For the external dimensions of the sequence CPU module, see Section 1.2.5, "External Dimensions."

1.2.1 List of Specifications

Table 1.1 Functional Specification (F3SP□□-□S) (1/2)

Item		Specifications					
		F3SP22-0S	**F3SP28-3S**	**F3SP53-4S**	**F3SP38-6S**	**F3SP58-6S**	**F3SP59-7S**
Control method		Repetitive operation based on stored programs					
I/O method		Refresh method/Direct I/O instruction					
Programming language		Structured ladder language and mnemonic language					
Number of I/O points		4096 max.			8192 max. (including remote I/O points)		
Number of internal relays (I)		16384	16384	16384	32768	32768	65535
Number of shared relays (E)		2048	2048	2048	2048	2048	2048
Number of extended shared relays (E)		2048	2048	2048	2048	2048	2048
Number of link relays (L)		8192	8192	8192	16384	16384	16384
Number of special relays (M)		9984	9984	9984	9984	9984	9984
Number of timers (T)		2048 in total	2048 in total	2048 in total	3072 in total	3072 in total	3072 in total
Number of counters (C)							
Number of data registers (D)		16384	16384	16384	32768	32768	65535
Number of shared registers (R)		1024	1024	1024	1024	1024	1024
Number of extended shared registers (R)		3072	3072	3072	3072	3072	3072
Number of file registers (B)		32768	32768	32768	262144	262144	262144
Number of link registers (W)		8192	8192	8192	16384	16384	16384
Number of special registers (Z)		1024	1024	1024	1024	1024	1024
Number of labels		1024	1024	1024	1024	1024	1024
Number of input interrupt processing routines		4	4	4	4	4	4
Constants	Decimal constant	16-bit instruction: -32768 to 32767 32-bit instruction: -2147483648 to 2147483647					
	Hexadecimal constant	16-bit instruction: $0 to $FFFF (hexadecimal number) 32-bit instruction: $0 to $FFFFFFFF (hexadecimal number)					
	Character-string constant	16-bit instruction: e.g. “AB” 32-bit instruction: e.g. “ABCD”					
	IEEE single-precision floating-point constant	32-bit instruction: e.g. 1.23, -3.21 Approximately -3.4×10^{38} to 3.4×10^{38}					
	Index constant	0 to 2047					
Program size (ROM resident)		10K steps max.	30K steps max.	56K steps max.	120K steps max.		254K steps max.
(Program + Tag Name Definition) size							360K steps max.
ROM-resident size (Program + Tag Name Definition)		120K steps max.			360K steps max.		
Number of program blocks		1024 max.					
Number of instructions	Basic instructions	37					
	Application instructions	329					
Number of macro instructions		256 max.					
Instruction execution time	Basic instruction	0.045 to 0.18μs per instruction	0.045 to 0.18μs per instruction	0.0175 to 0.07μs per instruction	0.045 to 0.18μs per instruction	0.0175 to 0.07μs per instruction	
	Application instruction	0.18 μs per instruction	0.18 μs per instruction	0.07 μs per instruction	0.18 μs per instruction	0.07μs per instruction	

Table 1.2 Functional Specification (F3SP□□-□S) (2/2) (Continued)

Item	Specifications					
	F3SP22-0S	F3SP28-3S	F3SP53-4S	F3SP38-6S	F3SP58-6S	F3SP59-7S
Special module High-speed Read (HRD) Instruction /special module High-speed Write (HWR) Instruction	64 instructions each					
Sampling trace functions	Available. These functions collect and display the states of multiple devices for a maximum of 1024 scans.					
Support for personal computer link functions by programming tool connection port	Available. These functions allow a personal computer or a monitor to be connected to the programming tool connection port to perform communications equivalent to the personal computer link module.					
User log functions	Available. These functions allow the user to execute a user log command to log (record the history of) errors in the user system, including information on the state of occurrence and system operation, etc.					
Number of personal computer link modules	6 max.					
Macro instruction functions	Available. These functions allow the user to create and register new user-defined instructions.					
Scan monitoring time	Variable from 10 to 200 ms.					
Startup at power-on or recovery from power failure	Automatic (Auto-logging of power-on time, power-off time and momentary power failure time)					
Sensor control functions	Available. In addition to normal scanning, these functions allow one specified block to be scanned at high-speed fixed intervals.					
Constant scan time	1 to 190 ms, configurable in 0.1 ms increments.					
Self-diagnosis	Detection of memory failure, CPU failure and I/O module failure, syntax checking, etc.					
Link functions	FA link, FL-net, personal computer link, and remote I/O link (fiber-optic FA-bus, μ-bus)					
Comment storage functions	Available. Circuit comment, sub-comment, tag name definition (including I/O comment).					
Other functions	- Online editing - Forced SET/RESET instructions - Clock (year, month, day, hour, minute, second, and day of the week) - Configuration (setup of parameters, including device capacities, range of devices to be latched at power failure, and external outputs to be latched at sequence stop) - Protection - Stop refreshing function					

Table 1.3 Functional Specification (F3SP□□-□N, F3SP□□-□H)

Item		Specifications			
		F3SP28-3N	F3SP53-4H	F3SP38-6N	F3SP58-6H
Control method		Repeated operation based on stored programs			
I/O method		Refresh method/Direct I/O instruction			
Programming language		Structured ladder language and mnemonic language			
Number of I/O points		4096 max.		8192 max., including remote I/O points	
Number of internal relays (I)		16384	16384	32768	32768
Number of shared relays (E)		2048	2048	2048	2048
Number of extended shared relays (E)		2048	2048	2048	2048
Number of link relays (L)		8192	8192	16384	16384
Number of special relays (M)		9984	9984	9984	9984
Number of timers (T) Number of counters (C)		2048 in total	2048 in total	3072 in total	3072 in total
Number of data registers (D)		16384	16384	32768	32768
Number of shared registers (R)		1024	1024	1024	1024
Number of extended shared registers (R)		3072	3072	3072	3072
Number of file registers (B)		32768	32768	262144	262144
Number of link registers (W)		8192	8192	16384	16384
Number of special registers (Z)		1024	1024	1024	1024
Number of labels		1024	1024	1024	1024
Number of input interrupt processing routines		4	4	4	4
Constants	Decimal constant	16-bit instruction: -32768 to 32767 32-bit instruction: -2147483648 to 2147483647			
	Hexadecimal constant	16-bit instruction: $0 to $FFFF (hexadecimal number) 32-bit instruction: $0 to $FFFFFFFF (hexadecimal number)			
	Character-string constant	16-bit instruction: e.g. “AB”, etc 32-bit instruction: e.g. “ABCD”, etc.			
	Floating-point constant	32-bit instruction: e.g. 1.23, -3.21 approximately -3.4×10^{38} to $+3.4\times10^{38}$			
Program size (that can be ROM resident)		30K steps max.	56K steps max.	120K steps max.	
Number of program blocks		1024 max.			
Number of instructions	Basic instructions	33			
	Application instructions	312			
Number of macro instructions		64 max.			
Instruction execution time	Basic instruction	0.045 to 0.18 µs per instruction	0.0175 to 0.07 µs per instruction	0.045 to 0.18 µs per instruction	0.0175 to 0.07 µs per instruction
	Application instruction	0.18 µs min. per instruction	0.07 µs min. per instruction	0.18 µs min. per instruction	0.07 µs min. per instruction
Special module High-speed Read instruction (HRD)/special module High-speed Write instruction (HWR)		64 instructions each			
Sampling trace functions		Available. This function collects and displays the states of multiple devices for a maximum of 1024 scans.			
Support for personal computer link functions by programming tool connection port		Available. The function allows a personal computer or a monitor to be connected to the programming tool connection port to perform communications equivalent to the personal computer link module.			
User logging functions		Available. These functions allow the user to execute a user log instruction to log (record the history of) errors in the user system, including information on the state of occurrence and system operation, etc.			
Number of personal computer link modules		6 max.			
Macro instruction functions		Available. These functions allow a user to create and register new user-defined instructions.			
Scan monitoring time		Variable from 10 to 200 ms			
Startup at power-on or recovery from power failure		Automatic (Auto-logging of power-on time, power-off time and momentary power failure time)			
Sensor control functions		Available. In addition to normal scanning, these functions allow one specific block to be scanned at high-speed fixed intervals.			
Constant scan time		1 to 190 ms, configurable in 0.1 ms increments			
Self-diagnosis		Detection of memory failure, CPU failure and I/O module failure, syntax checking, etc.			
Link function		FA link, FL-net, personal computer link, and remote I/O link (fiber-optic FA-bus, µ-bus)			
Other functions		- Online editing - Forced SET/RESET instructions - Clock (year, month, day, hour, minute, second, and day of the week) - Configuration (setup of parameters, including device capacities, range of devices to be latched at power failure, and external outputs to be latched when sequence stops) - Protection			

1.2.2 Device List

Table 1.4 Device List

Device		Code	F3SP22-0S F3SP28-3N/3S F3SP53-4H/4S		F3SP38-6N/6S F3SP58-6H/6S		F3SP59-7S		Remarks
			Range	Quantity	Range	Quantity	Range	Quantity	
Input relay		X	X00201 to X71664 (discontinuous)	4096	X00201 to X71664 (discontinuous)	8192	X00201 to X71664 (discontinuous)	8192	The range used depends on the module type
Output relay		Y	Y00201 to Y71664 (discontinuous)		Y00201 to Y71664 (discontinuous)		Y00201 to Y71664 (discontinuous)		
Internal relay		I	I00001 to I16384	16384	I00001 to I32768	32768	I00001 to I65535	65535	
Shared relay		E	E0001 to E2048	2048	E0001 to E2048	2048	E0001 to E2048	2048	These devices default to zero in quantity. Be sure to configure the devices when using the CPU module in a multi-CPU configuration.
Extended shared relay	Non-latched type		E2049 to E4096	2048	E2049 to E4096	2048	E2049 to E4096	2048	
Link relay	Non-latched type	L	L0001 to L72048 (discontinuous)	8192	L0001 to L72048 (discontinuous)	16384	L0001 to L72048 (discontinuous)	16384	Used in FA link and FL-net communications.
Special relay		M	M0001 to M9984	9984	M0001 to M9984	9984	M0001 to M9984	9984	
Timer	100 µs Timer	T	T0001 to T0016	2048 in total	T0001 to T0016	3072 in total	T0001 to T0016	3072 in total	Configurable for up to 16 timers
	1 ms Timer		T0001 to T2048		T0001 to T3072		T0001 to T3072		Configuration limit correlated to counters (C) (*1).
	10 ms Timer								
	100 ms Timer								
Continuous timer	100 ms Timer								
Counter	Latched type	C	C0001 to C2048		C0001 to C3072		C0001 to C3072		Configuration limit correlated to Timers (T) (*1)
Data register	Latched type	D	D00001 to D16384	16384	D00001 to D32768	32768	D00001 to D65535	65535	
File register	Latched type	B	B000001 to B32768	32768	B000001 to B262144	262144	B000001 to B262144	262144	
Link register	Non-latched type	W	W00001 to W72048 (discontinuous)	8192	W00001 to W72048 (discontinuous)	16384	W00001 to W72048 (discontinuous)	16384	Used in FA link and FL-net communications.
Special register		Z	Z0001 to Z1024	1024	Z0001 to Z1024	1024	Z0001 to Z1024	1024	
Index register		V	V001 to V256	256	V001 to V256	256	V001 to V256	256	
Shared register	Non-latched type	R	R0001 to R1024	1024	R0001 to R1024	1024	R0001 to R1024	1024	These devices default to zero in quantity. Be sure to configure the devices when using the CPU module in a multi-CPU configuration.
Extended shared register			R1025 to R4096	3072	R1025 to R4096	3072	R1025 to R4096	3072	

*1: See Table 1.5.

Table 1.5 Device Capacities and Configuration Restrictions

Device	Code	F3SP22-0S F3SP28-3N/3S F3SP53-4H/4S		F3SP38-6N/6S F3SP58-6H/6S		F3SP59-7S	
		Default value	Setup Restrictions	Default value	Setup Restrictions	Default value	Setup Restrictions
Timer	T	1024	Total for timers and counters: 2048 max. Default value for 100-µs and 1-ms timers: 0	2048	Total for timers and counters: 3072 max. Default value for 100-µs and 1-ms timers: 0	2048	Total for timers and counters: 3072 max. Default value for 100-µs and 1-ms timers: 0
Counter	C	1024		1024		1024	
Shared relay	E	0	2048 max.	0	2048 max.	0	2048 max.
Extended Shared relay	E	0	2048 max.	0	2048 max.	0	2048 max.
Shared register	R	0	1024 max.	0	1024 max.	0	1024 max.
Extended shared register	R	0	3072 max.	0	3072 max.	0	3072 max.

1.2.3 Configuration

This section describes the configuration function. The configuration setup ranges are summarized in the table below.

■ Configuration Function

The sequence CPU contains the predefined defaults of device sizes and operation methods.

You can use these defaults to run programs. In some applications, however, they may not suit your specific purpose of use. In such a case flexibility allows for defaults to be changed to meet your needs. Changing the defaults is called “configuration” and can be performed through the FA-M3 programming tool WideField3 (hereinafter simply referred to as WideField3) and the FA-M3 programming tool WideField2.

■ Tables of Configuration Ranges

Table 1.6 Configuration Range (1/5)

F3SP22-0S, F3SP28-3N, F3SP28-3S, F3SP53-4H, F3SP53-4S				
Item			**Default**	**Configuration Range**
Device capacities	Shared Device (E, R)	Shared relay (E)	0	2048 points max. on 32-point basis for all CPUs combined
		Extended shared relay (E)	0	2048 points max. on 32-point basis for all CPUs combined
		Shared register (R)	0	1024 points max. on 2-point basis for all CPUs combined
		Extended shared register (R)	0	3072 points max. on 2-point basis for all CPUs combined
	Link Device (L, W)	Link relay (L)	System 1 to 4: 2048	8192 points max. on 16-point basis (Note) for all links combined.
			System 5 to 8: 0	
		Link register (W)	System 1 to 4: 2048	8192 points max. on 16-point basis (Note) for all links combined.
			System 5 to 8: 0	
	Configuration of Timer (T) /Counter (C)	100 μs timer	0	2048 points on 1-point basis for timers and counters combined; 16 max. for 100 μs timers; Timer numbers are continuous.
		1 ms timer	0	
		10 ms timer	512	
		100 ms timer	448	
		100 ms continuous timer	128	
		Counter	1024	2048 points on 1-point basis for timers and counters combined.

Table 1.7 Configuration Range (2/5)

F3SP38-6N, F3SP38-6S, F3SP58-6H, F3SP58-6S, F3SP59-7S				
Item			Default	Configuration Range
Device capacities	Shared Device (E, R)	Shared relay (E)	0	2048 points max. on 32-point basis for all CPUs combined
		Extended shared relay (E)	0	2048 points max. on 32-point basis for all CPUs combined
		Shared register (R)	0	1024 points max. on 2-point basis for all CPUs combined
		Extended shared register (R)	0	3072 points max. on 2-point basis for all CPUs combined
	Link Device (L, W)	Link relay (L)	2048 for each system	16384 points max. on 16-point basis(Note) for all links combined
		Link register (W)	2048 for each system	16384 points max. on 16-point basis(Note) for all links combined
	Configuration of Timers (T)/counters (C)	100 µs timer	0	3072 points on 1-point basis for timers and counters combined; 16 points max. for 100 µs timers; timer numbers are continuous.
		1 ms timer	0	
		10 ms timer	1024	
		100 ms timer	896	
		100 ms continuous timer	128	
		Counter	1024	3072 points on 1 point basis for timers and counters combined.

Table 1.8 Configuration Range (3/5)

F3SP22-0S, F3SP28-3N/3S, F3SP53-4H/4S, F3SP38-6N/6S, F3SP58-6H/6S, F3SP59-7S				
Item			Default	Configuration Range
Extended device configuration	Configuration of the range of devices to be latched in case of power failure	Internal relay (I)	I0001 to I1024	Configurable on 32-point basis; continuous from the starting number
		shared relay (E) Extended shared relay (E)	Non-latched type	
		Link relay (L)	Non-latched type	Configurable on 16-point basis
		Timer (T)	Non-latched type (except for continuous timers)	Configurable on 1-point basis; continuous from the starting number
		Counter (C)	All latched	
		Data register (D)	All latched	Configurable on 2-point basis; continuous from the starting number
		Shared registers (R) Extended shared registers (R)	Non-latched type	
		Link register (W)	Non-latched type	Configurable on 16-point basis

Note: The configuration range of each of shared relays (E) and extended shared relays (E) and shared registers (R) and extended shared registers (R) to be latched in case of power failure is assigned numbers continuous from the starting number. However, if the number of shared relays (E) is smaller than 2048, the last of them is followed by the first extended shared relay (E) numbered E2049. Likewise, if the number of shared registers (R) is smaller than 1024, the last of them is followed by the first extended shared register (R) numbered R1025.

Example)

In a case where there are 1024 shared relays (E) and 2048 extended shared relays (E):
If you define the starting number as 513 and the number of units as 1024 for the range of devices to be latched in case of power failure, then the devices that are latched include:
E513 to E1024 shared relays (E); and
E2049 to E2560 extended shared relays (E).

Note: The configuration range of each of link relays (L) and registers (W) to be latched in case of power failure is assigned numbers continuous from the starting number.
However, the following exceptions apply.
The number following L/W01024 is L/W11024.
The number following L/W11024 is L/W21024.
The number following L/W21024 is L/W31024.
The number following L/W31024 is L/W41024.
The number following L/W41024 is L/W51024.
The number following L/W51024 is L/W61024.
The number following L/W61024 is L/W71024.
(The rules noted above are true when the number of link relays (L) or registers (W) to be used is defined as 1024. If the number is 2048, the number following L/W02048 is L/W10001.)

Example:

When there are 1024 link relays (L) each for link 1, link 2 and link 3:
If you define the starting number as 10513 and the number of units as 1024 for the range of devices to be latched in case of power failure, then the devices included in the latching are:
L10513 to L11024 link relays (L) for link 1; and
L20001 to L20512 link relays (L) for link 2.

Table 1.9 Configuration Range (4/5)

F3SP22-0S, F3SP28-3N/3S, F3SP53-4H/4S, F3SP38-6N/6S, F3SP58-6H/6S, F3SP59-7S				
Item			**Default Value**	**Value Range**
Initial data of data register		Data register (D)	None	Configurable for up to 1024 contiguous points from a starting number
Operation Control	Scan monitoring time		200 ms	Configurable from 10 to 200 ms in increments of 10 ms
	Constant scan		Do not Use	Configurable from 1.0 to 190.0 ms in increments of 0.1 ms
	Error-time action (operating mode in case of error)	I/O module error	Stop	Run/Stop (configurable)
		I/O comparison error	Stop	
		Instruction parameter error	Stop	
		Scan timeout	Stop	
		Subroutine error	Stop	
		Interrupt error	Stop	
		Subunit communication error	Run	
		Sensor CB scan timeout	Stop	
	Program execution mode		All Blocks	All Blocks/Specified Blocks
	Momentary power failure detection mode	Valid for all power supply modules except F3PU01-0N	Standard mode	Standard/Immediate
	Peripheral processing time		Not set up	100 µs to 190 ms in increments of 100 µs
Interrupt Setup	Sensor CB	Execution interval	200 µs	200 µs to 25.0 ms in increments of 100 µs
		Timing of interrupt	Immediate (during instruction execution)	After Instruction/Immediate (during instruction execution)
	Input interrupt	Timing of interrupt	After instruction	After Instruction/Immediate (during instruction execution)
	Priority of interrupts		Sensor CB interrupt has priority	Sensor CB interrupt has priority/Input module has priority
Input/Output Setup	Terminal usage (Module used/not used)		Used	Used/Not used/Use with SCB; Configurable on 16-terminal basis
	Data code		BIN	BIN/BCD; configurable on 16-terminal basis
	Input sampling interval		16 ms	16 ms/1 ms/250 µs/62.5 µs/Always; configurable on 16-terminal basis
	Output when stopped (Reset/hold external outputs when sequence stops)		Reset	Reset/hold; configurable on 16-terminal basis
ROM Setup	Device current values to be resident in ROM	Data registers (D) File registers (B)	None	Up to 32768 contiguous points from a starting number

Note: Configure on 32-terminal basis when using the same input module for both sensor control block and regular blocks.

SEE ALSO

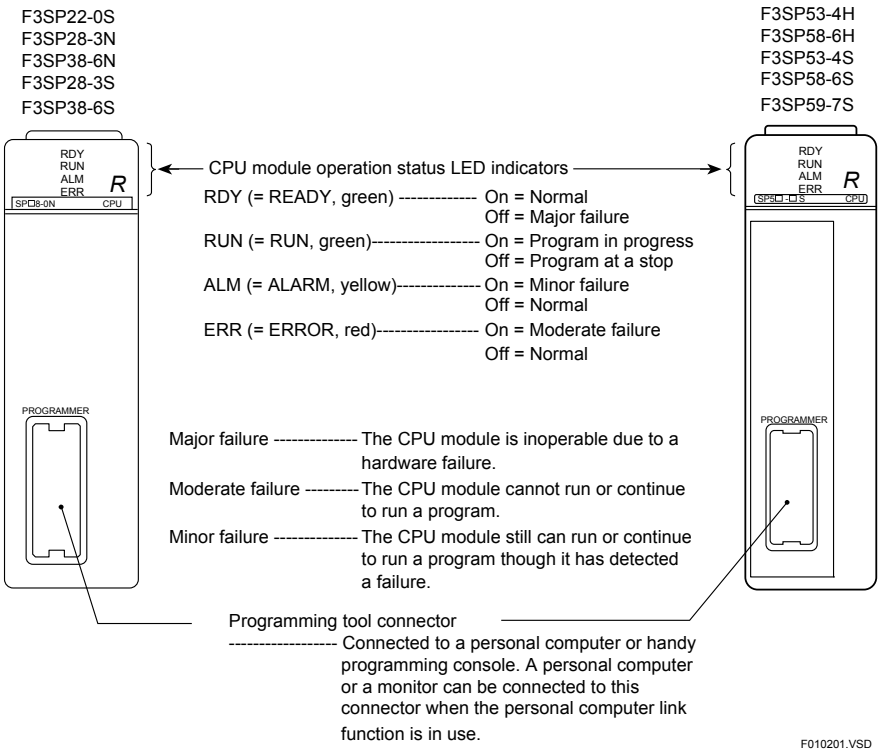
For more information on the subunit communication error, see “Fiber-optic FA-bus Module and Fiber-optic FA-bus Type 2 Module, FA-bus Type 2 Module” (IM 34M06H45-01E).

Table 1.10 Configuration Range (5/5)

F3SP22-0S, F3SP28-3N/3S, F3SP53-4H/4S, F3SP38-6N/6S, F3SP58-6H/6S, F3SP59-7S				
Item			**Default Value**	**Value Range**
Communications Setup	Connection port for programming tool	Communication mode	Mode 0: 9600bps, even parity	Mode 0: 9600bps, Even Parity Mode 1: 9600bps, No Parity Mode 2: 19200bps, Even Parity Mode 3: 19200bps, No Parity Mode 4: 38400bps, Even Parity Mode 5: 38400bps, No Parity Mode 6: 57600bps, Even Parity Mode 7: 57600bps, No Parity Mode 8: 115200bps, Even Parity Mode 9: 115200bps, No Parity
	CPU personal computer link	Used/Unused	Unused	Unused/Used
		Checksum	No	Yes/No
		End character	No	Yes/No
		(Program) protection	No	Yes/No
FA link setup (Mapping between FA link and FL-net numbers and slot numbers)			None	Yes/No Link number from 1 to 8 Slot numbers from 1 to 16
FL-net Refreshing	Common data refreshing mode		Peripheral process	Peripheral process or Control process
	Common data refreshing range		All nodes	All nodes or Some nodes; Node numbers 1 to 254
Shared Refreshing	Shared refreshing range (partial disabling of refreshing)		All refreshed	Enable/Disable refreshing, configurable separately for shared relays(E) , shared registers(R) , extended shared relays (E) and extended shared registers (R) of each CPU module
	Shared refreshed data		Simultaneous	Simultaneous/Non-simultaneous
	Shared refreshing mode		Peripheral process	Peripheral process/ Control process

1.2.4 Components and Their Functions

This section describes the LED indicators, their states, and the programming tool connector on the front side of the sequence CPU module. These features are common to the F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 and F3SP59 CPU modules.



The table below summarizes combinations of the LED indicators as classified by the severity of failure.

Table 1.11 LED Indicator Combinations Based on the Severity of Failure

Status / LED Indicator	Normal	Major Failure	Moderate Failure	Minor Failure
RDY	○	●	○	○
RUN	○	●	●	○
ALM	●	△	△	○
ERR	●	○	○	●

○: ON, ●: OFF, △: ON or OFF

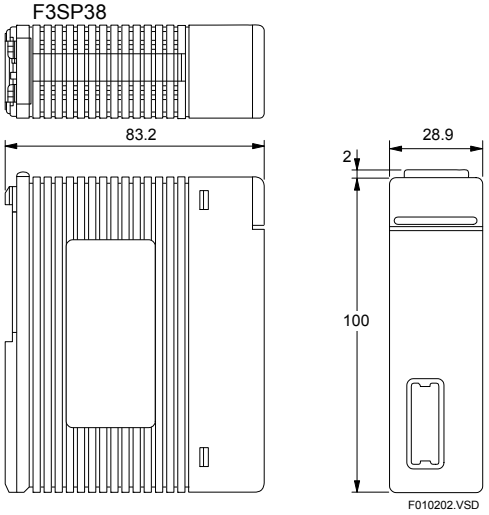
Table 1.12 Weight

Model	Weight
F3SP22, F3SP28, F3SP38	130g
F3SP53, F3SP58, F3SP59	210g

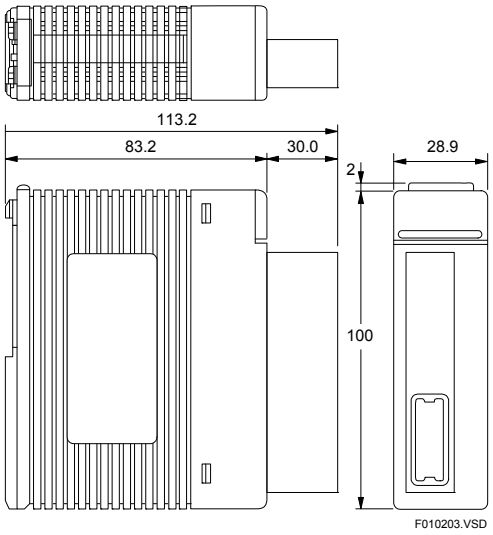
1.2.5 External Dimensions

F3SP22
F3SP28
F3SP38

Unit: mm



F3SP53
F3SP58
F3SP59



1.3 Basic Configuration

This section describes units, slot numbers and I/O relay numbers which form the basic configuration of an FA-M3.

Units, slots, and input/output relays are identified with unique numbers. These numbers are used in parameters of ladder instructions and configuration setup.

1.3.1 Unit

A unit is a system with the minimum configuration consisting of the following modules. Install these modules on the base module to compose the unit.

Table 1.13 Unit Components (Modules)

Name	Description
Base module	Five types are available allowing different number of modules to be mounted.
Power supply module	One power supply module must always be mounted on the base module.
CPU module	At least one CPU module is required. Several types are available with different functionalities.
I/O module	Various types are available with different types of I/O and number of I/O points.
Special module	Various types are available, including analog I/O and communication modules.

The location where you install a module is called a slot.

■ Main Unit

Install the power supply module in the leftmost slot of the base module and the CPU module in the slot on the immediate right of the power supply module. Then, install required I/O and special modules in the remaining slots. A system with this configuration is called a main unit.

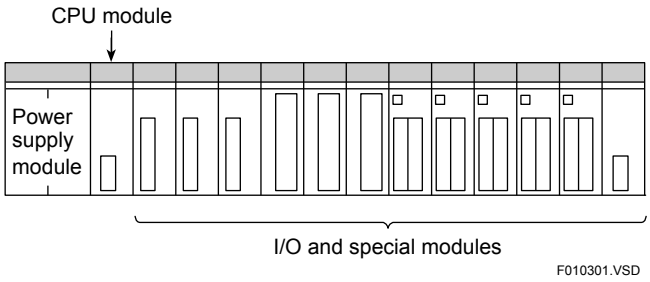


Figure 1.1 Main Unit

■ Subunit

A subunit is an I/O expansion unit. It is connected to the main unit through a fiber-optic FA-bus, fiber-optic FA-bus type 2 or FA-bus type 2 module.

A maximum of seven subunits can be connected to the main unit and are identified by their unit numbers. With fiber-optic FA-bus type 2, you can separate any single subunit into a maximum of eight stations. For more information on the method of separation, see “Fiber-optic FA-bus Module and Fiber-optic FA-bus Type 2 Module, FA-bus Type 2 Module” (IM 34M06H45-01E).

SEE ALSO

For details on unit number, see Section 1.3.2.

1.3.2 Slot Number

A slot number indicates the position of a slot where a module is installed.

A slot number is defined as a three-digit integer, as shown below.

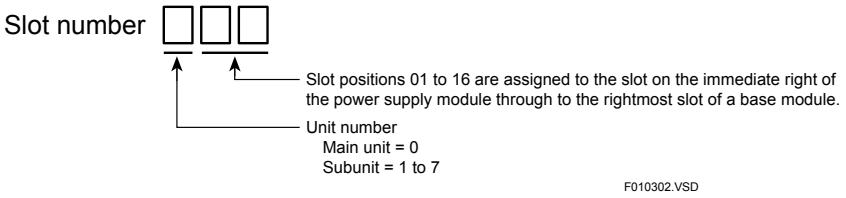


Figure 1.2 Slot Numbers (1 of 2)

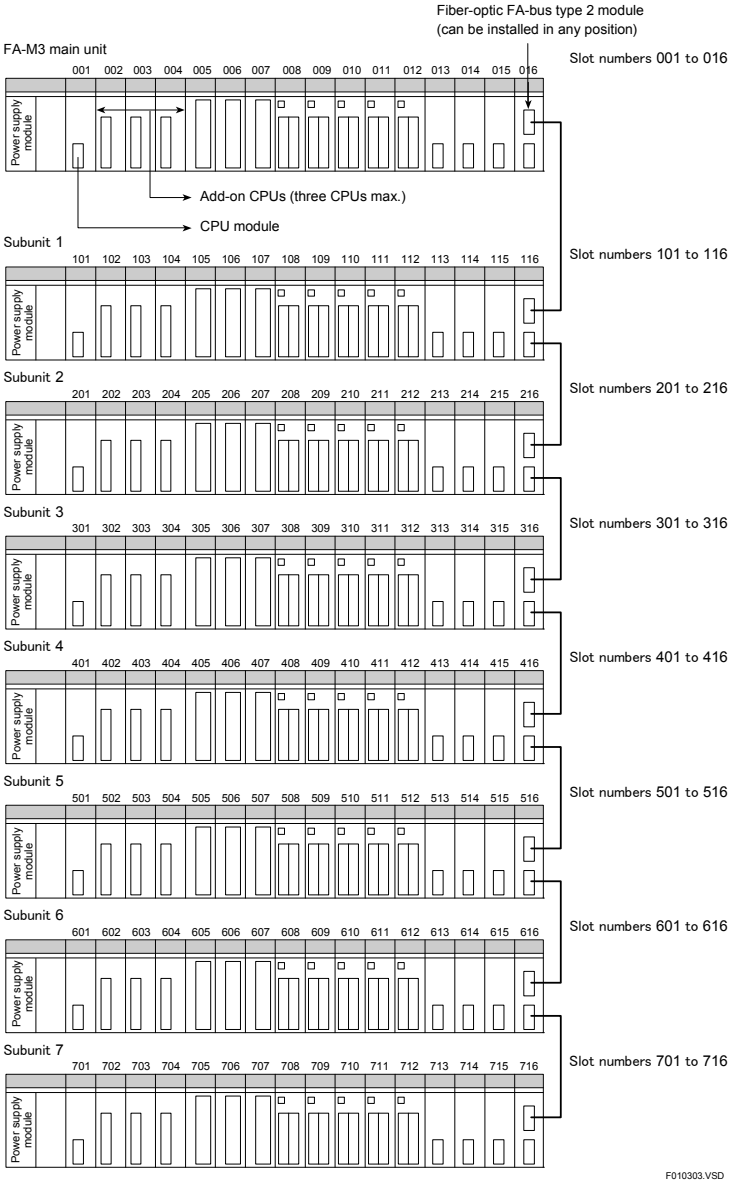


Figure 1.3 Slot Numbers (2 of 2)

Install fiber-optic FA-bus type 2 modules in both the main unit and a subunit and connect these modules with a fiber-optic cable.

You can attach up to seven subunits to the main unit. Subunit numbers are assigned by setting the rotary switch on the front panel of each fiber-optic FA-bus type 2 module.

1.3.3 I/O Relay Number

Each input relay (X) or output relay (Y) number is defined as a slot number followed by an I/O relay number. The I/O relay number is a number corresponding to each terminal of an I/O module.

Example:

The output relay number for terminal 6 of an F3YC08-0N module installed in slot 005 is defined as follows.

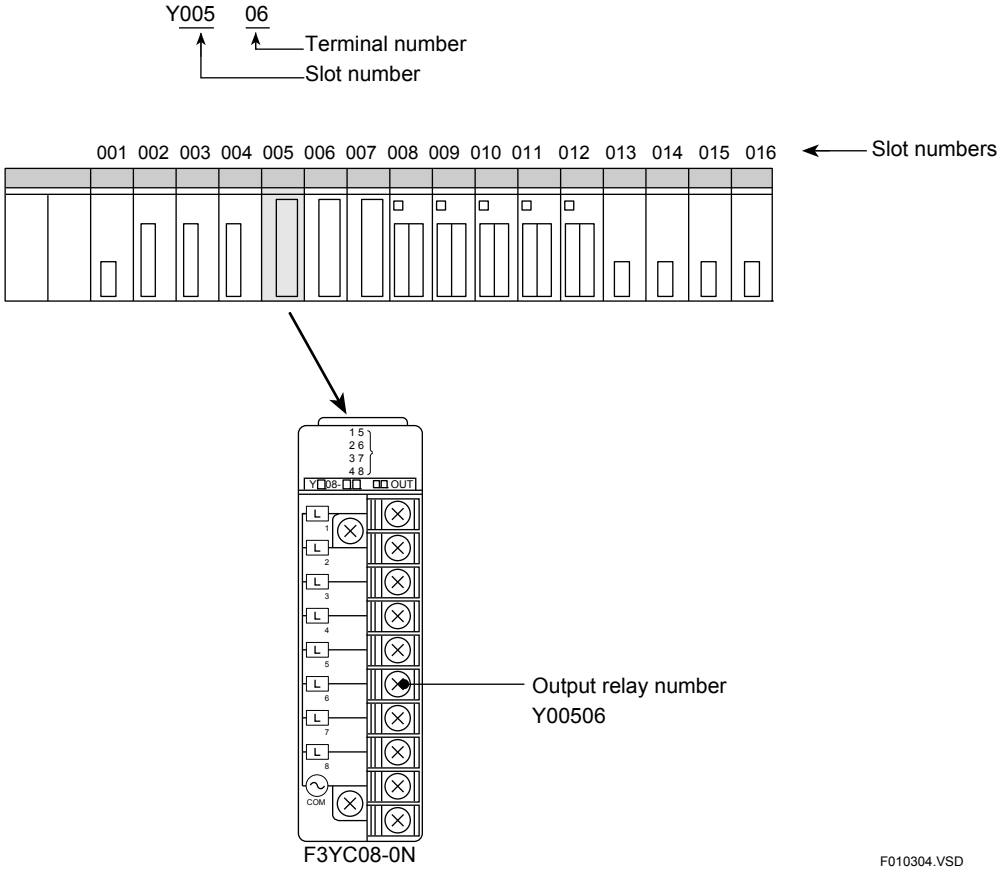


Figure 1.4 Output Relay Number

The input and output terminal numbers of a mixed-I/O module or special module with 32 input and output points each are assigned as 1 to 32 and as 33 to 64, respectively.

Blank Page

2. System Configuration

This chapter describes the FA-M3 system configuration and programming tools.

2.1 Basic System Configuration

The basic system configuration refers to a system consisting of a main unit only.

For more information on the main unit, see subsection 1.3.1, “Unit.”

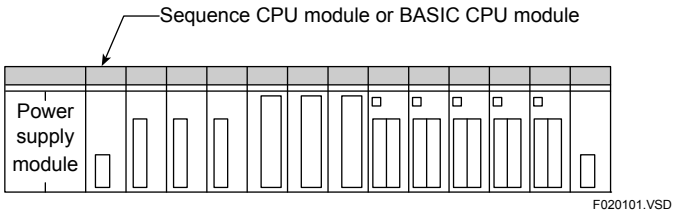


Figure 2.1 Example of Basic System Configuration (when a 13-slot base module is used)

2.2 Multi-CPU System Configuration

This section describes a multi-CPU system configuration and the handling of I/O modules in a multi-CPU system.

2.2.1 Multi-CPU System Configuration

A multi-CPU system configuration refers to a system comprising multiple CPU modules. A maximum of four CPU modules can be installed in the slots (slots 001 to 004) on the main unit.

A CPU module installed in slot 001 serves as the main CPU module and CPU modules installed in slots 002 to 004 serve as add-on CPU modules.

A maximum of four sequence CPU modules can be installed at the same time, while only one F3BP□□ BASIC CPU module is allowed in this system configuration.

A CPU module installed in the Nth (N = 1 to 4) slot is called the Nth CPU (module) or CPU N.

TIP

A BASIC CPU module refers to a CPU module which is controlled by BASIC programs.

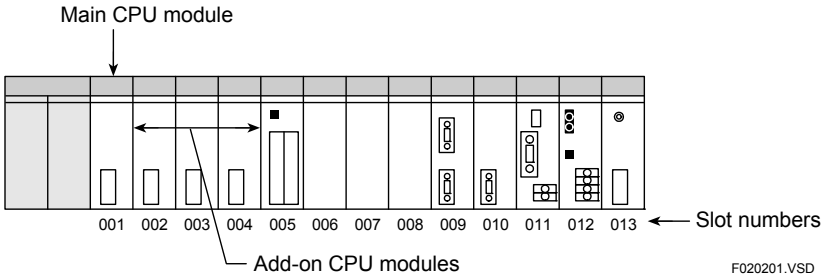


Figure 2.2 Example of Multi-CPU System Configuration

CAUTION

Be careful not to install any CPU module in the 5th or later slot and turn on the power. Otherwise, the memory is cleared and factory settings are restored.

2.2.2 Handling I/O Modules in Multi-CPU System

■ Input Modules

With input modules, you can read input data through multiple CPUs. To do this, configure the CPUs so that they share the same input sampling interval for the input module in question. Be careful, as the sampling interval that can be set varies, depending on the CPU type.

■ Output Modules and Special Modules with Y□□□□□ Output Relays (Y)

● Combination of F3SP22, F3SP28, F3SP38, F3SP53, F3SP58, F3SP59, F3SP66, F3SP67, F3SP71 and F3SP76 CPU Modules

You can output data from multiple sequence CPU modules separately to the output relays of the same output module on 16-point basis. To do this, configure the unused output relays (Y) of the output module as “Not used” on 16-point basis. In addition, all output relays within the same module must be configured with the same output mode (hold or reset) in the event that sequence processing stops.

● Other Combinations of CPU Modules

You may not use the same output module with multiple CPUs. Configure all CPUs that do not use the output module so that the output module is set to “Not Used.”

SEE ALSO

For details on modules that are not used, see Section 4.1.4.

2.3 Extended System Configuration

An extended system configuration refers to a system configured by adding remote I/O modules, a personal computer link module, an FA link module, and an FL-net module to the basic system.

2.3.1 Remote I/O System

The remote I/O system refers to a system configured using fiber-optic FA-bus, fiber-optic FA-bus type 2 and FA-bus type 2 modules.

The number of remote I/O points is included in the count of all I/O points.

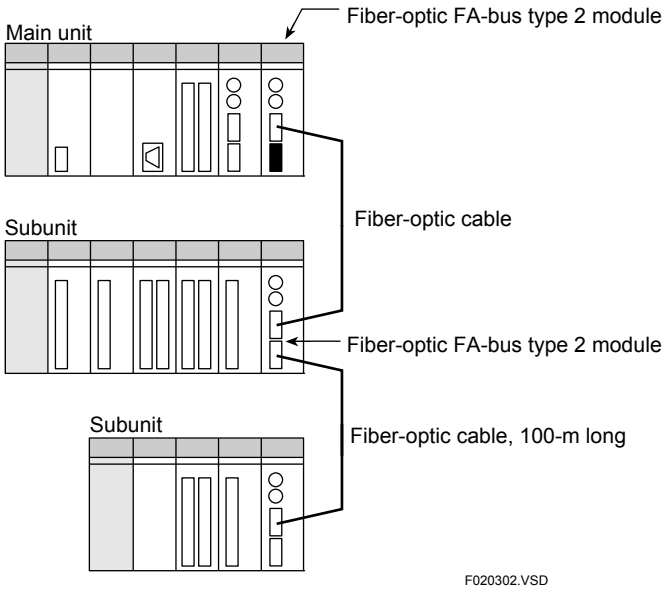


Figure 2.3 Example of System Using Fiber-optic FA-bus Type 2 Modules

2.3.2 Personal Computer Link System

The personal computer link system refers to a system configured by connecting a personal computer or a monitor to the main unit through a personal computer link module. The sequence CPU module can be connected directly to a personal computer or a monitor.

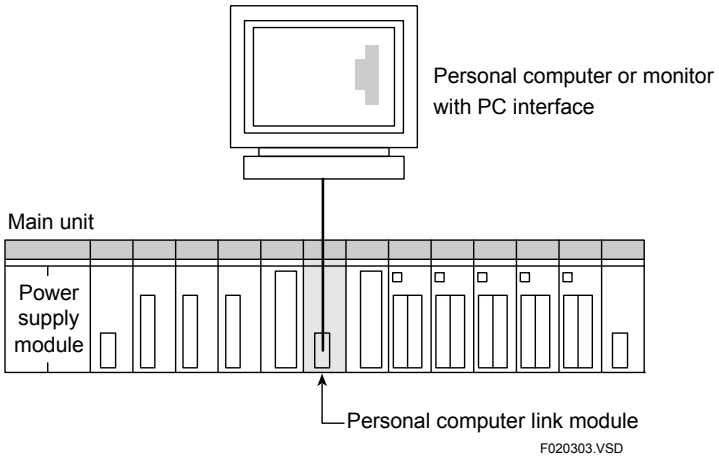


Figure 2.4 Example of Personal Computer Link System

2.3.3 FA Link System

The FA link system refers to a system that employs FA link communication to build a network system with programmable controllers.

The types of communication covered by an FA link system are:

- FA link H communication (FA link H module), and
- Fiber-optic FA link H communication (fiber-optic FA link H module).

Unless otherwise specified, the term “FA link” in this manual comprehensively refers to these two types of communication. For more information on the FA link, see “FA Link H Module, Fiber-optic FA Link H Module” (IM 34M06H43-01E).

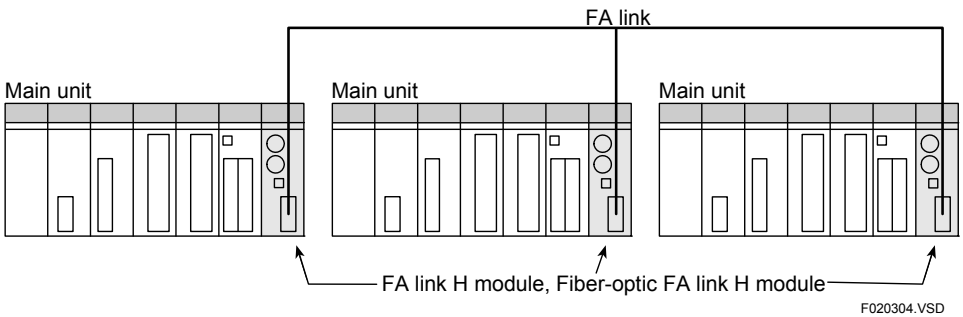
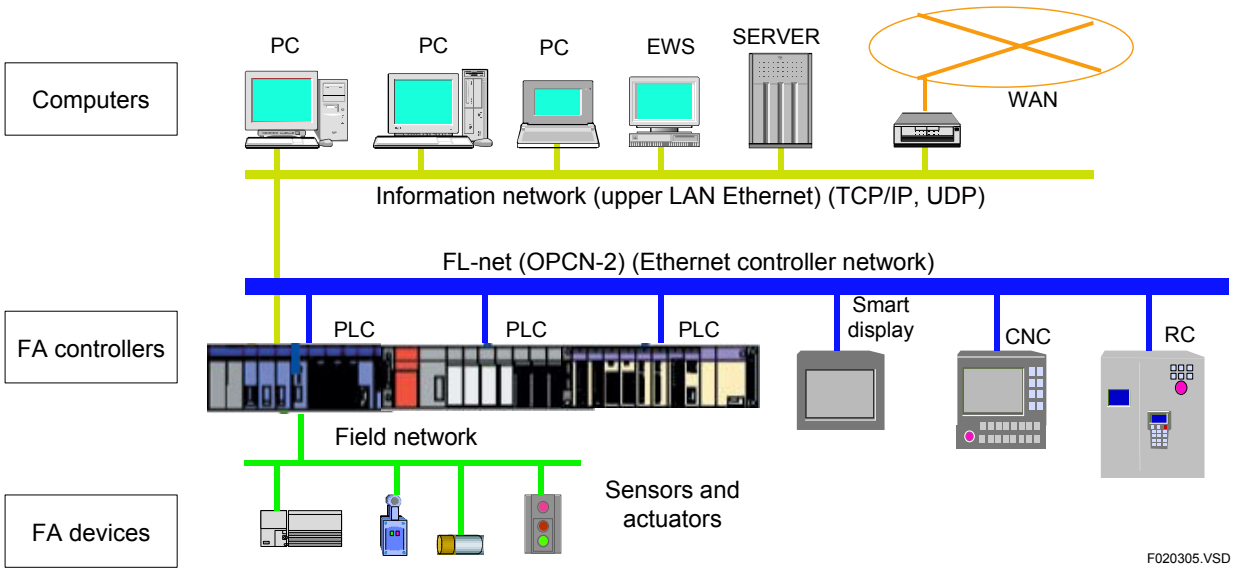


Figure 2.5 Example of FA Link System

2.3.4 FL-net System

An FL-net system is based on FL-net, which is an open network for connecting various programmable controllers (PLC), computerized numerical controllers (CNC) and other factory automation (FA) controllers (including personal computers (PC)) from multiple vendors.

For details on FL-net, see "FL-net (OPCN-2) Interface Module" (IM 34M06H32-02E).



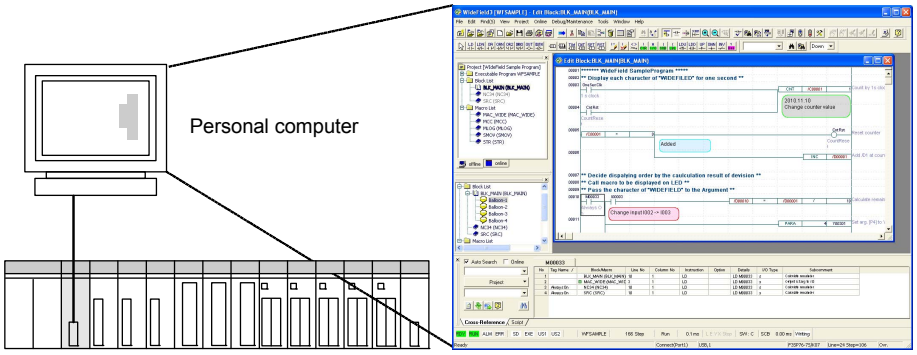
2.4 Programming Tool

The FA-M3 programming tool WideField3 and WideField2, or simply WideField3 and WideField2, is available as a programming tool for the F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 and F3SP59 sequence CPU modules.

2.4.1 WideField3/WideField2

Description	Software Model	Compatible Sequence CPU Modules
FA- M3 Programming Tool WideField3	SF630-☐CW	F3SP05 F3SP08 F3SP21 F3SP22 F3SP25 F3SP35 F3SP28 F3SP38 F3SP53 F3SP58 F3SP59 F3FP36 F3SP66 F3SP67 F3SP71 F3SP76

Description	Software Model	Compatible Sequence CPU Modules
FA- M3 Programming Tool WideField2	SF620-MCW	F3SP05 F3SP08 F3SP21 F3SP25 F3SP35 F3SP28 F3SP38 F3SP53 F3SP58 F3SP59 F3FP36 F3SP66 F3SP67



F020401.VSD

Figure 2.6 WideField3

■ Object Ladder

WideField3 (or WideField2) defines “blocks” and “instruction macros” that compose a ladder program as “objects,” a term commonly used in the computing world. All objects are responsible for their provided functions and have a high degree of independence. Consequently, the language offers higher productivity, better maintainability and more effective program reuse, as compared to a structured programming language.

■ Features

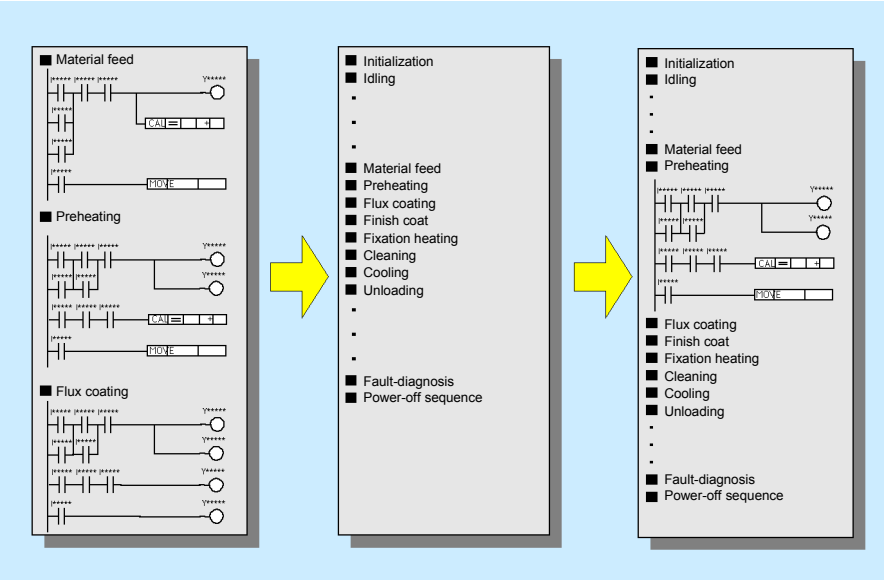
● Componentization

With componentization, blocks can be reused as complete components. Devices that are used within a block are defined separately from the main program. Thus, blocks can be easily recombined without undesirable duplicate use of the same device. Macro functions can also be turned into components.

.

Index View

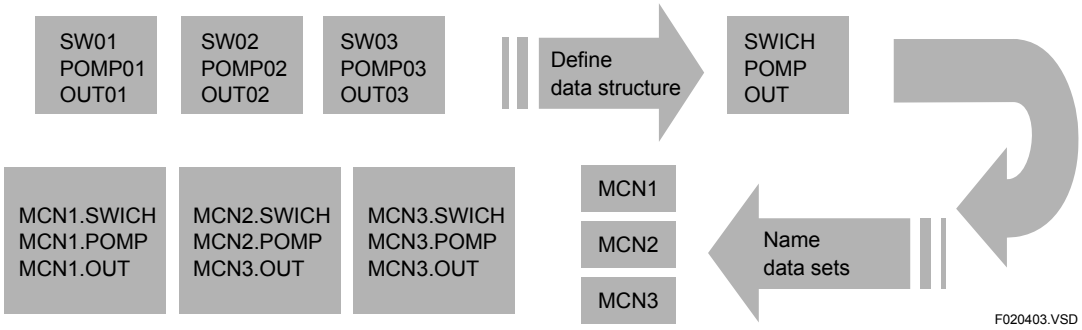
You can display an outline view of a large program by “hiding” non-required details. This enables more efficient debugging.



F020402.VSD

Group Tag Names

You can group individual tag names into a group tag name to enable definition of data sets.



F020403.VSD

Easy Data Exchange with Windows-based Applications

You can select data items such as device names and comments on a Microsoft Excel screen and copy them to WideField3 (or WideField2) (drag-and-drop function). You can also copy ladder circuits created using WideField3 (or WideField2) to Windows applications such as Microsoft Word.

3. Basic Operations of Sequence CPU Module

This chapter describes the basic operating modes of the sequence CPU module and add-on CPU modules, as well as their methods of program execution.

3.1 Operating Modes of Sequence CPU Module

The sequence CPU module has three operating modes: Run mode, Debug mode and Stop mode.

■ Run Mode

The Run mode is a state in which the sequence CPU module is running a program, and is used for actual system operation. You can monitor the operating status of a sequence CPU module or devices. However, you can use none of the debug functions available from the WideField3 (or WideField2) programming tool. In this mode, the **RDY** and **RUN** LED indicators turn on.

■ Debug Mode

The Debug mode is used when debugging and tuning programs.

You can execute programs in the same way as with the Run mode. In Debug mode, you can use debugging functions, such as forced SET/RESET instructions and online edit, through WideField3 (or WideField2). These functions, however, affect the scan time. Disable the functions at the end of debugging and tuning, and set the CPU to Run mode. In this mode, the **RDY** and **RUN** LED indicators turn on.

The Debug mode includes a pause state in which the sequence CPU module suspends program execution during such debugging operation as scan operation. In this state, the **RUN** LED indicator turns off and all external outputs being generated by the program are latched.

■ Stop Mode

The Stop mode is a state in which the sequence CPU module stops program execution.

In this mode, you can remove programs and clear devices, in addition to using forced SET/RESET instructions, online editing and debug operation. In this mode, the **RUN** LED indicator turns off.

The external outputs being generated by the program are set to ON (hold) or OFF (reset), according to the setting of the configuration item "Output When Stopped" of "Input/Output Setup." By default, all external outputs are set to OFF.

Table 3.1 LED Indicator Combinations Based on the Operating Mode

Operating Mode / LED Indicator	Run	Debug	Stop
RDY	○	○	○
RUN	○	△	●
ALM	●	●	●
ERR	●	●	●

○: ON, ●: OFF, △: ON or OFF

3.2 Operation at Power-on/off

This section describes the operations when power is turned off or turned on.

3.2.1 Operation at Power-on

When the power is turned on, the CPU performs initialization to get ready for program execution.
During initialization, the CPU performs I/O collation and instruction interpretation to check whether its hardware and programs are normal.
If no error is detected, the CPU begins executes a program from its beginning.
If equipped with a ROM pack, the CPU reads programs from the pack and begins system operation.
If in the **ROM Writer** mode, however, the CPU does not read programs from the ROM pack. Alternatively, it enters a command-wait state (e.g., waits for a ROM transfer command from the WideField3 (or WideField2)) without executing a program.

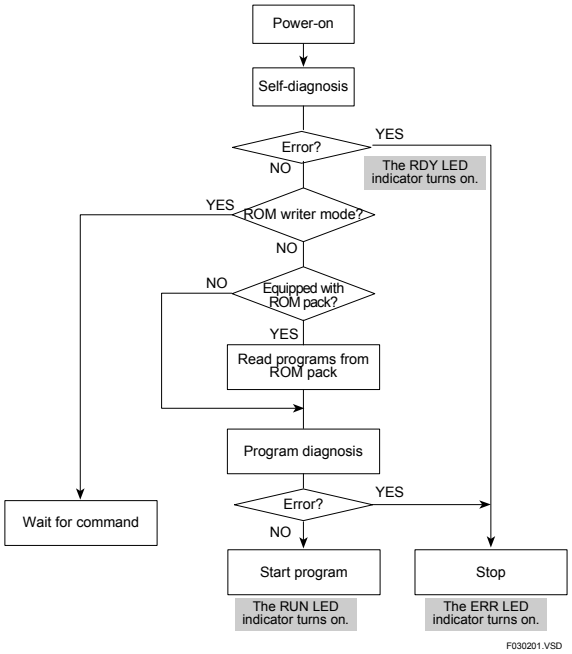


Figure 3.1 Operation at Power-on

3.2.2 Operation at Power-off

When the power is turned off, the sequence CPU module records the date and time in its error log file and stops system operation.

TIP

The error log function saves to an error log file information such as time of occurrence and type of error when a system error occurs or when the power is turned on or turned off.

SEE ALSO

For details on the error log, see Chapter C3 of "FA-M3 Programming Tool WideField3" (IM 34M06Q16-□□E) or Chapter B23 of "FA-M3 Programming Tool WideField2" (IM 34M06Q15-01E).

3.3 Operation in Case of Momentary or Complete Power Failure

This section describes settings for operation in case of momentary power failure, specifying the momentary power failure detection mode, as well as operation in case of complete power failure, specifying the data latch range in case of complete power failure.

3.3.1 Operation in Case of Momentary Power Failure

Two types of power failure detection mode are available for detecting a momentary power failure: the standard mode and the immediate detection mode.

The CPU operates differently in case of a momentary power failure, depending on the type of power failure detection mode selected.

The immediate detection mode can be selected by configuration only when the F3PU10-0☐, F3PU16-0N, F3PU20-0☐, F3PU26-0N, F3PU30-0☐ or F3PU36-0☐ power supply module is used.

■ Standard Mode

If a momentary power failure occurs, the sequence CPU module records the date and time in its error log file. The sequence CPU module suspends processing until the power is restored. This causes a delay in the scan time and timer update process.

When the power is restored, the sequence CPU module resumes execution at the point where processing was suspended.

A program can detect a momentary power failure by monitoring a special relay (M195).

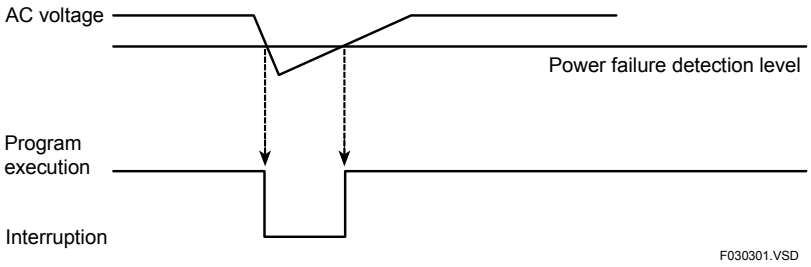


Figure 3.2 Operation in Case of Momentary Power Failure

CAUTION

While the sensor control block is active, a momentary power failure may result in a sensor control block scan timeout. In this case, the sensor control block stops and must be restarted after the power is restored.

■ Immediate Detection Mode

If a momentary power failure occurs, the sequence CPU module records the date and time in its error log file. The sequence CPU module suspends processing until power is restored. At this point the CPU sets the external outputs generated by a program to OFF, and actuates the FAIL contact.

When power is restored, the sequence CPU module performs a reset-and-start sequence and executes the program from its beginning.

3.3.2 Momentary Power Failure Detection Mode Setup

This configuration item defines the momentary power failure detection mode.
You can select either the standard mode or the immediate detection mode. The default is the standard mode. For details on each of these modes, see "Hardware Manual" (IM 34M06C11-01E).

CAUTION

In a multi-CPU configuration, all CPU modules must be configured with the same momentary power failure detection mode.

3.3.3 Operation in Case of Complete Power Failure

If a complete power failure occurs, the CPU operates as it does at power off.
You can configure the types and ranges of devices to be latched in case of a complete power failure. This strategy allows the CPU to resume operation from its previous state after power is restored.
When power is restored, the CPU executes the program from its beginning.

TIP

Latching devices at power failure stores device states immediately before a power failure so that a program can continue execution in the same state after power is restored.

3.3.4 Data Latch Range at Power Failure

This configuration item sets the range of devices to be latched in case of a complete power failure.
Specify the starting number and the number of devices for each device type.
The following table shows the default setting and the configurable range of each device type.

Table 3.2 Data Latch Range at Power Failure of Configuration

Item			Default			Configuration Range
			F3SP22 F3SP28 F3SP53	F3SP38 F3SP58	F3SP59	
Extended device configuration	Data Latch Range at Power Failure	Internal relay (I)	I0001 to I1024	I0001 to I1024	I0001 to I1024	Configurable on 32-relay basis; contiguous from a starting number*1
		Shared relay (E) Extended shared relay (E)	Non-latching type	Non-latching type	Non-latching type	
		Link relay (L)	Non-latching type	Non-latching type	Non-latching type	Configurable on 16-relay basis*2
		Timer (T)	Non-latching type (except for continuous timers)	Non-latching type (except for continuous timers)	Non-latching type (except for continuous timers)	Configurable for timers/counters, contiguous from a starting number
		Counter (C)	All latched (C0001 to C1024)	All latched (C0001 to C1024)	All latched (C0001 to C1024)	
		Data register (D)	All latched (D00001 to D16384)	All latched (D00001 to D32768)	All latched (D00001 to D65535)	Configurable on 2-register basis; contiguous from a starting number*1, *2
		Shared register (R) Extended shared register (R)	Non-latching type	Non-latching type	Non-latching type	
		Link register (W)	Non-latching type	Non-latching type	Non-latching type	Configurable on 16-register basis*2

*1: If the upper limit of the range of shared relays (E) used is smaller than E2049, the last device number for shared relays (E) is followed by the first device number for extended shared relays (E). Likewise, if the upper limit of shared registers (R) used is smaller than R1025, the last device number for shared registers (R) is followed by the first device number for extended shared registers (R).

*2: The data latch range setup for link relay and link register are mapped to contiguous devices starting from their respective starting numbers with the following exceptions:

L/W01024 is followed by L/W10001;
L/W11024 is followed by L/W20001;
L/W21024 is followed by L/W30001;
L/W31024 is followed by L/W40001;
L/W41024 is followed by L/W50001;
L/W51024 is followed by L/W60001;
L/W61024 is followed by L/W70001.

The above rule applies when the number of link relays or registers to be used is defined as 1024. If the number used is 2048, L/Wn2048 is followed by L/Wn0001. If the number used is 8192, L/W08192 is followed by L/W10001.

3.4 Operation Processing Method

This section outlines data operation processing (scan processing) in the sequence CPU module. Details are explained in subsequent sections.

The CPU employs a stored-program iterative operation method.

In this method, a created program is pre-stored in the memory of the sequence CPU module. The sequence CPU executes instructions, one at a time, starting from the first step of the program. After executing the last step in the program, the CPU performs required processing, such as self-diagnosis. It then repeats the instructions from the first step.

Each of these iterative cycles is called “one scan” and the time required for one scan is called a “scan time.”

In the case of F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 and F3SP59 CPU modules, the CPU executes instructions and peripheral processes concurrently to perform each scan in a shorter time.

Common processing, instruction execution, input refreshing, output refreshing, and synchronization processing are classified as a system of control-related processes, while tool service, personal computer link service, CPU service, link refreshing, and shared refreshing are classified as a system of peripheral processes. The CPU performs these two kinds of processes concurrently to speed up the control-related processes.

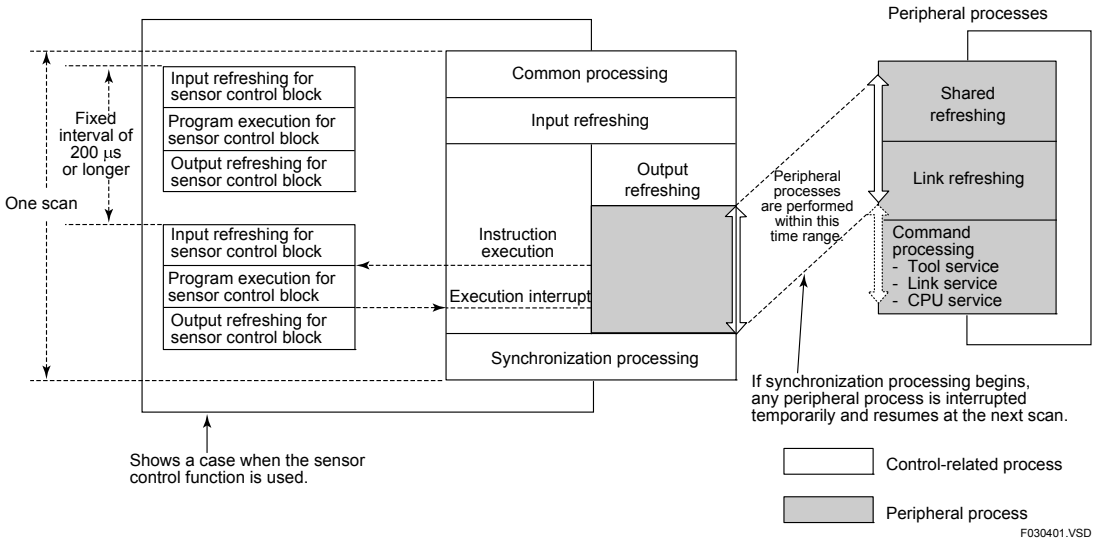


Figure 3.3 Processing Method

SEE ALSO

For details on the sensor control function, see Section 6.15, “Sensor Control Function.”

TIP

Common processing includes self-diagnosis, updating of special relays (M) and special registers (Z), as well as updating of timers. The END processing is sometimes known as an END scan.

TIP

The input refreshing process reflects data from input contacts of, say a DI module, to input relays (X).

The output refreshing process reflects data from output relays (Y) to output contacts of, say a DO module.

TIP

The synchronization process synchronizes the system of control-related processes with the system of peripheral processes. In particular, it reflects data to the link refresh and shared refresh (inter-PLC communication) devices.

● System of Control-related Processes

This system performs basic operations of the sequence CPU module, such as instruction execution and I/O refreshing.

Execution of the system of control-related processes is called one scan, and the execution time required by the system is usually called the scan time.

● System of Peripheral Processes

This system supports the programming tool WideField3 and performs communication between the CPU module and a personal computer or an FA link module.

The system of peripheral processes is concurrent with and independent of the system of control-related processes. Therefore, neither the number of modules connected nor the content of each peripheral process affects the operation of the system of control-related processes.

● Synchronization between Systems of Control-related Processes and Peripheral Processes

The system of peripheral processes executes concurrent with and independent of the system of control-related processes. For processes related to operation control (e.g., run or stop) or processes requiring simultaneity of data, however, the CPU synchronizes these two systems using a synchronization process included in the system of control-related processes.

The time required for the synchronization process varies depending on its content. Using debugging functions, such as online editing, affects the scan time.

CAUTION

If the ratio of the instruction execution time to the scan time is too small, insufficient time may be allocated for executing the system of peripheral processes. Consequently, the responses of link refreshing, shared refreshing, tool service, personal computer link service and CPU service will become extremely slow. If this happens, use a constant scan with an interval somewhat longer than the normal scan time, or set up the peripheral processing time to secure sufficient time for executing the system of peripheral processes.

3.5 Method of Executing Peripheral Processes

Peripheral processes are executed concurrently with instructions in a program.

When the execution of program instructions is completed, any peripheral process is interrupted until the next scan, in order to prevent the process from affecting the scan time. This means the peripheral processing time is affected by the program execution time.

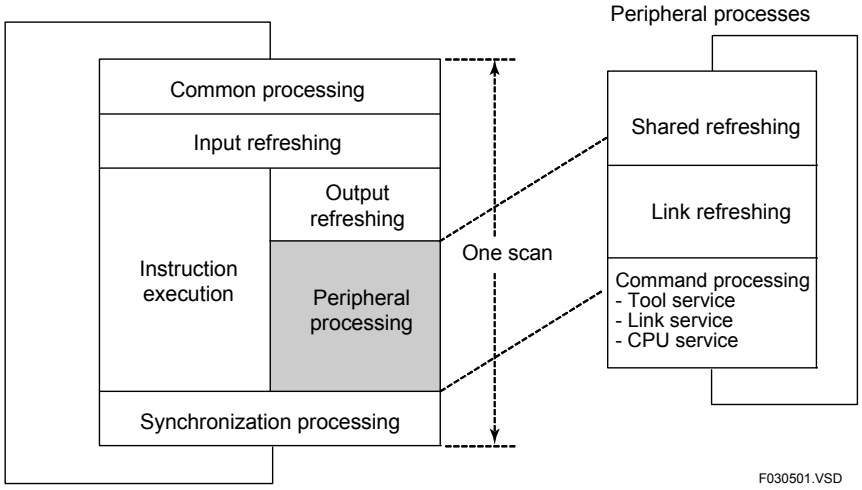


Figure 3.4 Peripheral Processing

■ Specifying Peripheral Processing Time

You can define the peripheral processing time. Use this configuration item when the instruction execution time is so short that insufficient time is allocated to peripheral processes. To secure enough time, the scan time is lengthened, or the delays of shared refreshing, link refreshing and command processing included in the peripheral processes are shortened. If at the end of instruction execution, any peripheral process is found to have not run as long as the defined time, the process is prolonged until the expiry of the defined time. In that case, the scan is also prolonged by as much as the extended portion of the defined time. Beware that the CPU ignores the defined peripheral processing time if a constant scan time is defined.

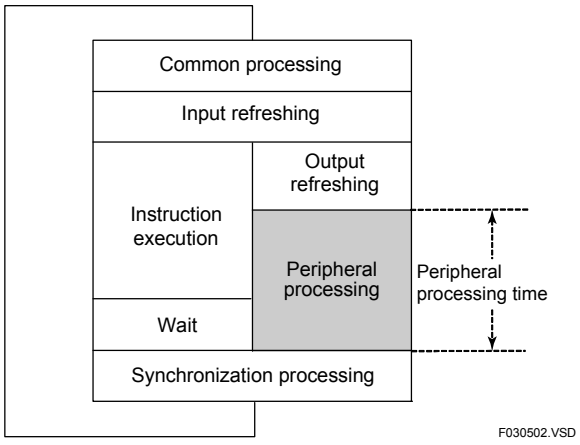


Figure 3.5 Peripheral Processing Time

The configurable range is from 0.1 ms to 190 ms, on 0.1 ms basis. If no peripheral processing time is defined, the CPU operates with a peripheral processing time of 0.2 ms by default.

3.6 Method of I/O Processing

This section describes how I/O processing is performed, I/O response delay, as well as I/O processing in a multi-CPU system.

3.6.1 Method of I/O Processing

As the method of I/O processing, the CPU uses batch refreshing.

In this method, the sequence CPU module acquires all data changes in the input module into the input relay (X) area of the CPU's data memory before executing each scan.

The sequence CPU module uses data contained in this area when performing operations.

Operation results are output to the output relay (Y) area of the CPU's data memory each time an operation is performed. The results are sent to the output module, collectively and concurrently with the execution of instructions in the next scan.

Only modules that are installed in the system and configured in Input/Output Setup as "Used" will have their input and output refreshed. No error occurs even if a program tries to access input or output relays of a module which is not installed or a module which is configured in Input/Output Setup as "Not used".

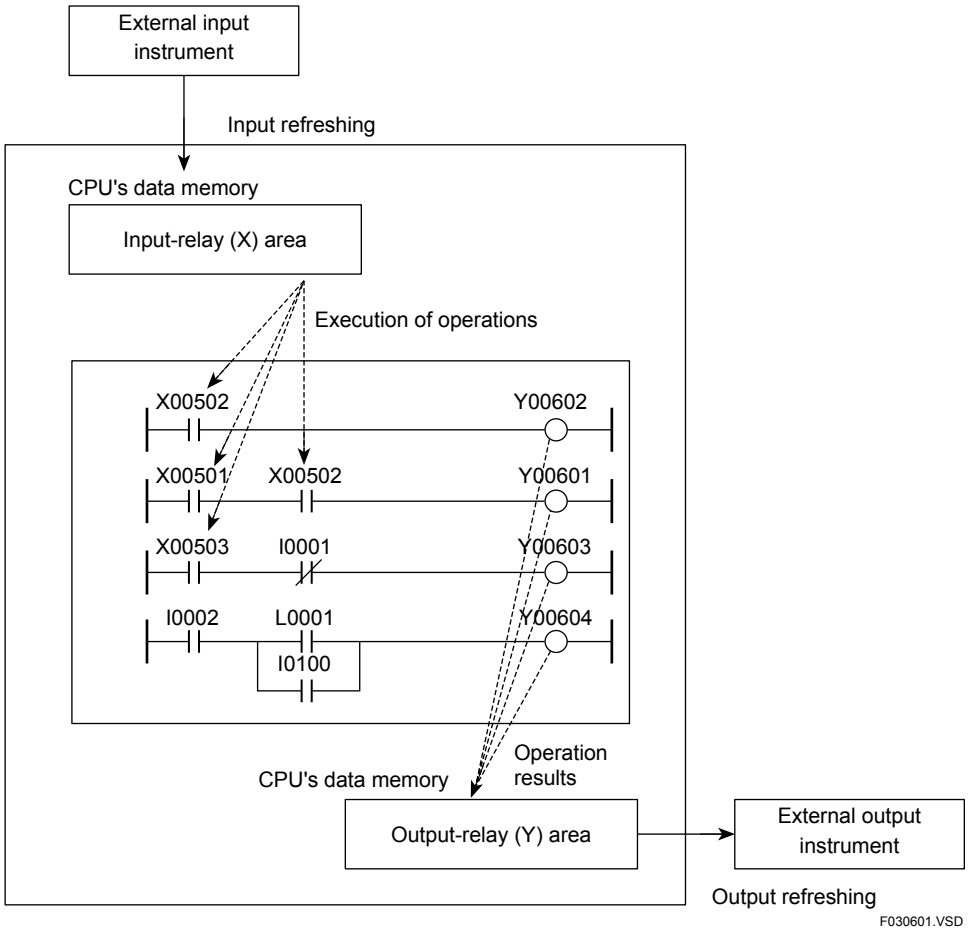


Figure 3.6 Method of I/O Processing

3.6.2 Response Delay

The maximum response delay of the output module against a change in the input module is two scans. For more information, see Chapter 7, “I/O Response Time Based on Scan Time”.

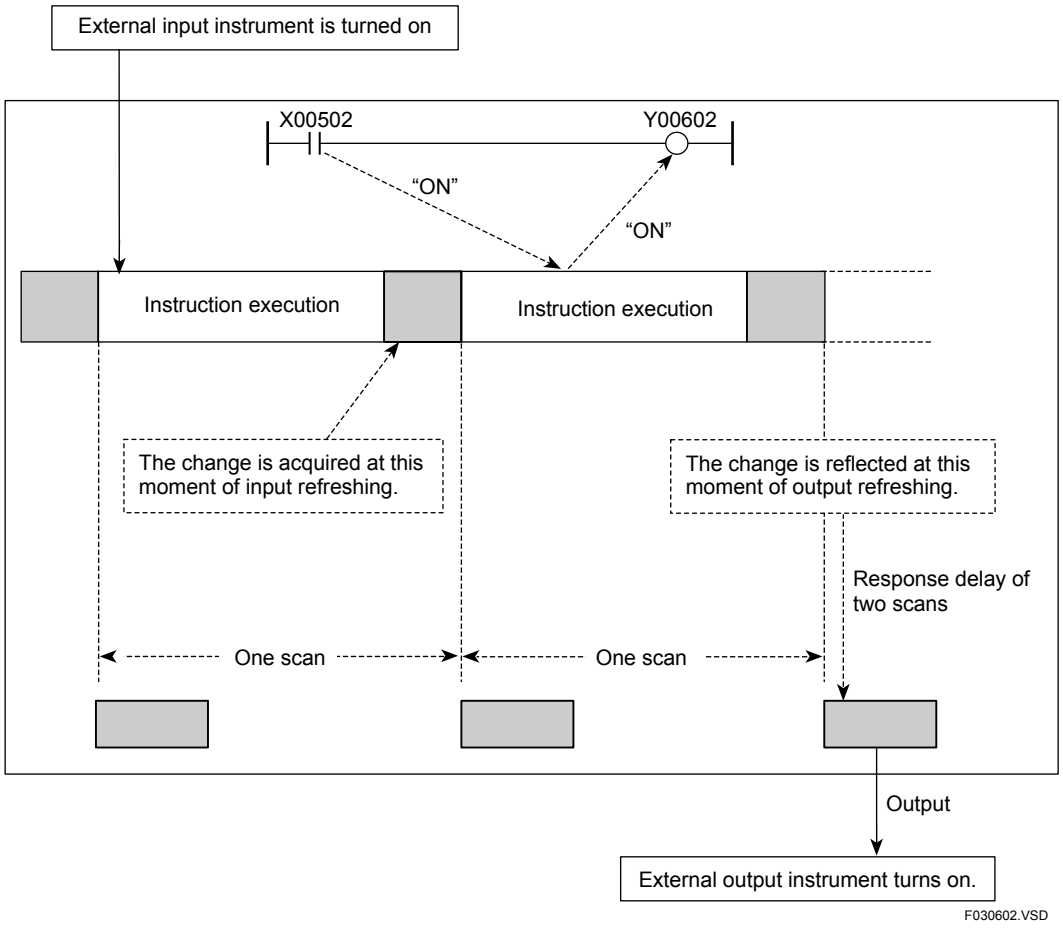


Figure 3.7 Response Delay

3.6.3 I/O Processing in Multi-CPU System

The sequence CPU module performs refreshing when the configuration item “Terminal Usage” of “Input/Output Setup” is set to “Used” or “Use with SCB.”

In the configuration, define the terminals to be refreshed for each sequence CPU module. Each sequence CPU module refreshes the terminals independently according to the definition.

Be careful not to configure the CPUs so that more than one CPU refreshes the same terminal of the output module. Otherwise, the resultant output of the output module will be indefinite.

SEE ALSO

For details on the parameters set for I/O modules and their limitation of use, see Subsection 2.2.2, “Handling I/O Modules in Multi-CPU System” and subsection 4.1.4, “Configuring DIO Modules.”

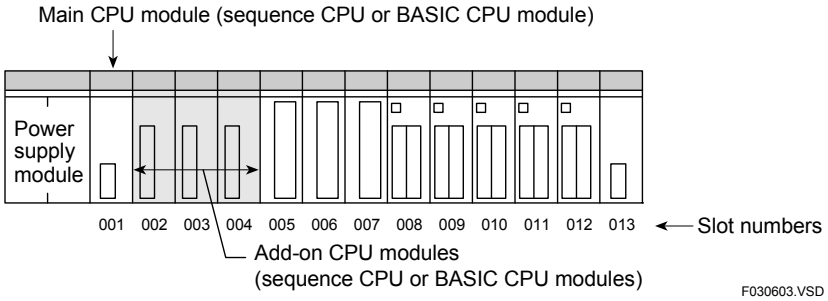


Figure 3.8 Example of Multi-CPU System

3.7 Method of Executing Commands from WideField3

Commands from WideField3 (or WideField2) are executed by the tool service. These commands include downloading and uploading of programs, as well as monitoring of devices.

3.7.1 Tool Service

The tool service executes commands sent from the FA-M3 programming tool WideField3 or WideField2.

Since the tool service runs concurrently to the execution of instructions, it does not affect the scan time.

The CPU does not execute the tool service if there is no command to be processed.

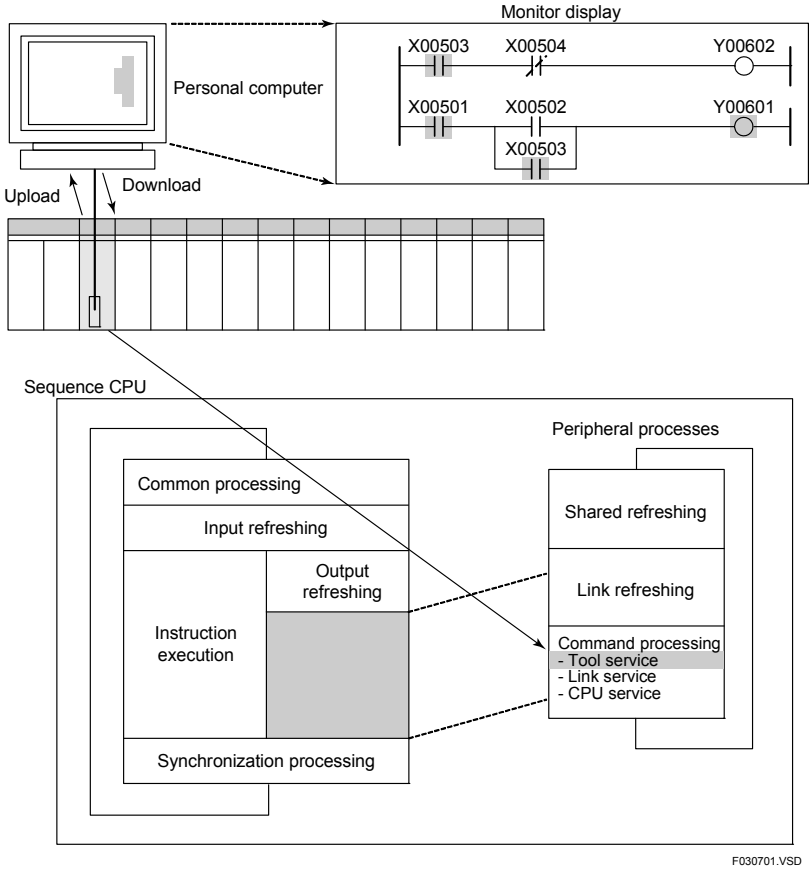


Figure 3.9 Execution of Commands Sent from the WideField3/WideField2

3.8 Method of Executing Commands through Personal Computer Link

The CPU uses the personal computer link service to execute commands sent through the personal computer link. These commands include downloading and uploading programs and reading from and writing to devices.

3.8.1 Personal Computer Link Service

The personal computer link service executes commands sent from a personal computer or a monitor connected to the personal computer link module.

Since the personal computer link service runs concurrently to the execution of instructions, it does not affect the scan time.

The CPU does not execute the personal computer link service if there is no command to be processed.

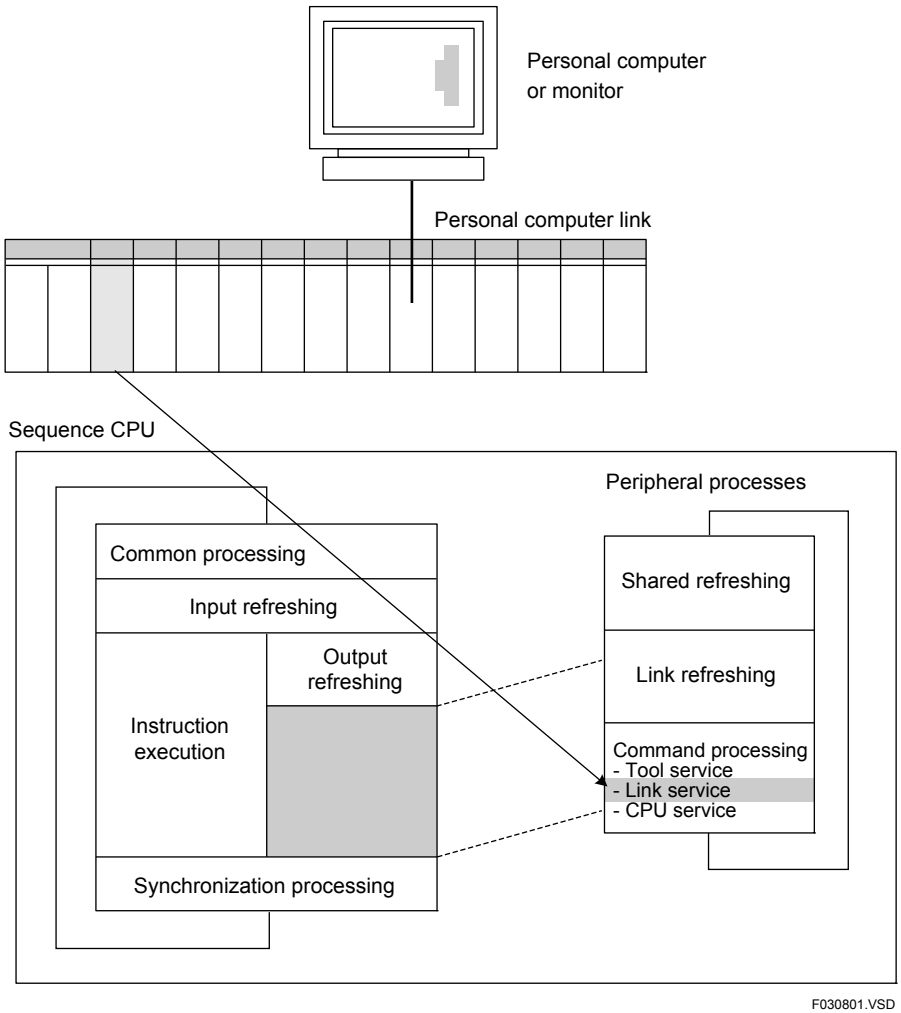


Figure 3.10 Execution of Commands through Personal Computer Link

3.9 Method of CPU-to-CPU Data Communication

CPU-to-CPU communication in a multi-CPU system configured using add-on CPUs is carried out using a shared data communication method and CPU services. Communications between sequence CPU modules is carried out using shared data communications, while communications between sequence CPU modules and other types of CPU modules, such as BASIC CPU modules, is carried out using shared data communications or CPU services.

This section describes methods for updating shared data, configuration of shared refreshing and the CPU service.

3.9.1 Method of Updating Shared Data

CPU-to-CPU data exchange in a multi-CPU system configured using multiple CPU modules is carried out through shared relays (E), extended shared relays (E), shared registers (R) and extended shared registers (R). Hereafter, shared relays (E), extended shared relays (E), shared registers (R) and extended shared registers (R) are collectively referred to as shared devices.

You must configure in advance the range of shared devices to be used for each installed CPU module. The configuration of a CPU module must tally with the configuration of the other CPU modules. You can both read from and write to shared devices within a CPU module's own area. However, you can only read from shared devices within the areas of the other CPU modules.

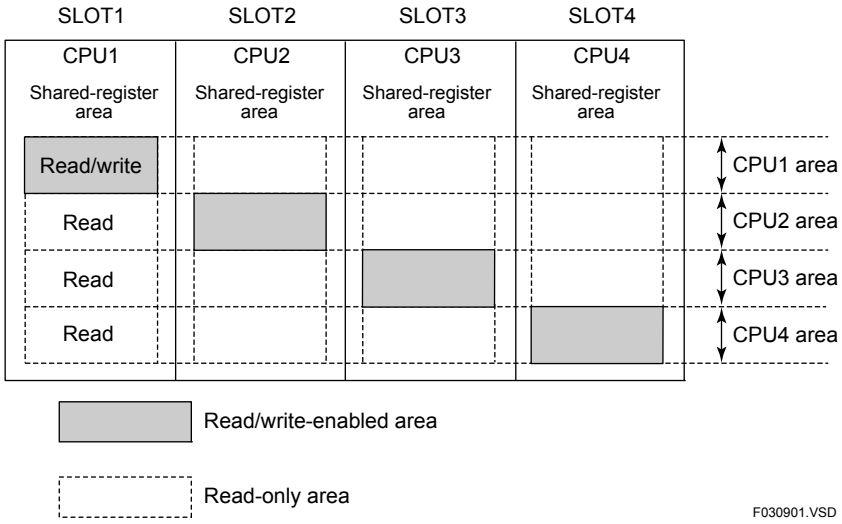


Figure 3.11 Example of Configuring Shared Registers (R)

The figure below shows an example of shared refreshing carried out between a sequence CPU module and an add-on CPU module. In this example, shared relays (E) and shared registers (R) are allocated as shown below.

- Sequence CPU module: (Slot1 CPU) — Shared relays (E)= E0001 to E0512; Shared registers (R) = R0001 to R0256
- Add-on CPU module: (Slot2 CPU) — Shared relays (E)= E0513 to E1024; Shared registers (R) = R0257 to R0512

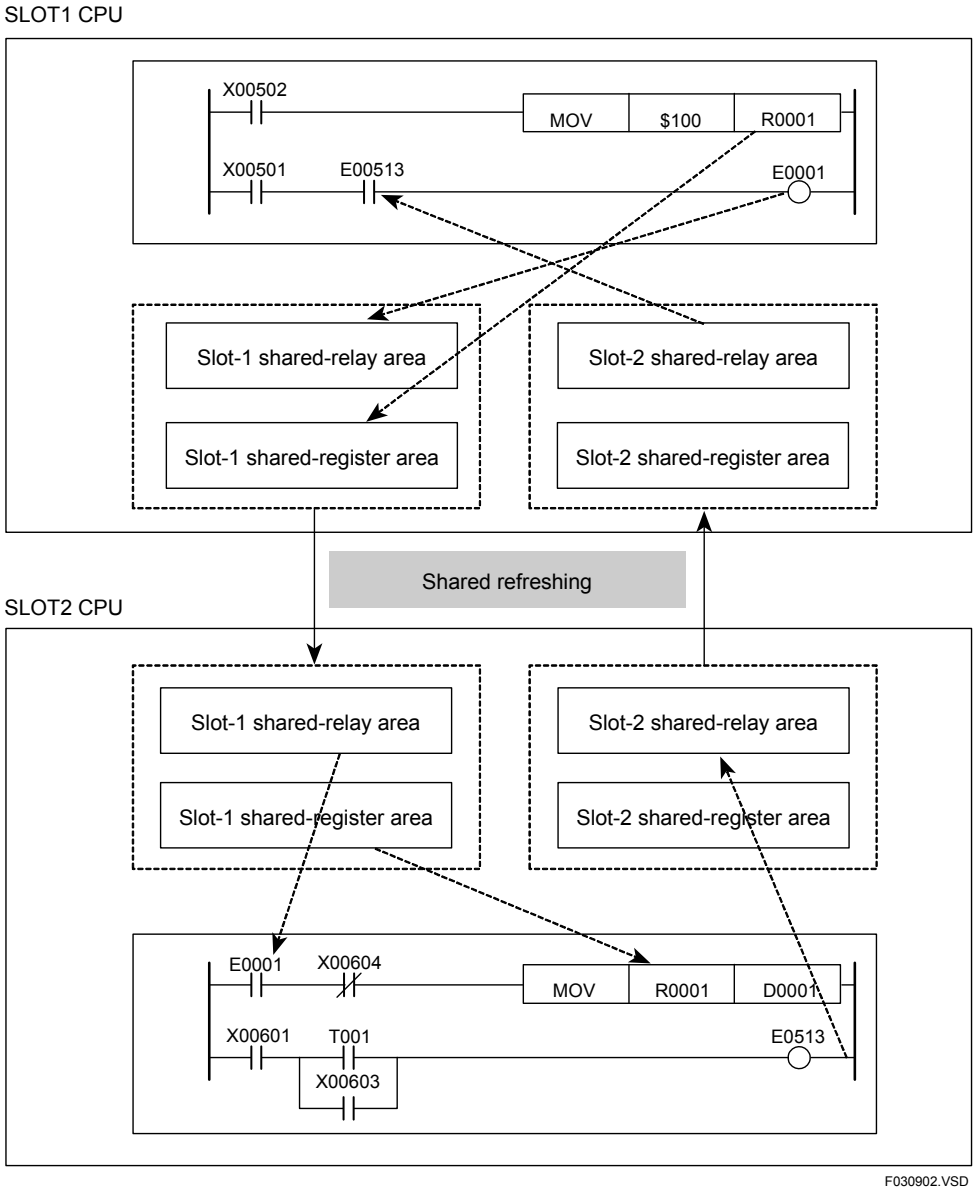


Figure 3.12 Shared Refreshing

3.9.2 Configuration of Shared Refreshing

This subsection describes the shared refreshing range (partial disabling of refreshing), simultaneity of shared refreshed data, and shared refreshing mode (changing to control-related process).

■ Shared Refreshing Range (Partial Disabling of Refreshing)

Using configuration, you can disable shared refreshing for selected device types of shared relay (E), extended shared relay (E), shared register (R) and extended shared register (R) of each CPU module. Disabling shared refreshing between CPU modules that do not need to exchange data shortens the overall shared refreshing interval.

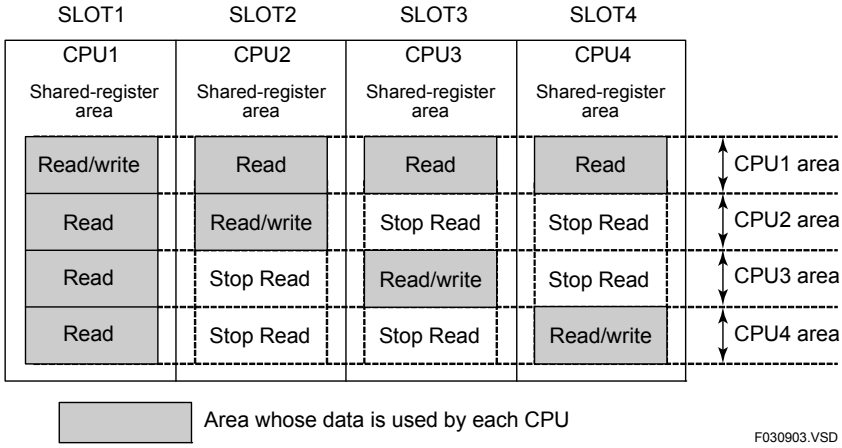


Figure 3.13 Example of Shared Refreshing Configuration

In the example shown in the figure above, if data need not be shared among add-on CPU modules, the refreshing intervals of CPU2, CPU3 and CPU4 are shortened if “CPU3 and CPU4,” “CPU2 and CPU4” and “CPU2 and CPU3” respectively are excluded from shared refreshing.

TIP

If you exclude a CPU module from shared refreshing, its scan time shortens because data updating done by its synchronization process is disabled. However, this prohibits sharing of data in all areas of the other CPU modules.

■ Simultaneity of Shared Refreshed Data

You can specify by configuration whether to maintain simultaneity of shared refreshed data. If you select “Simultaneous” for this configuration item when a sequence CPU module (F3SP22, F3SP28, F3SP38, F3SP53, F3SP58, F3SP59, F3SP66, F3SP67, F3SP71 or F3SP76 module) is combined with any number of F3SP22, F3SP28, F3SP38, F3SP53, FS3P58, F3SP59 F3SP66, F3SP67, F3SP71 and F3SP76 modules defined as add-on CPU modules, simultaneity of shared refreshed data is guaranteed by the unit of shared devices (shared relays (E) and registers (R), or extended shared relays (E) and registers (R)) being refreshed.

If anyone of the F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 and F3SP59 modules is combined with any of the F3SP21, F3SP25, F3SP35 and F3BP□□ modules, simultaneity of data is not guaranteed irrespective of the configuration settings.

The “Non-simultaneous” option of this configuration item is intended for compatibility with the F3SP21, F3SP25 and F3SP35 modules. Select this option when replacing these modules with the F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 or F3SP59 modules.

■ Shared Refreshing Mode (Changing to Control-related Process)

You can change by configuration the mode of shared refreshing, which updates the data of relays (E) and registers (R) shared with other CPUs, so that it runs as a control-related process. Run shared refreshing as a peripheral process if a short scan time is important. Alternatively, run it as a control-related process if the speed of exchanging shared data is important.

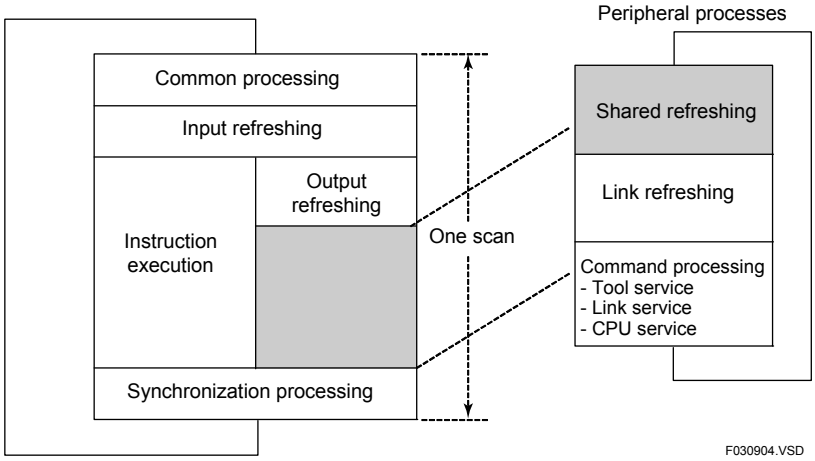


Figure 3.14 Executing Shared Refreshing as a Peripheral Process

Executing shared refreshing as a peripheral process reduces its effects on scanning.

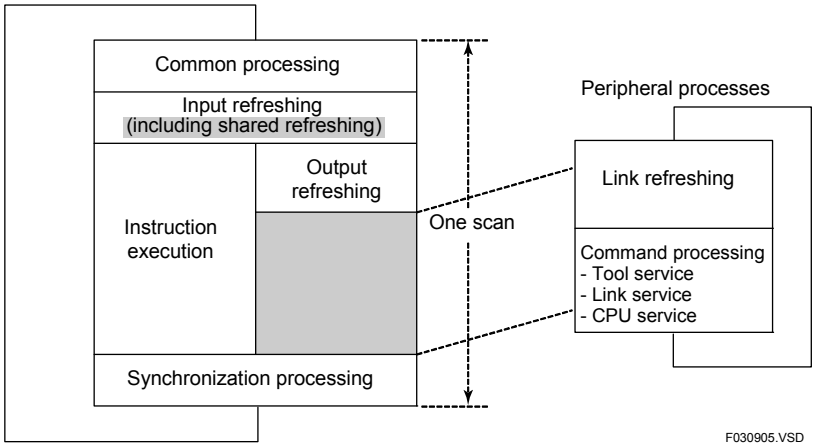


Figure 3.15 Executing Shared Refreshing as a Control-related Process

If you execute shared refreshing as a control-related process, the scan time lengthens. However, this ensures that shared refreshing is not affected by link refreshing or the command processing time.

TIP

- Sequence of Shared Refreshing
  For a main CPU, shared refreshing is executed in the order of CPU2's shared relays(E)/registers(R), CPU2's extended shared relays(E)/registers(R), CPU3's shared relays(E)/registers(R), CPU3's extended shared relays(E)/registers(R), CPU4's shared relays(E)/registers(R), and CPU4's extended shared relays(E)/registers(R).
- When the configuration item "Shared Refreshing Execution" is set to "Peripheral Process"
  Each single scan of peripheral processing refreshes CPU N's shared relays(E)/registers(R) or extended shared relays(E) /registers(R). The data that has been read is reflected in device areas during the synchronization process occurring after the completion of shared refreshing. Note however that if the configuration item "Shared Refreshing Data" is set to "Simultaneous," refreshing may be delayed by as much as three scans of peripheral processing due to the need for synchronization with the CPU N.
- When the configuration item "Shared Refreshing Execution" is set to "Control Process"
  Each single scan of control-related processing refreshes the CPU N's shared relays(E)/registers(R) or extended shared relays(E)/registers(R). Note however that when the configuration item "Shared Refreshing Data" is set to "Simultaneous," refreshing may be delayed by as much as three scans due to the need for synchronization with the CPU N.
- Reference to a CPU's Own Write Area
  You can read data in a CPU's write area from other CPUs. That is, you can read the data alternately from shared relays(E)/registers(R) and from extended shared relays(E)/registers(R) in that area during the synchronization process of each scan. Note however that when the configuration item "Shared Refreshing Data" is set to "Simultaneous" for any of the other CPUs, refreshing may be delayed by as much as the longest of those CPUs' scans due to the need for synchronization with the slowest CPU.

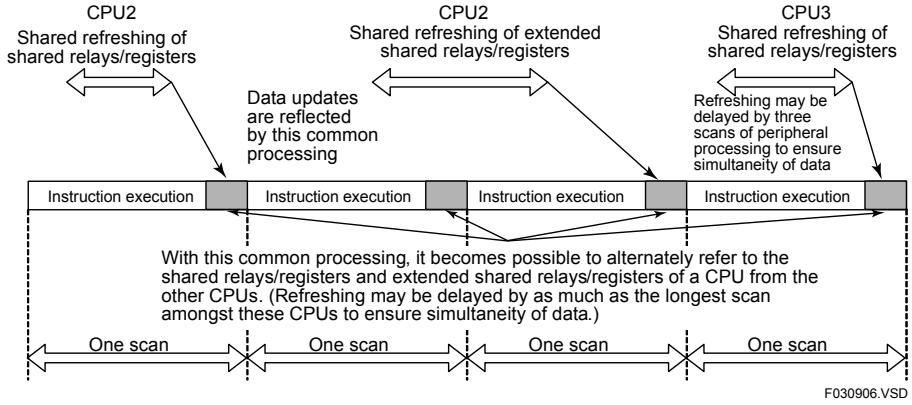


Figure 3.16 Shared Refreshing as a Peripheral Process

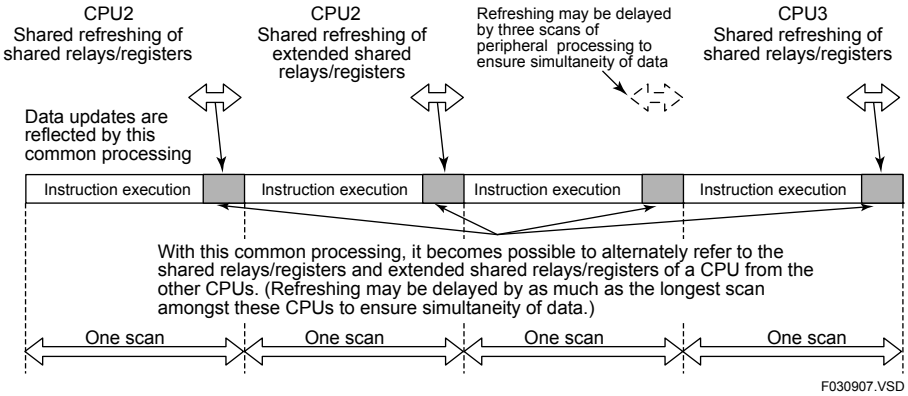


Figure 3.17 Shared Refreshing as a Control-related Process

SEE ALSO

Tables 3.3 and 3.4 show examples of how shared refreshing affects the scan time. For more information, see Section 7.1, “Description of Scan Time.”

These examples assume that both CPU1 and CPU2 have 512 shared relays and 512 shared registers.

Table 3.3 Durations of Interference with the Scan Time by Shared Refreshing Running as a Peripheral Process

Other Sequence CPU Module	Duration of Shared Refreshing	Duration of Synchronization Processing	Duration of Interference with Scan Time (= Duration of Synchronization Processing)	Duration of Interference with Scan Time of Peripheral Processing (=Duration of Shared Refreshing)
F3SP22/28/38/53/58 /59	0.916ms	1.138ms	1.138ms	0.916ms
F3SP21/25/35	3.908ms	1.138ms	1.138ms	3.908ms

Table 3.4 Durations of Interference with the Scan Time by Shared Refreshing Running as a Control-related Process

Other Sequence CPU Module	Duration of Shared Refreshing	Duration of Synchronization Processing	Duration of Interference with Scan Time (= Duration of Extended Shared Refreshing plus Duration of Shared Refreshing)	Duration of Interference with Scan Time of Peripheral Processing
F3SP22/28/38/53/58 /59	0.916ms	0.866ms	1.782ms	0ms
F3SP21/25/35	3.908ms	0.866ms	4.774ms	0ms

3.9.3 CPU Service

CPU service exchanges data and process commands between the sequence CPU and a BASIC CPU.

CPU service is processed concurrently with instruction execution so it does not affect the scan time. The sequence CPU does not execute the CPU service unless it receives a command (ENTER, OUTPUT, etc.) to be processed from a BASIC CPU.

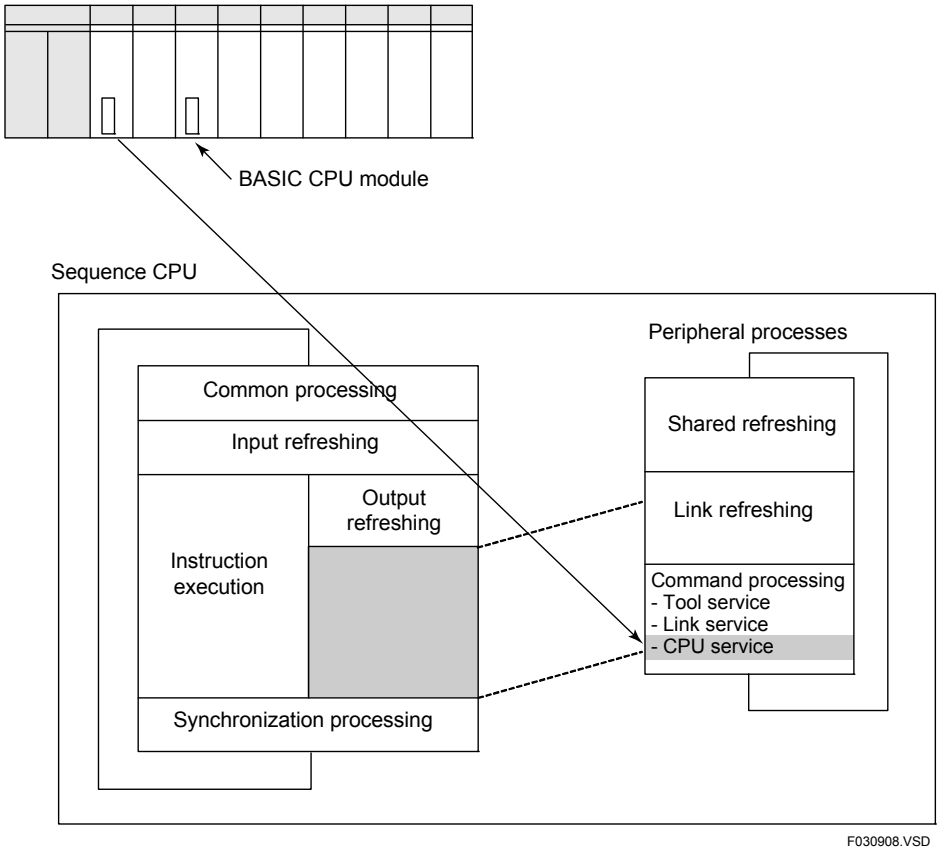


Figure 3.18 CPU Service

3.10 Method of Link Data Updating

This section describes methods of link data updating and link refreshing for FA link systems and FL-net systems.

3.10.1 Link Data Updating

Link data updating is a process of exchanging data with sequence CPU modules in remote stations through link relays (L) and registers (W).

You must configure in advance the ranges of link relays (L) and registers (W) to which data is written in the local and remote stations.

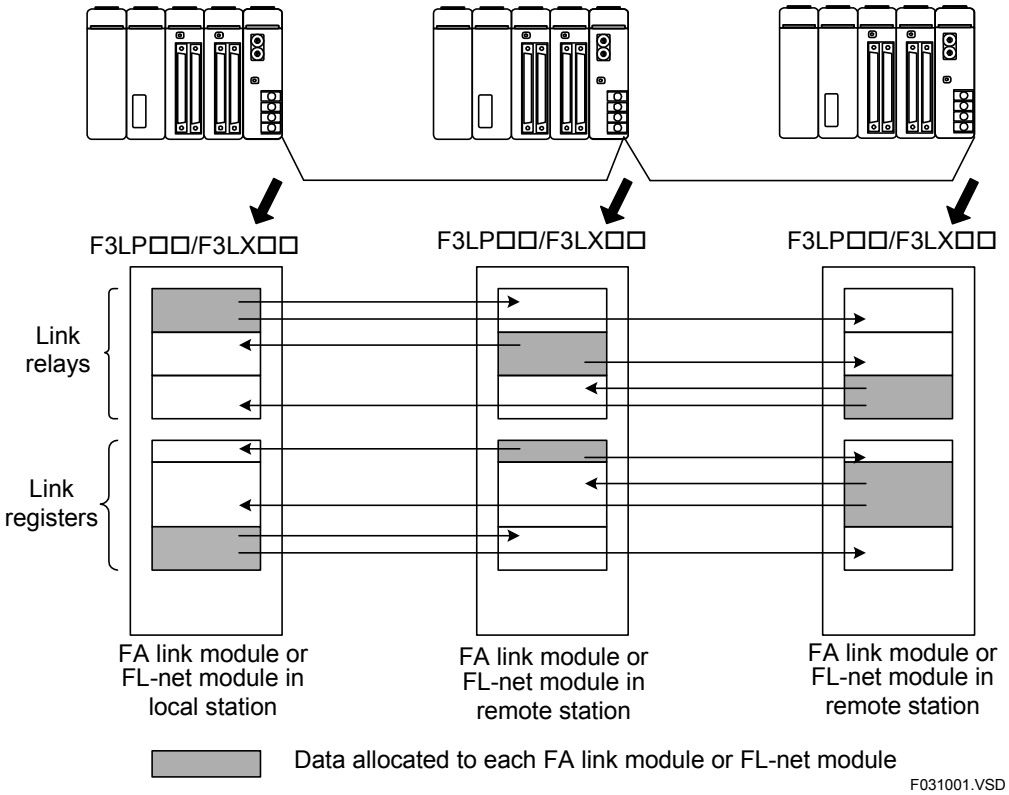


Figure 3.19 Link Data Updating

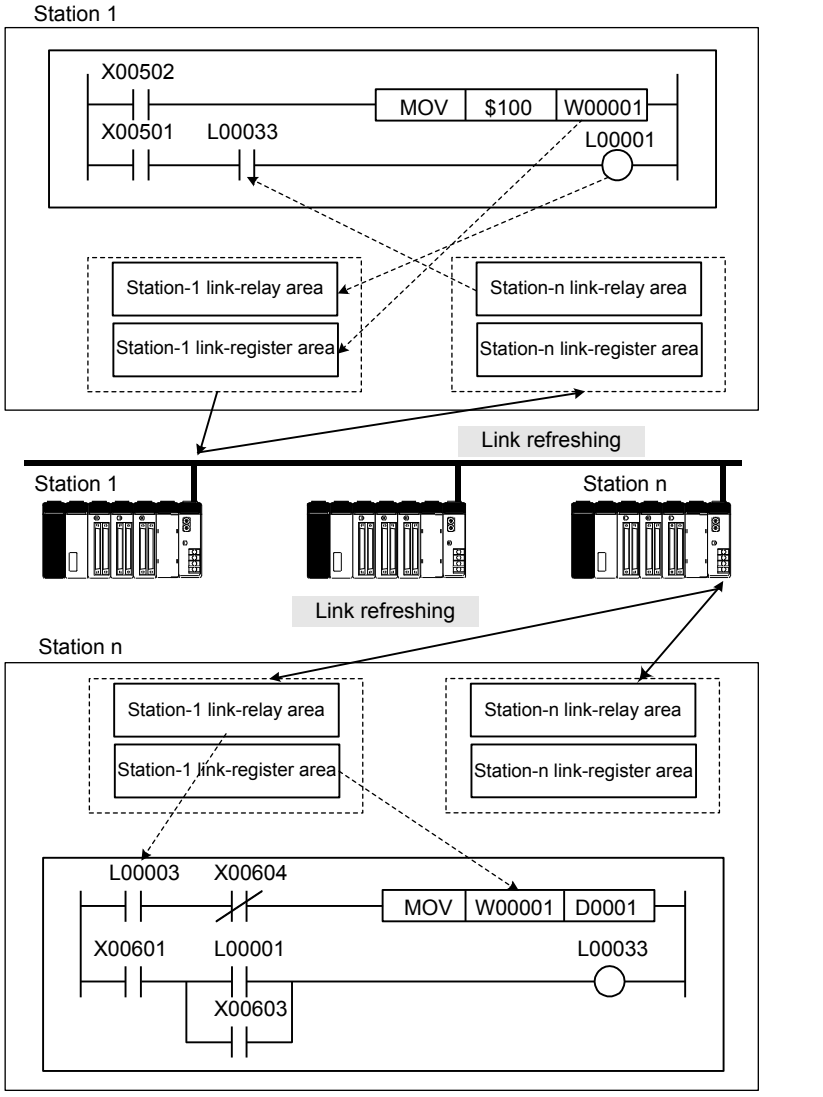
SEE ALSO

For details on link data updating and link refreshing, see “FA Link H Module, Fiber-optic FA Link H Modules” (IM 34M06H43-01E) and “FL-net (OPCN-2) Interface Module” (IM 34M06H32-02E).

3.10.2 Link Refreshing

Link refreshing reads data from or writes data to devices such as link relays (L) and registers (W) of the sequence CPU module via an FA link module or FL-net (OPCN-2) interface module installed in the local unit. It maps the link data in the storage area of the sequence CPU module to those of the FA link module.

The sequence CPU module reads the link data of the FA link module or FL-net (OPCN-2) interface module automatically so data communication is transparent to a user.



F031002.VSD

Figure 3.20 Link Refreshing

■ Execution of FA Link Refreshing

FA link refreshing is executed in peripheral processing.

Link refreshing runs concurrently with instruction execution so it does not affect the scan time.

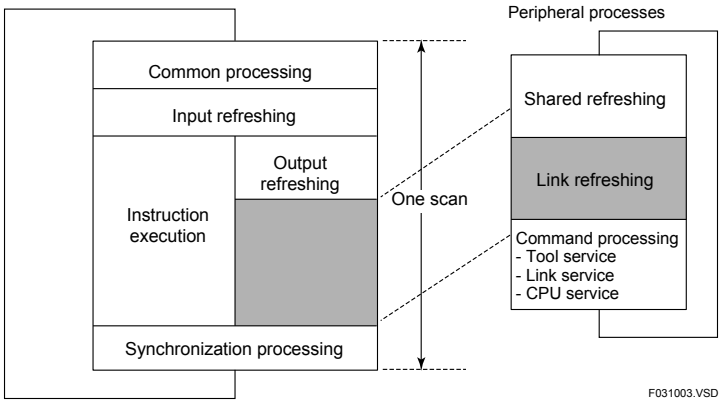


Figure 3.21 Executing FA Link Refreshing as a Peripheral Process

Link refreshing updates the link relays (L)/registers (W) of FA link 1 to FA link 8 in each cycle of peripheral processing.

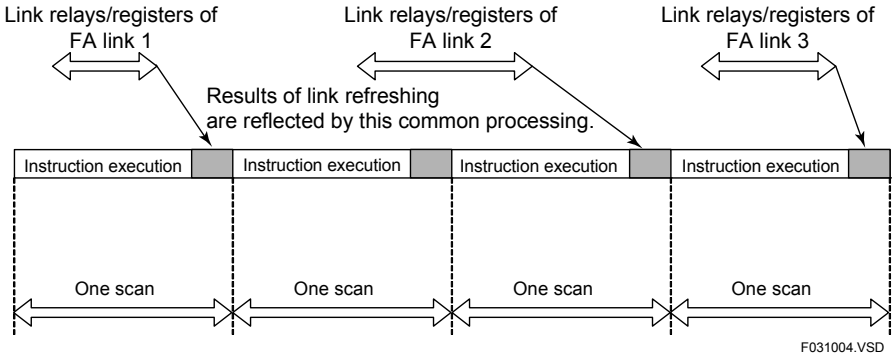


Figure 3.22 FA Link Refreshing Sequence

TIP

Table 3.5 shows an example of how link refreshing affects the scan time. For more information, see Section 7.1, “Description of Scan Time.”

Table 3.5 Durations of Interference by Link Refreshing with the Scan Time

	Number of Link Devices	Duration of Interference with Scan Time	Duration of Interference with Scan Time of Peripheral Processing
Example 1	Link relay (L) = 1024 units Link register (W) = 1024 units	3.314 ms	16.38 ms
Example 2	Link relay (L) = 2048 units Link register (W) = 2048 units	6.578 ms	32.7 ms

■ Execution of FL-net Link Refreshing

You can specify by configuration whether to execute FL-net link refreshing as a peripheral process or a control-related process.

Link refreshing executed as a peripheral process does not affect the scan time.

Executing link refreshing as a control-related process may lengthen the scan time but it ensures that link refreshing is not affected by the shared refreshing or command processing time.

Include link refreshing in peripheral processes if a short scan time is important. Alternatively, include it in control-related processes if the speed of exchanging link data is important.

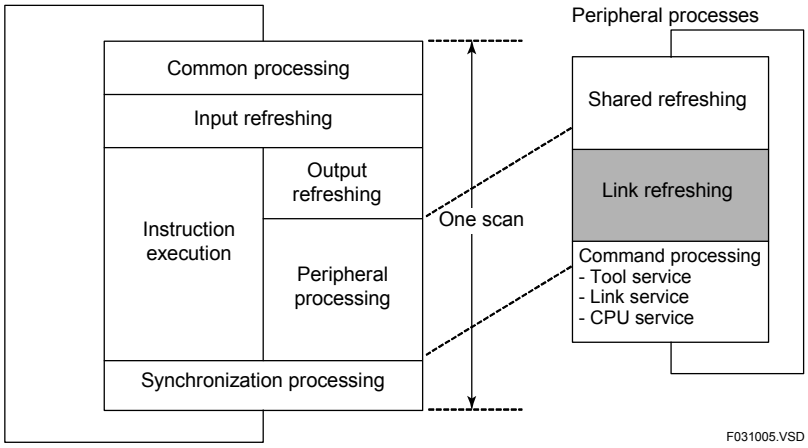


Figure 3.23 Executing FL-net Link Refreshing as a Peripheral Process

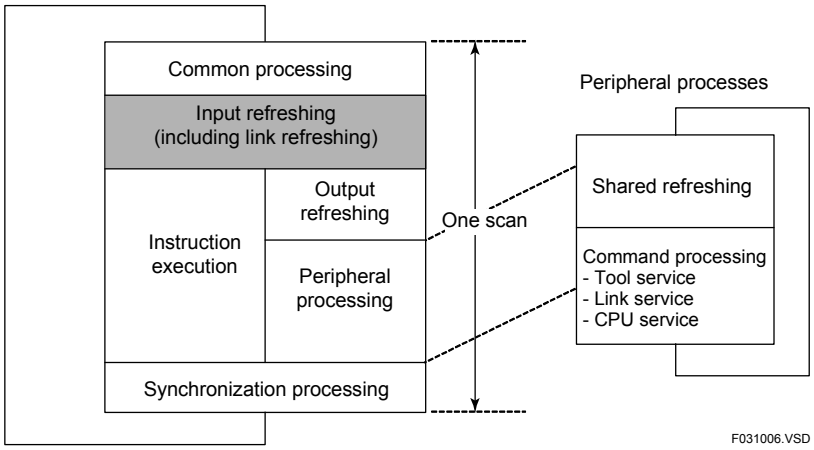


Figure 3.24 Executing FL-net Link Refreshing as a Control-related Process

When executed as a peripheral process, link refreshing updates the link relays (L) and link registers (W) of system 1 or system 2 in each cycle of peripheral processing.

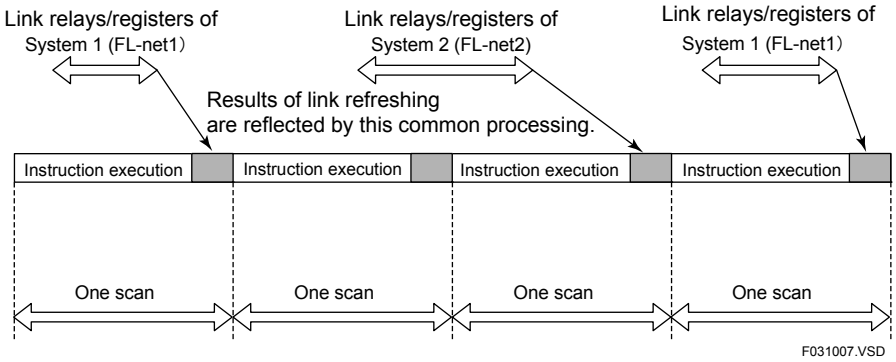


Figure 3.25 FL-net Link Refreshing as a Peripheral Process

When executed as a control-related process, link refreshing updates the link relays (L) and link registers (W) of system 1 or system 2 in each cycle of control processing.

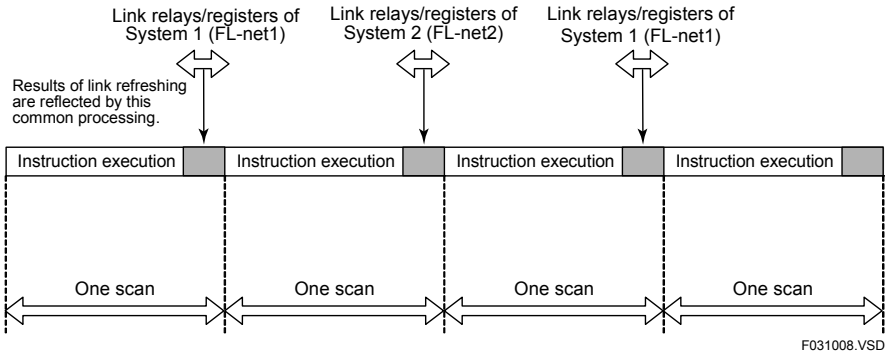


Figure 3.26 Executing FL-net Link Refreshing as a Control-related Process

TIP

Table 3.6 shows an example of how link refreshing affects the scan time. For more information, see Section 7.1, "Description of Scan Time."

Table 3.6 Durations of Interference by FL-net Link Refreshing with the Scan Time

	Number of Link Devices	Duration of Interference with Scan Time	Duration of Interference with Scan Time of Peripheral Processing
Refreshing as control-related process	Link relay (L) = 8192 units Link register (W) = 8192 units	4.652 ms	0 ms
Refreshing as peripheral process	Link relay (L) = 8192 units Link register (W) = 8192 units	3.782 ms	4.652 ms

■ Inter-mixing FA Link Modules and FL-net Interface Modules

Where FA link and FL-net are intermixed in a system configuration, FA link refreshing and FL-net link refreshing are executed independently of each other.

If FL-net link refreshing is configured as a peripheral process, FA link refreshing and FL-net link refreshing are executed in each cycle of peripheral processing.

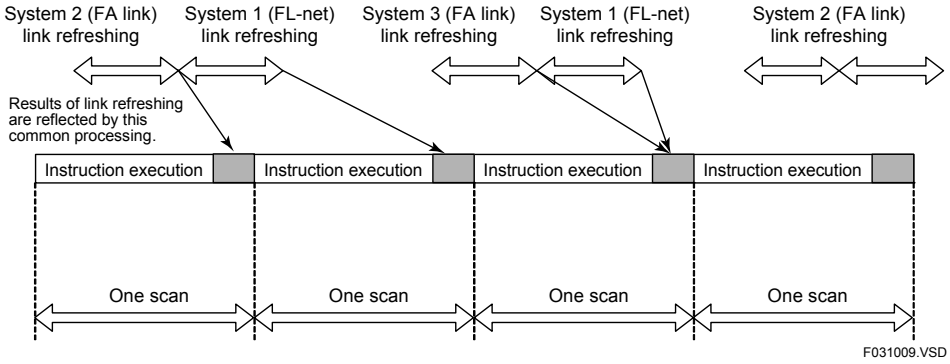


Figure 3.27 Intermixing FA Link and FL-net

Even if FL-net link refreshing is configured as a control-related process, FA link refreshing always remains executed as a peripheral process.

3.11 Method of Input Interrupt Processing

This section describes input interrupt processing, input interrupt processing control, interrupt timing, and priority of interrupts.

3.11.1 Input Interrupt Processing

The sequence CPU module executes an input interrupt program when it detects the rising edge of an interrupt input from an input module.

You can register a maximum of four input interrupt programs with the sequence CPU module using input interrupt instructions (INTP instructions.)

The module can accept a maximum of eight concurrent interrupts. Input interrupt programs are executed in the order of occurrence of their interrupt factors. If any interrupt factor occurs during execution of an input interrupt program, the factor is processed when the input interrupt program completes execution.

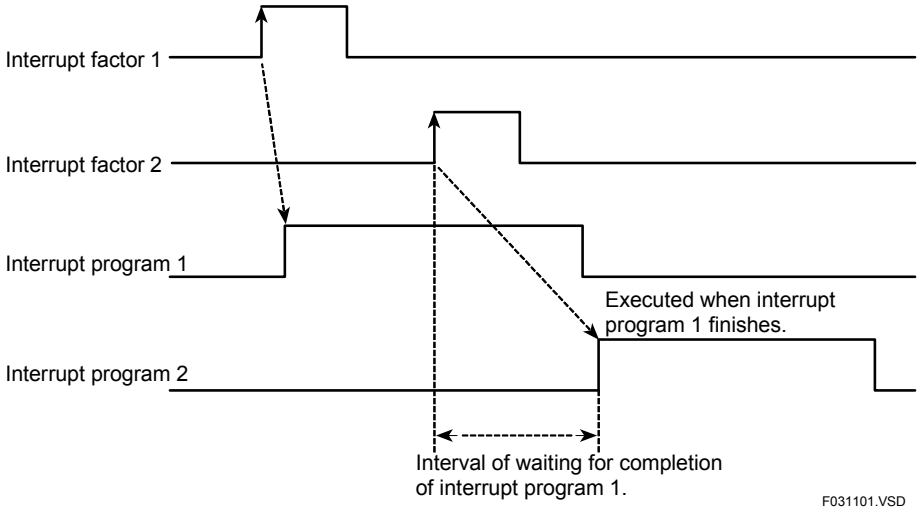


Figure 3.28 Input Interrupt Processing

CAUTION

- Do not register an input interrupt program for an input module with two or more CPU modules. Otherwise, input interrupt processing may fail to be executed.
- Do not use a TIMER instruction in any input interrupt program because the instruction may not work correctly.

3.11.2 Input Interrupt Processing Control

You can control the execution of input interrupt programs by means of programming. Use the Enable Interrupt (EI) instruction and Disable Interrupt (DI) instructions respectively to specify whether to execute (enable) or not execute (disable) an interrupt. Interrupts are enabled by default.

An interrupt that is disabled by a DI instruction continues to be detected by the sequence CPU but its input interrupt program is not executed. Such interrupts are processed in order of their occurrence if and after they are enabled by an EI instruction.

A maximum of eight concurrent interrupts are accepted. The ninth or subsequent concurrent interrupt generates an interrupt error.

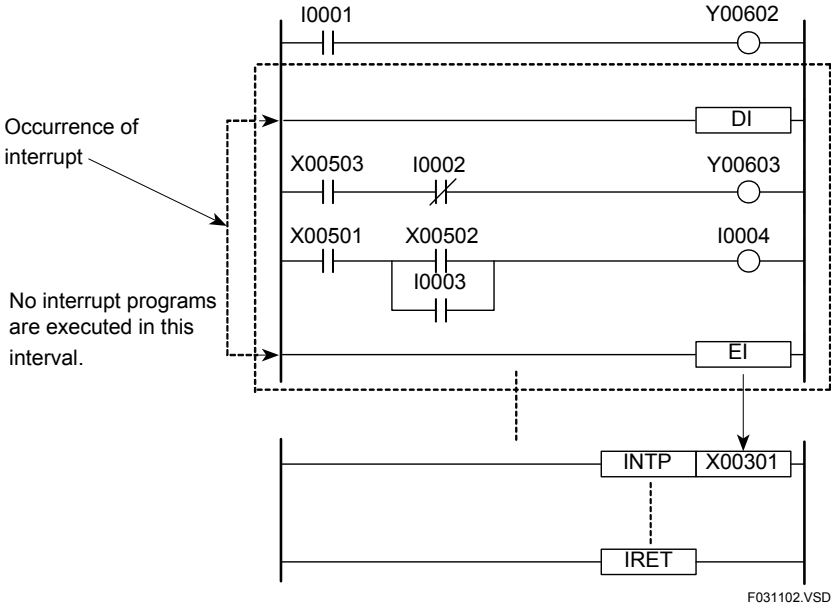


Figure 3.29 Input Interrupt Processing Control

3.11.3 Interrupt Timing

Using the configuration function of WideField3 (or WideField2), you can specify when an interrupt program is to be executed if an interrupt occurs during program execution. The following two options are available.

Table 3.7 Interrupt Timing Options of Input Module Interrupt Processing

Interrupt Timing	Description
After instruction (default)	The sequence CPU switches execution to an input interrupt program (program code between INTP and IRET instructions) after it finishes executing a ladder instruction. This switching does not take place, however, during synchronization processing, common processing or input refreshing.
Immediate (during instruction execution)	The sequence CPU switches execution to an input interrupt program (program code between INTP and IRET instructions) during execution of a ladder instruction. This switching takes place even during synchronization processing, common processing or input refreshing.

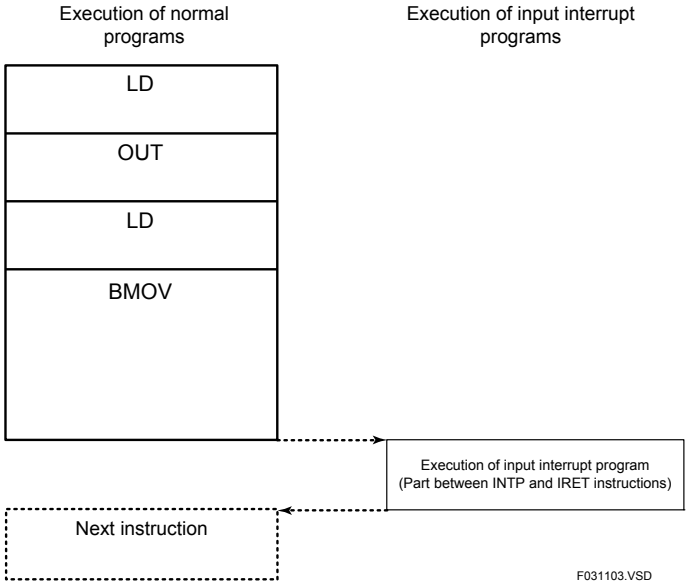


Figure 3.30 Execution of Input Interrupt Program after the Completion of Instruction Execution

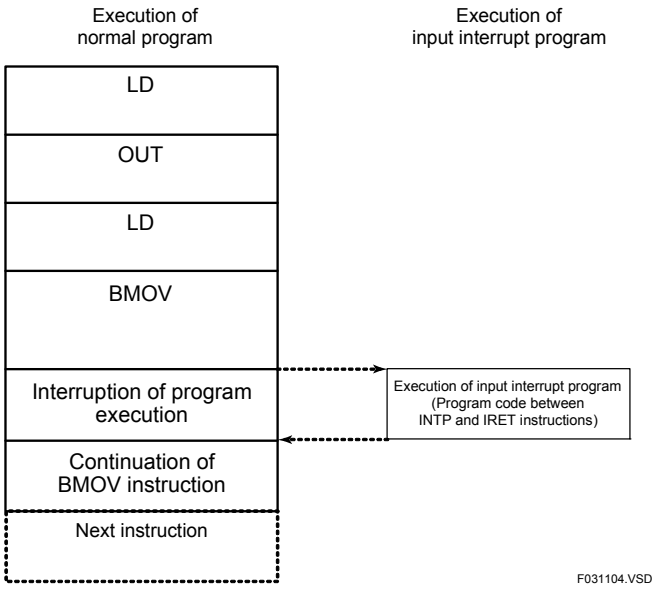


Figure 3.31 Immediate Execution of Input Interrupt Program during Instruction Execution

The characteristics of these two interrupt timing options are summarized in the table below.

Table 3.8 Characteristics of Interrupt Timing Options

Item	Execution of Input Interrupt Program after the Completion of Instruction Execution	Immediate Execution of Input Interrupt Program during Instruction Execution
Execution delay *1	"Processing time of instruction being executed *2 + switching time *3," or "synchronization processing time *4 + common processing time + input refreshing time *4 + switching time *3 "	Switching time only *3
Simultaneity of data	Guaranteed on an instruction basis	None for multiple devices

*1: The indicated time does not include the response time of an input module. For details on the response time of input modules, see "Hardware Manual" (IM 34M06C11-01E).
*2: For details on the instruction processing time, see the appendix of "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E),
*3: 120 μs for F3SP22, F3SP28 and F3SP38 modules and 100 μs for F3SP53, F3SP58 and F3SP59 modules.
*4: See Section 7.1, "Description of Scan Time."

CAUTION

- **Output of data to relays with input interrupt programs executed immediately during instruction execution**

  Be careful when outputting data to relays using an output instruction (e.g. OUT, SET or RST) if input interrupt processing is configured with the timing option of "Immediate (during instruction execution)." In such cases, do not output data in a normal-scan program to any of the relays numbered 1 to 16 if data is output to any of these relays in an input interrupt program (there is no limitation on data input, however).

  Example: Input interrupt program: OUT I2
  Normal-scan program: OUT I1 - Not allowed.
  OUT I17 - Allowed.

  The same rule applies to relays numbered 17 to 32, 33 to 48, 49 to 64, and so on. If both the interrupt program and normal-scan program output to these relays, no output may be generated.

- **Simultaneity of multi-device data when input interrupt programs are executed immediately during instruction execution**

  Simultaneity of data for multiple devices is not guaranteed if input interrupt programs are executed immediately during instruction execution.
  Simultaneity of data is required when data of multiple devices is exchanged between a normal-scan program and an input interrupt program using a block transfer (BMOV) instruction, a long-word instruction with IEEE single-precision floating point data, or two or more instructions.
  For example, consider the case shown in Figure 3.30 where an input interrupt program is executed during execution of a block transfer (BMOV) instruction in a normal scan. There is a risk that block data partially transferred may be overwritten after the execution of the input interrupt program.
  If simultaneity of data is required when interrupt timing is configured as "Immediate (during instruction execution)," use any of the following means to ensure data simultaneity:

  1. Use a Disable Interrupt (DI) instruction and an Enable Interrupt (EI) instruction to prevent all interrupt programs from being executed during exchange of multi-device data.
  2. Write an application program to perform flag control between the normal-scan program and the input interrupt program using relays.

CAUTION

- **Simultaneity of refreshed data when input interrupt programs are executed immediately during instruction execution**
  If input interrupt processing is configured with the timing option of “Immediately (during instruction execution),” an input interrupt program may be executed even during synchronization processing, input refreshing and common processing.
  When an input interrupt program is executed during synchronization processing or input refreshing, values of devices (I/O relays (X/Y), shared and extended shared relays (E), shared and extended shared registers(R), and link relays and registers (L/W)) which are being refreshed may be read by programs. If these device values are overwritten by the input interrupt program, simultaneity of data before and after the execution of the input interrupt program is lost.
  To prevent all input interrupt programs from being executed during synchronization processing, input refreshing and common processing, execute a Disable Interrupt (DI) instruction at the end of a normal-scan program. Along with this instruction, execute an Enable Interrupt (EI) instruction at the start of the normal-scan program.

3.11.4 Priority of Interrupts

You can specify the priority of interrupts by configuration using WideField3 (or WideField2) (“Priority of Interrupts” of “Interrupt Setup”) for conflict resolution in the event that input interrupt processing coincides with an interrupt from a sensor control block.

The table below lists the two options for “Priority of Interrupts”, along with how they work.

Table 3.9 Options for Priority of Interrupts

Priority of Interrupts	Functionality	
	When an interrupt from an input module occurs during execution of a sensor control block	When the time for executing a sensor control block arrives during interrupt processing
Sensor CB interrupt has priority (default)	Suspends the interrupt process after executing the sensor control block.	Suspends the interrupt process and resumes execution after executing the sensor control block.
Input interrupt has priority	Suspends the execution of the sensor control block and resumes execution after executing the interrupt process.	Executes the sensor control block after executing the interrupt process.

CAUTION

The sequence CPU applies the rule of interrupt execution timing (after completion of instruction execution or immediately during instruction execution) discussed earlier, even in the case where execution of the sensor control block or interrupt process is suspended due to priority of interrupts.

4. Devices

This chapter describes the types and functions of devices available with the sequence CPU modules.
Relay devices are accessed on a one-bit basis. Thus a relay device number corresponds to a bit.
Register devices are accessed on a 16-bit basis. Thus a register device number corresponds to 16 bits.

4.1 I/O Relays (X/Y)

I/O relays (X/Y) are devices used to exchange data with external equipment.

I/O relay (X/Y) numbers are determined by the position of the slot where an I/O module is installed. They are fixed, discontinuous numbers and are assigned on 64-relay basis for each slot.

The input relay (X) numbers never coincide with any of the output relay (Y) numbers. Data held in the I/O relays is not retained when the power is turned off.

For more information on I/O relay (X/Y) number definitions, refer to Section 1.3, “Basic Configuration.”

4.1.1 Input Relays (X)

Input relays are used to input the ON and OFF states of external equipment, such as pushbuttons and limit switches.

In programs, you can use these relays for contacts **a** and **b** and application instructions.

Input relay numbers are coded as X,L mmnn, where:

L mm = Slot number
L = Unit number (0 to 7)
mm = Slot position (01 to 16)
nn = Terminal number (1 to 64)

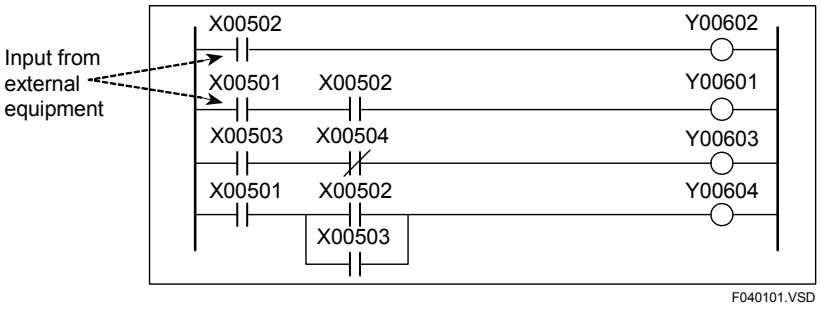


Figure 4.1 Input Relays (X)

4.1.2 Output Relays (Y)

Output relays are used to output the results of program-based control to external equipment, such as actuators. In programs, you can use these relays, for example, for contacts **a** and **b**, coils and application instructions.

Output relay numbers are represented as Y, Lmmnn, where:

Lmm = Slot number

L = Unit number (0 to 7)

mm = Slot position (01 to 16)

nn = Terminal number (1 to 64)

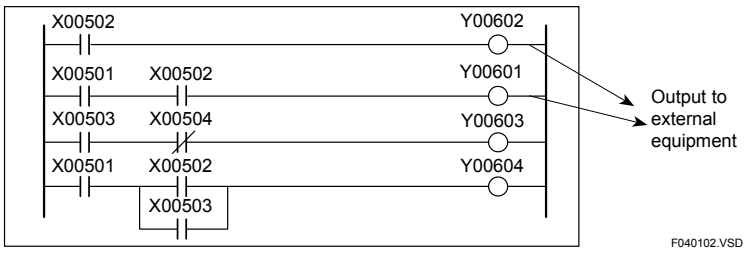


Figure 4.2 Output Relays (Y)

4.1.3 Allocation of I/O Addresses

There is no need to allocate I/O address through WideField3 (or WideField2).

I/O relay numbers are determined by the position of the slot where an I/O module is installed. They are fixed, discontinuous numbers and assigned on 64-relay basis for each slot. An empty slot is regarded as being equivalent to 64 relays.

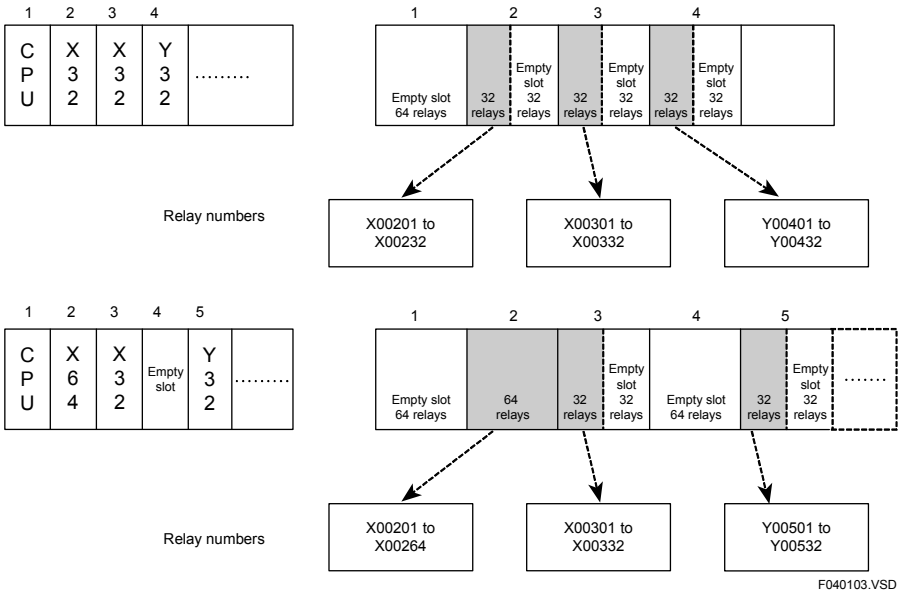


Figure 4.3 Allocation of I/O Addresses

4.1.4 Configuring DIO Modules

This section describes settings for terminal usage (use/not used/sensor control block), data code (BIN/BCD), input sampling interval (16 ms/1.0 ms/250 μs/62.5 μs/Always), and holding/resetting output relays when the program stops.

■ Specifying Terminal Usage

Using the configuration function, select one of the three options, “Use,” “Use with SCB,” and “Not Used” to specify whether the I/O module is used in programs, or used in the sensor control block, or not used at all. In this selection, configure the I/O module on 16-point basis (see the second caution below, when the selected option is “Use with SCB”). Configure special modules containing I/O relays (X/Y) in the same way as discussed here.

I/O relays that are included in the option “Not Used” are not refreshed at all. By default, all I/O modules are set to the option “Use.”

SEE ALSO

For details on the sensor control block, see Section 6.15.

CAUTION

- **Precautions when setting “Terminal Usage” in Input/Output Setup**

  Configure the input module on long word basis (i.e., 32 relays, 32 terminals, terminals 1 to 32 or terminals 33 to 64); and the output module on word basis (i.e., 16 relays or 16 terminals).
  If you set either of terminals 1 to 16 or terminals 17 to 32 to "Used" and the other to "Not Used", all terminals 1 to 32 are refreshed on long word basis.
  Let's suppose you configured the input module incorrectly on word basis (for example, you set terminals 1 to 16 to the option “Used” [with normal scan] and terminals 17 to 32 to the option “Use with SCB”). Since input refreshing is performed on long word basis, input (X) relays used under a normal scan are refreshed by the Refresh instruction of the sensor control block when the normal scan is in progress. Consequently, the simultaneity of data is not guaranteed before and after the refreshing. Simultaneity of data is also not guaranteed for input (X) relays used in the sensor control block.

- **When using output modules and special modules with Y□□□□□ output relays (Y) in a multi-CPU system configuration**

  - Combination of F3SP22, F3SP28, F3SP38, F3SP53, F3SP58, F3SP59, F3SP66, F3SP67, F3SP71 and F3SP76 CPU Modules
    You can output data from multiple sequence CPU modules separately to the output relays (Y) of the same output module on 16-relay basis. To do this, set the unused output terminals to the option “Not Used” on 16-terminal basis.

  - Other Combinations of CPU Modules
    You may not use the same output module with multiple CPUs. Configure all CPUs that do not use the output module so that the output module is set to “Not Used.”

CAUTION

- **When using a Direct Refresh (DREF) instruction**

  Set the output relays (Y) to be refreshed by a DREF instruction of a program to the option “Not Used.”

  If you set them to the option “Use” or “Use with SCB,” the values one scan earlier may be overwritten with the values output by the DREF instruction because of the timing of output refreshing, which is executed concurrently with instructions.

■ Specifying Data Code

Specify whether data held in I/O relays (X/Y) should be handled as "BIN" data or “BCD" data when they are used in a Compare, Arithmetic or Move instruction.

All internal operations are based on BIN data. For this reason, if you set the data code of an I/O relay to "BCD,” data is automatically converted from BCD to BIN for an input relay and from BIN to BCD for an output relay.

This option enables you to handle data easily, without worrying about the data representation during programming, especially in cases where data handled by external equipment are in BCD data code.

By default, I/O relays of all I/O modules are handled as “BIN” data. You can specify the data code on 16-relay basis.

■ Specifying Input Sampling Interval

Set the input sampling interval for input relays of input modules.

Note that this setting is ignored for some input modules. For details, refer to the data item “response time” in the specifications section of individual input modules given in "Hardware Manual" (IM 34M06C11-01E).

You can select from five options, namely, “16 ms,” “1.0 ms,” “250 µs,” “62.5 µs” and “Always.” By default, all input modules are set to “16 ms.” You can specify the sampling interval on 16-relay basis.

CAUTION

If a single input module (or special module with input relays X□□□□□) is used with two or more CPU modules in a multi-CPU system configuration, configure the CPUs so that they have the same sampling interval for all relays of that input module. (Also re-configure any CPU whose input relays (X) were set to the option “Not Used,” so that it has the same settings as the other CPUs.) Otherwise, system operation may be unstable.

■ Specifying Output When Stopped (Holding/Resetting Output Relays When Sequence Stops)

Specify whether the output relays (Y) of an output module (or special module with output relays Y□□□□□) should be placed in a “Hold” state or “Reset” state when a program stops (due to a moderate or major failure or a switch to stop mode).

The setting is, however, ignored by some output modules in the event of a major failure. Refer to the data item “Output status when the program stops HOLD/RESET” in the specifications section of each individual output module discussed in "Hardware Manual" (IM 34M06C11-01E).

For a special module, the setting for a stop of programs due to a major failure is always ignored.

By default, all output modules are set to the option “Reset.” You can perform this configuration on 16-relay basis.

CAUTION

- **When using output modules or special modules with Y□□□□□ output relays in a multi-CPU system configuration**
  - Combination of F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 F3SP59, F3SP66, F3SP67, F3SP71 and F3SP76 CPU Modules
    You can output data from multiple sequence CPU modules separately to the output relays of the same output module on 16-relay basis. To do this, configure the sequence CPU modules so that all of them share the same output option (“Hold” or “Reset”). (Also re-configure any sequence CPU module whose output relays are set to the option “Not Used,” so that it has the same settings as the other CPUs.)

  - Other Combinations of CPU Modules
    You may not use the same output module with multiple CPUs.

4.2 Internal Relays (I), Shared Relays (E) and Extended Shared Relays (E)

This section describes internal relays (I), shared relays (E), and extended shared relays (E).

Internal relays (I) are 1-bit variables that can be used without restriction in a program.

Shared relays (E) and Extended Shared relays (E) are 1-bit variables that can be used to perform data communications between CPUs in a multi-CPU system.

4.2.1 Internal Relays (I)

Internal relays are auxiliary relays available for use in programs.

In programs, you can use these relays, for example, for contacts **a** and **b**, coils and application instructions. Unlike I/O relays (X/Y) however, these relays cannot directly exchange signals with external equipment. There is no limit on the number of contacts **a** and **b** that can be used in a program.

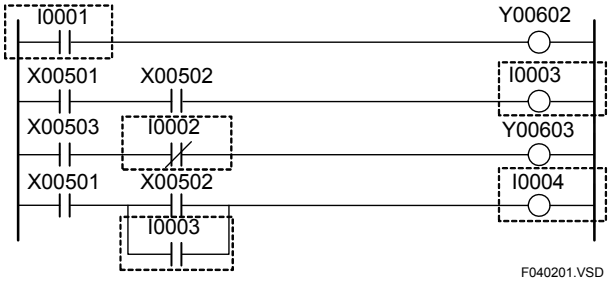


Figure 4.4 Internal Relays (I)

Using the configuration function, you can define the data latch range at power failure for devices whose operation results are to be latched when power is turned off.

A non-latched device will be cleared to “OFF (0)” when you perform any of the following:

- power on the module
- switch the operating mode to Run or Debug using WideField3 (or WideField2)
- execute a Clear Device command from WideField3 (or WideField2)

A latched device retains its operation result even after power off and power on, and is cleared to “OFF (0) ” when you:

- execute a Clear Device command from WideField3 (or WideField2)

4.2.2 Shared Relays (E) and Extended Shared Relays (E)

Shared and extended shared relays are used to perform communications between CPU modules in cases where a sequence CPU module and add-on CPU modules are installed.

Shared relays (E) are available with the F3SP21, F3SP22, F3SP25, F3SP28, F3SP35, F3SP38, F3SP53, F3SP58, F3SP59, F3SP66, F3SP67, F3SP71 and F3SP76 sequence CPU modules, as well as with any add-on sequence CPU modules that are combined with one of these sequence CPUs.

Extended shared relays (E) are only available if one of the F3SP22, F3SP25, F3SP28, F3SP35, F3SP38, F3SP53, F3SP58, F3SP59, F3SP66, F3SP67, F3SP71 and F3SP76 sequence CPU modules is combined with any one or more of these CPU modules installed as add-on CPU modules.

In programs, you can use these relays, for example, for contacts **a** and **b**, coils and application instructions.

You can exchange ON/OFF data between CPUs by using shared relays (E) of the own CPU as coils and those of the other CPUs as contacts.

CAUTION

If you write data to a device area not belonging to the own CPU, data of shared and extended shared relays (E) of the other CPUs are overwritten and so operation results are incorrectly reflected.

By default, no shared relays are allocated as devices. When using add-on CPU modules, configure the range of shared relays to be used. Allocate the same device range for all of the CPU modules. Otherwise, the shared relays (E) will not be correctly refreshed.

The following figure shows an example of how specific shared relays are shared if you allocate shared relays E0001-0512 to CPU1 and E0513-1024 to CPU2.

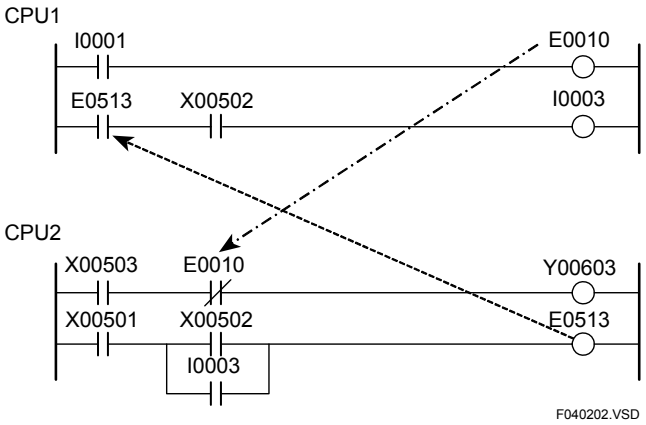


Figure 4.5 Shared Relays (E)

Using the configuration function, you can define the data latch range at power failure for devices whose operation results are to be latched when power is turned off.

By default, all shared relays (E) are non-latched.

A non-latched device will be cleared to “OFF (0)” when you perform any of the following:

- power on the module
- switch the operating mode to Run or Debug using WideField3 (or WideField2)
- execute a Clear Devices command from WideField3 (or WideField2)

A latched device retains its operation result even after power off and power on, and is cleared to “OFF (0) ” when you:

- execute a Clear Devices command from WideField3 (or WideField2)

When using shared or extended shared relays (E), observe the precautions given below.

CAUTION

(1) Index modification of shared or extended shared relays (E)

When applying index modification to a shared or extended shared relay (E) of the own CPU, ensure that the resultant relay number does not exceed the range specified for the own CPU in the configuration. Otherwise, data held by a shared or extended shared relay (E) of other sequence CPU modules are overwritten and operation results are not correctly reflected.

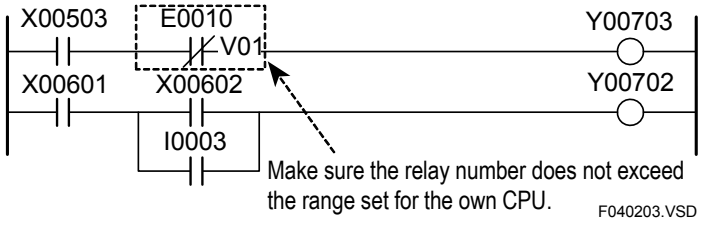


Figure 4.6 Precautions when Using Shared or Extended Shared Relays (E) (1 of 2)

CAUTION

(2) Block move and operation of multiple devices

When using shared or extended shared relays (E) in an instruction for transferring or operating data held by multiple devices, ensure that the specified range of these relays does not exceed the range specified for the own CPU in the configuration. Otherwise, data held by shared or extended shared relays (E) of other sequence CPU modules are overwritten and so operation results are not correctly reflected.

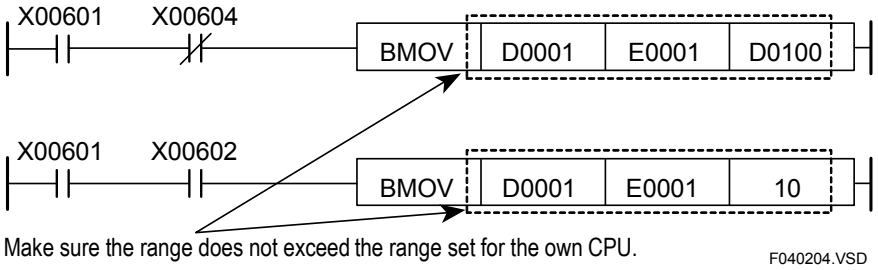


Figure 4.7 Precautions when Using Shared or Extended Shared Relays (E) (2 of 2)

(3) Simultaneity of data

Using the configuration function, you can select either “Simultaneous” or “Non-simultaneous” for simultaneity of data of shared devices.

If you select the “Simultaneous” option, simultaneity of data is guaranteed for units of devices (shared relays (E) /registers or extended shared relays (E) /registers) to be refreshed where one of the F3SP22, F3SP28, F3SP38, F3SP53, F3SP58, F3SP59, F3SP66, F3SP67, F3SP71 and F3SP76 sequence CPU modules is combined with any one or more of these CPUs installed as add-on CPU modules. (Simultaneity of data between shared relays (E) /registers and extended shared relays (E) /registers is not guaranteed, however.)

If any of the F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 and F3SP59 CPUs is combined with any of the F3SP21, F3SP25, and F3SP35 sequence CPU modules, simultaneity of shared refreshed data cannot be guaranteed regardless of the configuration setting.

The “Non-simultaneous” option is provided for compatibility with the F3SP21, F3SP25 and F3SP35 CPUs. Select this option when these CPUs are replaced with the F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 and F3SP59 sequence CPU modules.

■ Configuring Shared and Extended Shared Relays (E) in a Multi-CPU System

Specify the range of shared and extended shared relays (E) to be used by each CPU when add-on CPU modules are installed.

You can allocate any number of relays on 32-relay basis.

Extended shared relays (E) are only available if one of the F3SP22, F3SP25, F3SP28, F3SP35, F3SP38, F3SP53, F3SP58, F3SP59, F3SP66, F3SP67, F3SP71 and F3SP76 sequence CPU modules is combined with one or more CPUs from the same list installed as add-on CPUs.

Table 4.1 Configuration of Shared Relays (E)

Item	F3SP22, F3SP28, F3SP38 F3SP53, F3SP58, F3SP59	
	Default	Configuration Range
Shared relay (E)	0	2048 max. for all CPUs combined in increments of 32 (E0001 to E2048)
Extended shared relay (E)	0	2048 points max. for all CPUs combined in increments of 32 (E2049 to E4096)

CAUTION

- The starting number for extended shared relays (E) is always E2049 even if the range of shared relays (E) used is less than 2048.
- Apply the same allocation of shared/extended shared relays (E) to all CPUs. If the allocation differs among CPUs, shared refreshing will not execute correctly and as a result operation result will not be correctly reflected.

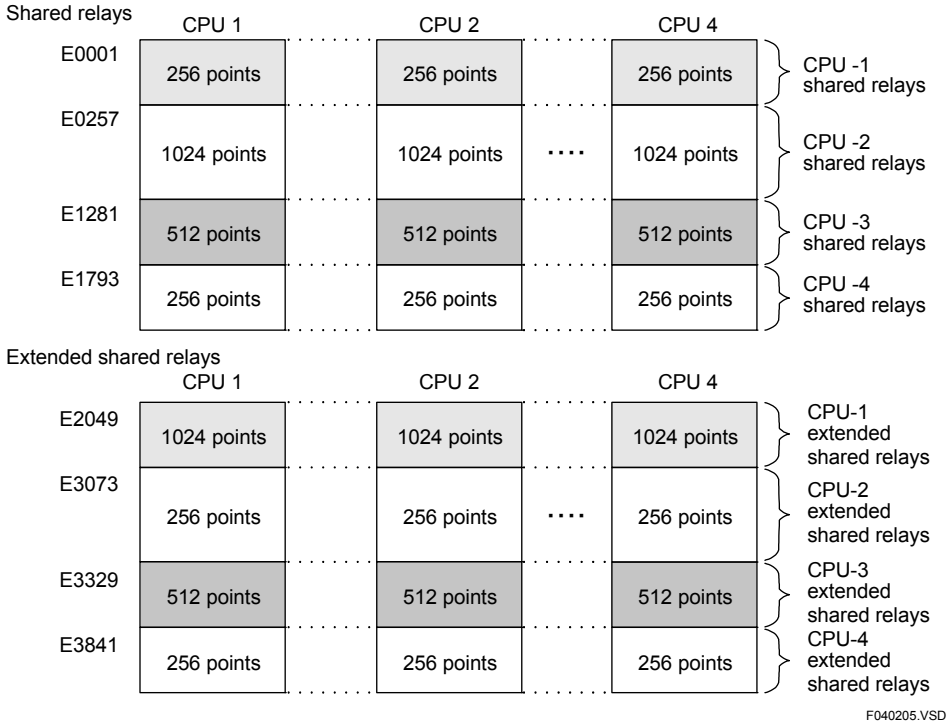


Figure 4.8 Example of Shared and Extended Shared Relay (E) Allocation with Four Sequence CPU Modules Installed

4.3 Link Relays (L) and Link Registers (W)

This section describes link relays (L), link registers (W), their settings, system numbers, as well as the link refreshing range.

Link relays (L) are 1-bit variables used for data communications with FA link systems and FL-net systems. Link registers (W) are 16-bit variables used for data communications with FA link systems and FL-net systems.

CAUTION

In this section, FL-net nodes are called "stations."

Link relays (L) and link registers (W) are devices used to exchange data with other programmable controllers via FA link modules and FL-net (OPCN-2) Interface modules.

Before using link relays, specify the range of links for both the local and remote stations.

Using the configuration function, you can define the data latch range at power failure for devices whose operation results are to be latched when power is turned off.

By default, all link relays (L) and link registers (W) are non-latched.

A non-latched device will be cleared to “OFF (0)” when you perform any of the following:

- power on the module
- switch the operating mode to Run or Debug using WideField3 (or WideField2)
- execute a Clear Devices command from WideField3 (or WideField2)

A latched device retains its operation result even after power off and power on, and is cleared to “OFF (0) ” when you:

- execute a Clear Devices command from WideField3 (or WideField2)

4.3.1 Link Relays (L)

Link relays are used to exchange data with other programmable controllers via FA link modules or FL-net (OPCN-2) Interface modules.

In programs, you can use these relays, for example, for contacts **a** and **b**, coils, and application instructions. In addition, you can exchange ON/OFF data between CPUs by using link relays (L) of the local station as coils and those of remote stations as contacts.

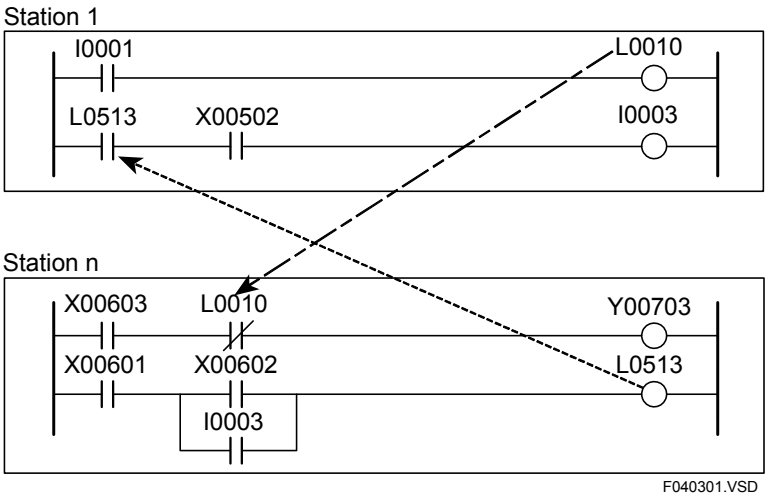


Figure 4.9 Link Relays (L)

The relay number is coded as Lmnnnn, where:

m = System number 1 (0 to 7)

nnnn = Link relay number

Table 4.2 Range of Link Relay Numbers

Module		Configuration Range
FA link H Module	High speed configuration	1 to 1024
Fiber-optic FA Link H Module	Normal configuration	1 to 2048
FL-net (OPCN-2) Interface Module		1 to 8192

4.3.2 Link Registers (W)

Link registers are used to exchange data with other programmable controllers via FA link modules or FL-net (OPCN-2) Interface modules. In programs, you can read from or write to link registers on 16-bit or 32-bit basis using application instructions.

When you use a long word, the low-order 16 bits are stored in the link register with the number specified in the instruction and the high-order 16 bits are stored in the link register (W) with that number incremented by 1.

Data exchange between the local station and remote stations can be achieved by writing data to link registers (W) of the local station and reading it from a remote station.

Before using link registers, configure the range of links for the local station and remote stations.

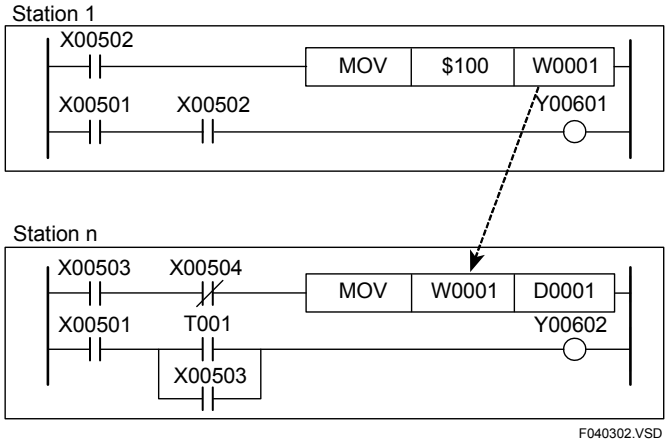


Figure 4.10 Link Registers (W)

The register number is coded as Wmnnnn, where:

m = System number 1 (0 to 7)

nnnn = Link register number

Table 4.3 Range of Link Register Number

Module		Configuration Range
FA Link H module	High speed configuration	1 to 1024
Fiber-optic FA Link H module	Normal configuration	1 to 2048
FL-net (OPCN-2) Interface module		1 to 8192

4.3.3 System Numbers

Modules in FA link and FL-net (OPCN-2) systems are are automatically assigned system numbers based on their slot positions, with the module having the smallest slot number named as system 1.

If FA link modules and FL-net (OPCN-2) modules are intermixed, the modules are assigned system numbers sequentially regardless of the module type.

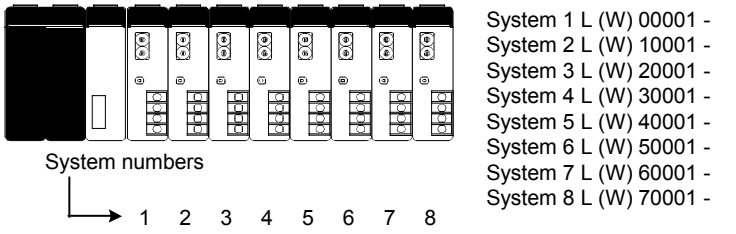


Figure 4.11 Assignment of System Numbers

To manually assign system numbers to modules independent of their slot positions, use configuration to assign fixed system numbers to slot positions.

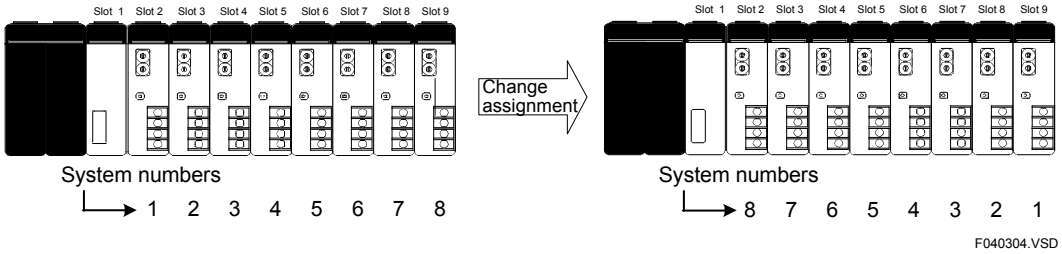


Figure 4.12 Changing System Number Assignment

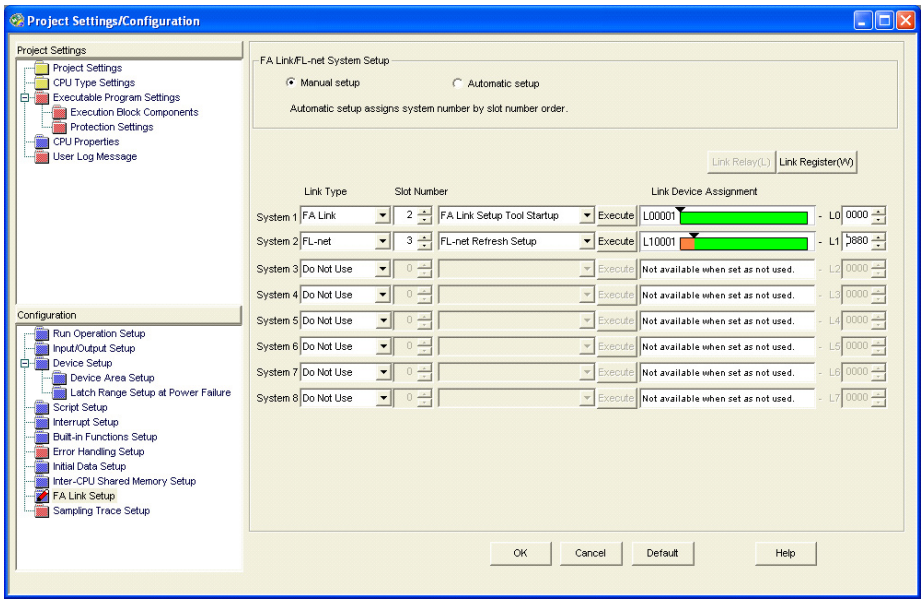


Figure 4.13 WideField3 Configuration Setup

4.3.4 Configuring Link Relays (L) and Registers (W)

Specify the range of link relays and registers to be included in each link system.
For each system, specify the number of link relays and link registers to be used.

Table 4.4 Configuration of Device Capacities

Item		F3SP22 F3SP28 F3SP53		F3SP38 F3SP58 F3SP59	
		Default	Configuration Range	Default	Configuration Range
Device Capacities	Link relays (L) for each system (FA-link or FL-net (OPCN-2) system)	System 1 to 4: 2048 System 5 to 8:0	8192 max. for all systems combined in increments of 16	2048 for each system	16384 max. for all systems combined in increments of 16
	Link registers (W) for each system (FA link or FL-net (OPCN-2) system)	System 1 to 4: 2048 System 5 to 8:0	8192 max. for all systems combined in increments of 16	2048 for each system	16384 max. for all systems combined in increments of 16

4.3.5 Link Refreshing Range

This section describes the link refreshing range for an FA link or FL-net (OPCN-2).

■ Setting Link Refreshing Range for an FL-net System

You can select whether to perform link refreshing on per node basis by configuration.

You can specify not to read the common area (i.e. refresh links) of nodes not involved in data exchange so as to shorten the processing time required for link refreshing.

By default, all nodes are refreshed.

■ Setting Link Refreshing Range for an FA Link System

Link refreshing of an FA link module is performed only for link relays (L) and link registers (W) that are used by instructions coded in a program.

● Link Relay (L)

- If link relays (L) are directly coded in a program, each word containing such a link relay is refreshed.
- If link relays (L) are specified by index modification, each word including the link relay (L) designated by the index register with index value of 0 is refreshed.

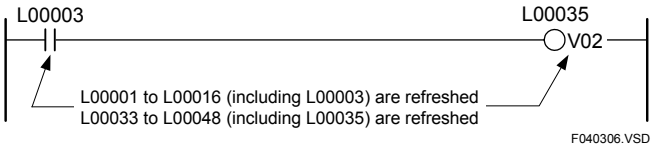


Figure 4.14 Link Relay (L) Link Refreshing Range

● Link Register (W)

- For an instruction that handles word data, the link register (W) specified in the instruction is refreshed.
- For an instruction that handles long-word data or IEEE single-precision floating-point data, the link register (W) specified in the instruction and the data in link register (W) +1 are refreshed.
- For an instruction that handles two or more words of data, the specified range of words is refreshed if the range is specified by a constant, while only the first word is refreshed if the range is specified by a register.

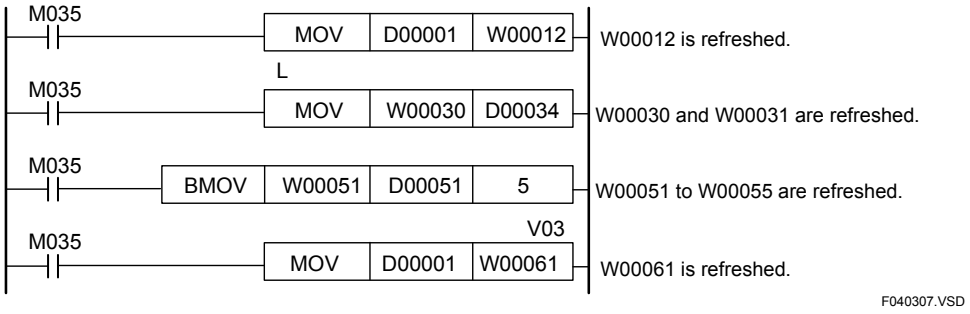


Figure 4.15 Link Refreshing Range for Link Registers (W)

TIP

Link relays (L) and link registers (W) specified in a program are included in link refreshing irrespective of whether the relevant instructions are executed.

If you want to include all link relays (L) and link registers (W) in link refreshing, include the following code in your program:

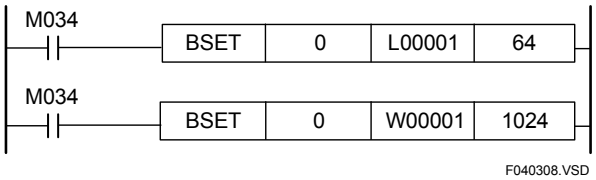


Figure 4.16 To include L00001 to L01024 and W00001 to W01024 in Link Refreshing

If link relays (L) and link registers (W) are specified by index modification, define the index modification range as shown below so that they will be included in link refreshing.

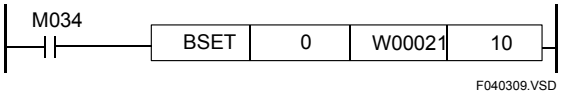


Figure 4.17 When Using Registers W00021 to W00030 with Index Modification

CAUTION

(1) Index modification/indirect specification

- Index modification/indirect specification must not be made across different systems.
- When specifying link relays and registers with index modification or indirect specification, see TIP to link-refresh to ensure that all relevant devices are link-refreshed.

(2) Block move and operation involving multiple devices

- Block move or operation for multiple devices must not be made across different systems. Be careful especially when specifying the number of bytes of data to be moved or the number of devices for operation using devices.
- When specifying the number of bytes of data to be moved or the number of devices for operation using devices, see TIP to ensure that all relevant devices are link-refreshed.

(3) Multi-CPU configuration

- Multiple CPU modules cannot share the same FA link module or FL-net (OPCN-2) interface module. Ensure that only one CPU module is accessing an FA link module or FL-net (OPCN-2) interface module.

4.4 Special Relays (M)

Special relays have specific functions, such as indicating the internal state of the sequence CPU module or detecting errors.

In programs, these relays are used mainly for contacts **a** and **b**.

4.4.1 Block Start Status Relays

Block Start Status relays indicate which blocks are executed when only specified blocks are executed.

These relays are numbered in ascending order as M001, M002, . . ., to correlate with block 1, block 2, ...

Table 4.5 Block Start Status Relays

Item	Block Start Status Relays		
Relay Number	**Description**	**Functionality**	**Explanation**
M0001 to M0032	Block n start status	ON : Run OFF : Stop	Indicates whether block n is executed when the module is configured to execute specified blocks only.
M2001 to M3024			

Note: The Start Status relays assigned to blocks 1 to 32 are M0001 to M0032 and M2001 to M2032 (M0001 to M0032 have the same values as M2001 to M2032. Similarly, Start Status relays M2033 to M3024 map to blocks 33 to 1024.

CAUTION

Do not write to a special relay, including those not listed in tables in this section (e.g., M067 to M128), unless otherwise stated. Special relays are used by the sequence CPU module. Writing to these relays incorrectly may lead to system shutdown or other failures. Using forced set/reset instruction in debug mode is also prohibited.

CAUTION

Special relays with index modification cannot be specified as destinations for data output and if specified, will result in instruction processing errors during execution.

CAUTION

Special relays cannot be specified as output destinations in block transfer and table output ladder instructions, and if specified, will cause instruction processing errors during execution.

- **Block transfer instructions: BMOV, BSET, SMOV, etc.**
- **Table output instructions: ULOGR, FIFWR, etc.**

4.4.2 Utility Relays

Utility relays are used to provide timing in a program or issue instructions to the CPU module.

Table 4.6 Utility Relays

Item	Utility Relays		
No.	Name	Function	Description
M033	Always ON	ON OFF	Used for initialization or as a dummy contact in a program.
M034	Always OFF	ON OFF	
M035	1 scan ON at program start	1 Scan	Turns on for one scan only after a program starts execution.
M036*	0.01 s clock	0.005s 0.005s	Generates a clock pulse with a 0.01s period.
M037*	0.02 s clock	0.01s 0.01s	Generates a clock pulse with a 0.02 s period.
M038*	0.1 s clock	0.05s 0.05s	Generates a clock pulse with a 0.1 s period.
M039*	0.2 s clock	0.1s 0.1s	Generates a clock pulse with a 0.2 s period.
M040*	1 s clock	0.5s 0.5s	Generates a clock pulse with a 1 s period.
M041*	2 s clock	1s 1s	Generates a clock pulse with a 2 s period.
M042*	1 min clock	30s 30s	Generates a clock pulse with a 1 min period.
M047*	1 ms clock	0.5ms 0.5ms	Generates a clock pulse with a 1 ms period.
M048*	2 ms clock	1ms 1ms	Generates a clock pulse with a 2 ms period.
M066	Normal subunit transmission line	ON : Normal transmission line or no fiber-optic FA-bus installed OFF: Unspecified or abnormal transmission line	
M067	On for one scan at sensor CB startup	ON : When the block starts OFF: In all other cases	Turns on for one scan when the sensor control block starts (at the first execution of the sensor control block).

*: Blocks M036 to M048 have their rising and falling clock timing synchronized. Updates are done at the end of a scan.

SEE ALSO

For details on the M066 Utility relay (Normal Subunit Transmission Line), see "Fiber-optic FA-bus Module and Fiber-optic FA-bus Type 2 Module, FA-bus Type 2 Module" (IM 34M06H45-01E).

4.4.3 Sequence Operation and Mode Status Relays

Sequence operation and mode status relays indicate the status of sequence operation and various modes.

Table 4.7 Sequence Operation and Mode Status Relays

Item	Sequence Operation and Mode Status Relays		
No.	**Name**	**Function**	**Description**
M129	Run mode flag	ON : Run mode OFF: Other modes	Indicates the status of CPU operation.
M130	Debug mode flag	ON : Debug mode OFF: Other modes	Indicates the status of CPU operation.
M131	Stop mode flag	ON : Stop mode OFF: Other modes	Indicates the status of CPU operation.
M132	Pause flag	ON : Pause OFF: Run	Indicates the status of program execution during debug mode operation.
M133	Execution flag	ON : Specified blocks OFF: All blocks	Indicates whether all blocks or specified blocks are executed.
M135	RAM/ROM-based operation flag	ON : ROM-based operation OFF: RAM-based operation	Indicates whether operation is based on the ROM or RAM.
M136	Power-on operation flag	ON : Power-on operation OFF: Other modes of operation	Indicates whether operation was initiated by power on or reset
M137	Sensor CB execution status	ON : Run OFF: Stop	Indicates the status of sensor control block operation.
M172 (write-enabled)	Set clock time	ON : Time being set OFF:	Requests to set clock data.
M173	Input-offline flag	ON : Offline OFF: Online	Indicates that input refreshing has stopped.
M174	Output-offline flag	ON : Offline OFF: Online	Indicates that output refreshing has stopped.
M175	Shared-I/O-offline flag	ON : Offline OFF: Online	Indicates that shared refreshing has stopped.
M176	Link-I/O-offline flag	ON : Offline OFF: Online	Indicates that link refreshing has stopped.
M177 to M187	Devices reserved for extended functions		
M188	Carry flag	ON : Carry enabled OFF: Carry disabled	A carry flag used by shift and rotate operations.
M189 to M192	Devices reserved for extended functions		

SEE ALSO

For more details on clock setup, see the specifications of special registers (Z49 to Z54) for clock data.

4.4.4 Self-diagnosis Status Relays

Self-diagnosis status relays indicate the results of self-diagnosis by the sequence CPU.

Table 4.8 Self-diagnosis Status Relays

Item	Self-diagnosis Status Relays		
No.	Name	Function	Description
M193	Self-diagnosis error	ON : Error OFF: No error	Result of self diagnosis is stored in special registers Z17 to Z19
M194	Battery error	ON : Error OFF: Normal	Indicates a failure in backup batteries.
M195	Momentary power failure	ON : Momentary power failure OFF: No momentary power failure	Indicates that a momentary power failure has occurred.
M196	Inter-CPU communication error	ON : Error OFF: Normal	Indicates that a communication failure has occurred in shared relays (E) or shared registers (R).
M197	Existence of CPU1	ON : Exists. OFF: Does not exist.	Indicates whether or not a CPU exists in slot 1.
M198	Existence of CPU2	ON : Exists. OFF: Does not exist.	Indicates whether or not a CPU exists in slot 2.
M199	Existence of CPU3	ON : Exists. OFF: Does not exist.	Indicates whether or not a CPU exists in slot 3.
M200	Existence of CPU4	ON : Exists. OFF: Does not exist.	Indicates whether or not a CPU exists in slot 4.
M201	Instruction processing error	ON : An error is found. OFF: No error is found.	Information of instruction processing error is stored in special registers Z22 to Z24.
M202	I/O comparison error	ON : Error OFF: Normal	Indicates that the state of module installation is not consistent with the program.
M203	I/O module error	ON : Error OFF: Normal	Indicates that no access is possible to I/O modules. The slot number of the error module is stored in special registers Z33 to Z40.
M204	Scan timeout	ON : Error OFF: Normal	Indicates that scan time has exceeded the scan monitoring time.
M210	Subunit communication error	ON : Error OFF: Unspecified or normal line	An error has been detected in the fiber-optic FA-bus module. The slot number of the error module is stored in special registers Z89 to Z96.
M211	Subunit transmitter switching has occurred	ON : Error OFF: Unspecified or normal line	
M212	Sensor CB scan timeout	ON : Error OFF: Normal	Indicates that the execution interval of the sensor control block cannot be maintained.
M225	CPU1 sequence program execution	ON : Run OFF: Stop	Indicates whether sequence program of CPU in slot 1 is running.
M226	CPU2 sequence program execution	ON : Run OFF: Stop	Indicates whether sequence program of CPU in slot 2 is running.
M227	CPU3 sequence program execution	ON : Run OFF: Stop	Indicates whether sequence program of CPU in slot 3 is running.
M228	CPU4 sequence program execution	ON : Run OFF: Stop	Indicates whether sequence program of CPU in slot 4 is running.

SEE ALSO

For details on the M210 (Subunit communication error) and M211 (Subunit transmitter switching has occurred) self-diagnosis relays, see "Fiber-optic FA-bus Module and Fiber-optic FA-bus Type 2 Module, FA-bus Type 2 Module" (IM 34M06H45-01E).

4.4.5 FA Link Module Status Relays

FA Link module status relays indicate the status of FA links.

Table 4.9 FA Link Module Status Relays

Item	FA Link Module Status Relay		
No.	Name	Function	Description
M257 to M480 M8321 to M8992	FA link error	ON : Error OFF: Normal	Indicate the status of FA links.

SEE ALSO

For details on FA link module status relays, see special relays/registers sections of "FA Link H Module, Fiber-optic FA Link H Module" (IM 34M06H43-01E).

4.4.6 FL-net Interface Module Status Relays

FL-net interface module status relays indicate the status of FL-net.

Table 4.10 FL-net Interface Module Status Relays

Item	FL-net Interface Module Status Relay		
No.	Name	Function	Description
M3521 to M3774	Node participation status	1: Participating 0: Not participating	FL-net system 1 *1
M3777 to M4030	Upper layer operation signal error	1: Error 0: Normal	FL-net system 1 *1
M4033 to M4286	Operation status	1: Run 0: Stop	FL-net system 1 *1
M4289 to M4542	Common memory data valid	1: Valid 0: Invalid	FL-net system 1 *1
M4561 to M4814	Node participation status	1: Participating 0: Not participating	FL-net system 2 *2
M4817 to M5070	Upper layer operation signal error	1: Error 0: Normal	FL-net system 2 *2
M5073 to M5326	Operation status	1: Run 0: Stop	FL-net system 2 *2
M5329 to M5582	Common memory data valid	1: Valid 0: Invalid	FL-net system 2 *2

*1: If both FL-net and FA link are installed, FL-net are allocated smaller system numbers among the installed modules.
*2: If both FL-net and FA link are installed, FL-net are allocated larger system numbers among the installed modules.

SEE ALSO

For more details, see "FL-net (OPCN-2) Interface Module" (IM 34M06H32-02E).

4.5 Timers (T)

There are five types of timer (T): 100-µs, 1-ms, 10-ms and 100-ms timers and a 100-ms continuous timer.

For each type of timer (T), you can assign the number of timers using the configuration function. However, you can only assign a maximum of 16 100-µs timers.

CAUTION

Do not use a timer instruction in the sensor control block or an interrupt program. The timer used will not operate correctly.

4.5.1 100-µs, 1-ms, 10-ms, and 100-ms Timers

100-µs, 1-ms, 10-ms, and 100-ms timers are synchronized-scan, decremental timers (T) which update their current values and turn on/off their time-out relays using an end-of-scan process.

Setpoints:

100-µs timer	**0.0001 to 3.2767 s**
1-ms timer	**0.001 to 32.767 s**
10-ms timer	**0.01 to 327.67 s**
100-ms timer	**0.1 to 3276.7 s**

Each timer starts counting at the rising edge of the timer input, and expires when the current value reaches 0. When the timer (T) expires, its time-out relay turns on. The time-out relay is used for a contact **a** or **b**. The timer (T) is reset at the falling edge of the timer input and the current value returns to the timer's setpoint.

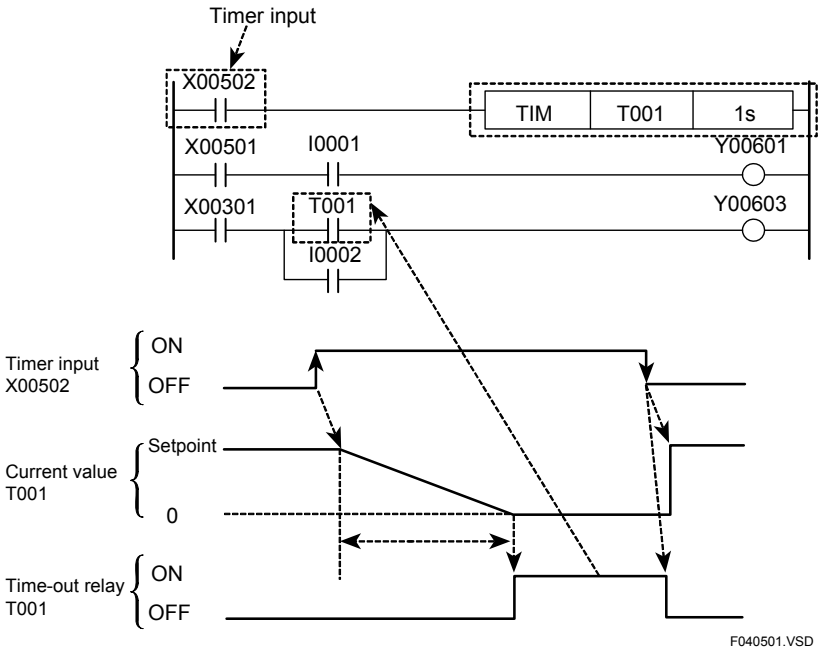


Figure 4.18 Timer (T)

TIP

- The setpoint of a timer refers to the duration from the time the timer starts running (starting time) until the timer expires. The setpoint can be specified using a Timer instruction.
- When a timer is running, its current value decrements as time passes. The current value is set to the setpoint when the timer starts running, and becomes 0 when the timer expires.

4.5.2 100-ms Continuous Timer

A 100-ms continuous timer (T) is a synchronized-scan, decremental timer which updates its current value and turns on/off its time-out relay using an end-of-scan process.

Setpoint: 0.1 to 3276.7 s

The 100-ms continuous timer retains its current value and the state of its time-out relay even when its input condition is OFF. When its input condition turns ON again, the timer resumes counting from its retained value.

When its input condition turns off after the continuous timer expires, the timer (T) is reset, its current value returns to the setpoint, and the time-out relay is set to OFF.

To reset a continuous timer before expiry, write “0” to the timer using a MOV instruction (MOV 0 Tnnn) when the timer input is in an OFF state.

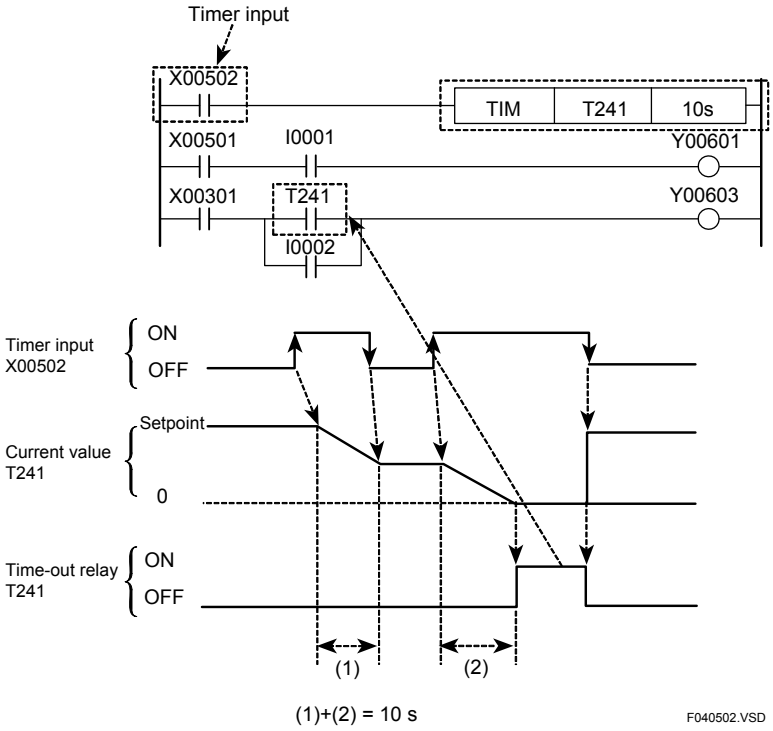


Figure 4.19 100-ms Continuous Timers

Using the configuration function, you can define a range of timer devices whose current values are to be latched when power is turned off. By default, all timers are non-latched.

A non-latched timer resets its current value to its setpoint when you perform any of the following:

- power on the module
- switch the operating mode to Run or Debug using WideField3 (or WideField2)
- execute a Clear Device command from WideField3 (or WideField2)

A latched timer retains its current value even after power off and power on, and resets its current value to its setpoint when you:

- execute a Clear Device command from WideField3 (or WideField2)

4.5.3 Selecting Timers

Configure the device range to be used for each type of 100-µs, 1-ms, 10-ms, and 100-ms timers and 100-ms continuous timers. To do so, specify the number of timers to be allocated for each timer (T) type.

The size of the first device numbers assigned to these timers (T) are related in the following manner:

100-µs timer < 1-ms timer < 10-ms timer < 100-ms timer < 100-ms continuous timer

100-µs, 1-ms, 10-ms and 100-ms timers and 100-ms continuous timers are assigned device numbers of the sequence CPU, in the given order.

Table 4.11 Configuration of Timers

Item		F3SP22 F3SP28 F3SP53		F3SP38 F3SP58 F3SP59	
		Default	**Configuration Range**	**Default**	**Configuration Range**
Configuration of Timer (T) and Counter (C)	100-µs timer	0	2048 for timers and counters combined in increments of 1; 16 max. for 100-µs timers; Timer numbers are continuous.	0	3072 for timers and counters combined in increments of 1; 16 max. for 100-µs timers; Timer numbers are continuous.
	1-ms timer	0		0	
	10-ms timer	512		1024	
	100-ms timer	448		896	
	100-ms continuous timer	64		128	

4.6 Counters (C)

This section describes the function and operation of counters, as well as selection of counters in the configuration.

All counters are decremental counters (C) and have two types of input: count input and counter reset input.

When a counter instruction is executed, a counter decrements its current value each time it detects a rising edge in its count input and terminates when its current value reaches 0.

When the counter (C) terminates, its end-of-count relay turns on. The end-of-count relay is used for a contact a or b.

A counter (C) is reset at the rising edge of its counter reset input and its current value returns to its setpoint. Count input is ignored when the counter reset input is on.

Setpoint: 1 to 32767

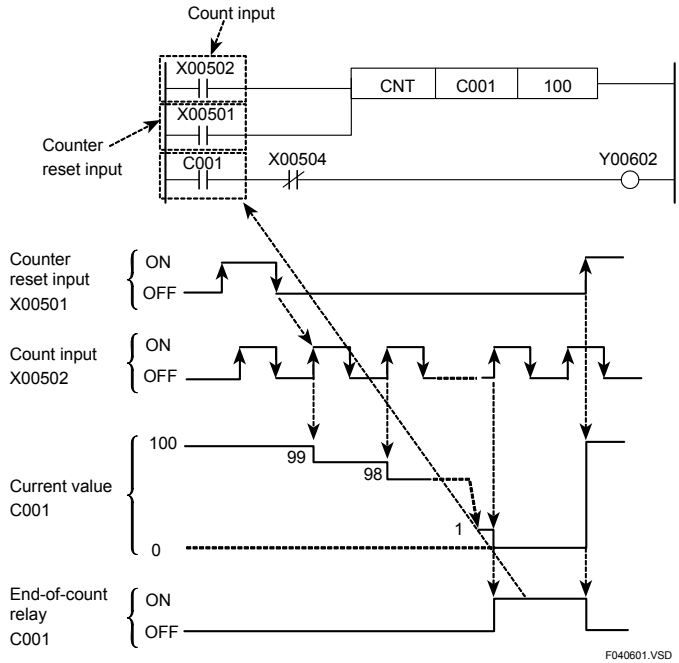


Figure 4.20 Counter (C)

Using the configuration function, you can define a range of counters whose current values are to be latched when power is turned off. By default, all counters are latched.

A non-latched counter resets its current value to its setpoint when you perform any of the following:

- power on the module
- switch the operating mode to Run or Debug using WideField3 (or WideField2)
- execute a Clear Device command from WideField3 (or WideField2)

A latched counter retains its current value even after power off and power on, and resets its current value to its setpoint when you:

- execute a Clear Device command from WideField3 (or WideField2)

TIP

A counter setpoint is used by a counter as its current value when it starts counting. The counter setpoint can be set using a Counter (CNT) instruction.

TIP

When a counter is running, its current value decrements until it reaches 0, at which time the counter is said to have terminated.

4.6.1 Selecting Counters

Select the range of counters (C) to be used.

Table 4.12 Configuration of Counters (C)

Item		F3SP22 F3SP28 F3SP53		F3SP38 F3SP58 F3SP59	
		Default	Configuration Range	Default	Configuration Range
Device Capacities	Timer (T)	T0001 to T1024	2048 max. for counters and timers combined in increments of 1; Timer numbers: T0001 to T2048 Counter numbers: C0001 to C2048	T0001 to T2048	3072 max. for counters and timers combined in increments of 1; Timer numbers: T0001 to T3072 Counter numbers: C0001 to C3072
	Counter (C)	C0001 to C1024		C0001 to C1024	

4.7 Data Register (D), Shared Register (R) and Extended Shared Register (R)

This section describes data registers (D), shared registers (R), extended shared register (R), and how to set the initial data.

Data registers (D) are 16-bit variables that can be used without restrictions in a program. Shared registers (R) and extended shared registers (R) are 16-bit variables that can be used for communications between CPUs in a multi-CPU system.

4.7.1 Data Registers (D)

Data registers serve as memory for storing the results of program-based operation. Each data register has 16 bits (1 word). In programs, you can read from or write to data registers on word or long word basis using application instructions.

When you use data registers on a long word basis, the low-order 16 bits are stored in the data register with the number specified in the instruction and the high-order 16 bits are stored in the data register with that number incremented by 1.

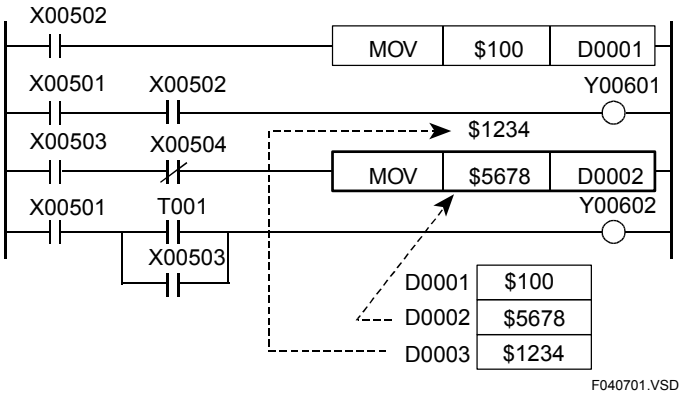


Figure 4.21 Data Registers (D)

Using the configuration function, you can define the data latch range at power failure for devices whose operation results are to be latched when power is turned off.

By default, all data registers (D) are latched.

A non-latched device will be cleared to “OFF (0)” when you perform any of the following:

- power on the module
- switch the operating mode to Run or Debug using WideField3 (or WideField2)
- execute a Clear Device command from WideField3 (or WideField2)

A latched device retains its operation result even after power off and power on, and is cleared to “OFF (0) ” when you:

- execute a Clear Device command from WideField3 (or WideField2)

4.7.2 Shared Registers (R) and Extended Shared Registers (R)

Shared registers and extended shared registers are used to exchange data between CPUs in a multi-CPU system configuration.

Shared registers (R) can be used regardless of how CPUs are combined.

Extended shared registers (R) can only be used with sequence CPU modules (F3SP22, F3SP25, F3SP28, F3SP35, F3SP38, F3SP53, F3SP58 and F3SP59).

In programs, you can read from or write to data registers on word or long word basis using application instructions.

When you use a long word string, the low-order 16 bits are stored in the data register with the number specified in the instruction and the high-order 16 bits are stored in the data register with that number incremented by 1.

Data can be exchanged between the own CPU module and other CPU modules by writing the data to shared registers in the own CPU module and reading it from other CPU modules.

If you write data to a device area not belonging to the own CPU module, data held by shared registers (R) of other CPU modules are overwritten and so operation results are not correctly reflected.

By default, no shared registers are allocated as devices. When using add-on CPU modules, configure the range of shared registers to be used. Allocate the same device range for all of the CPU modules. Otherwise, the shared registers (R) will not be correctly refreshed.

SEE ALSO

Shared and extended shared registers (R) are used to exchange data (data sharing) between CPUs in a multi-CPU system configuration between sequence CPU modules and BASIC CPU modules. For details on the functions of BASIC CPU modules, see "BASIC CPU Modules and YM-BASIC/FA Programming Language" (IM 34M06Q22-01E).

The following figure shows an example of how shared or extended shared registers (R) are shared if you allocate shared registers R0001 to R0256 for CPU 1 and shared registers R0257 to R0512 for CPU 2.

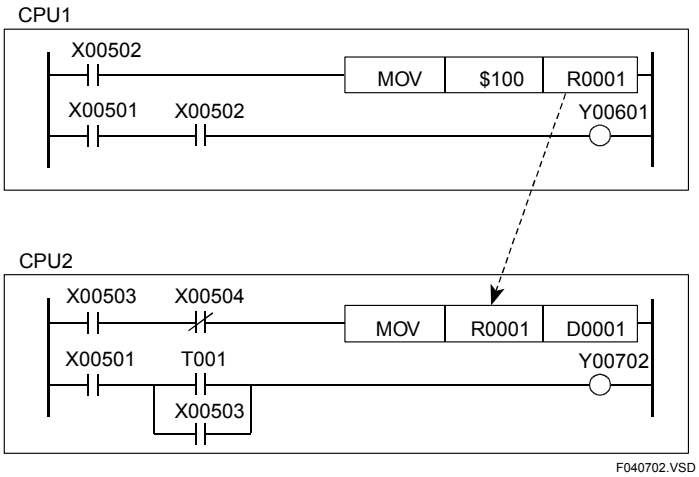


Figure 4.22 Shared Register (R)

Using the configuration function, you can define the data latch range at power failure for devices whose operation results are to be latched when power is turned off.

By default, all shared registers (R) are non-latched.

A non-latched device will be cleared to “OFF (0)” when you perform any of the following:

- power on the module
- switch the operating mode to Run or Debug using WideField3 (or WideField2)
- execute a Clear Devices command from WideField3 (or WideField2)

A latched device retains its operation result even after power off and power on, and is cleared to “OFF (0) ” when you:

- execute a Clear Devices command from WideField3 (or WideField2)

When using shared or extended shared registers (R), observe the precautions given below.

CAUTION

(1) Index modification of shared or extended shared registers (R)

- When applying index modification to a shared or extended shared register (R) of the own sequence CPU module, be careful that the register number, which is directly specified in an instruction, after adding the value of the index register, must not exceed the range specified by configuration for the own CPU. Otherwise, data held by shared or extended shared registers (R) of other CPU modules are overwritten and so operation results are not correctly reflected.

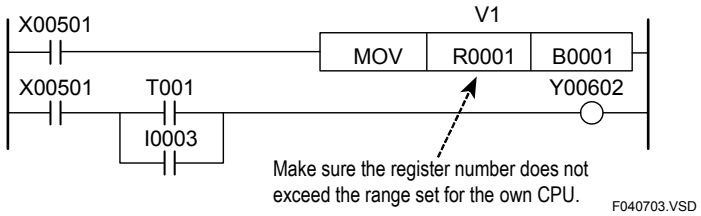


Figure 4.23 Precautions when Using Shared or Extended Shared Registers (R) (1 of 2)

(2) Block move and operation of multiple devices

- When using shared or extended shared registers (R) in an instruction for transferring or operating data held by multiple devices, be careful that the range of registers, which is defined by the register number specified directly in the instruction and the number of registers included in the transfer and operation, must not exceed the range specified by configuration for the own CPU. Otherwise, data held by shared or extended shared registers (R) of other CPU modules are overwritten and so operation results are not correctly reflected.

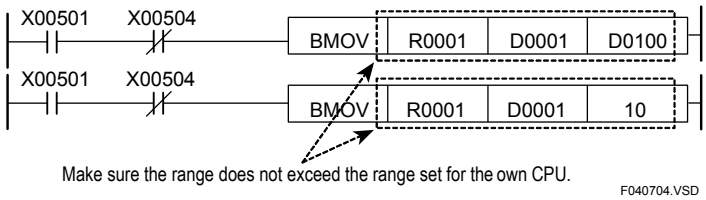


Figure 4.24 Precautions when Using Shared or Extended Shared Registers (R) (2 of 2)

CAUTION

(3) Simultaneity of data

Using the configuration function, you can select either “Simultaneous” or “Non-simultaneous” for simultaneity of data of shared devices.

If you select the “Simultaneous” option, simultaneity of data is guaranteed for units of devices (shared relays (E) /registers or extended shared relays (E) /registers) to be refreshed where one of the F3SP22, F3SP28, F3SP38, F3SP53, F3SP58, F3SP59 F3SP66, F3SP67, F3SP71 and F3SP76 sequence CPU modules is combined with any one or more of these CPUs installed as add-on CPU modules. (Simultaneity of data between shared relays (E) /registers and extended shared relays (E) /registers is not guaranteed, however.)

If any of the F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 and F3SP59 CPUs is combined with any of the F3SP21, F3SP25, and F3SP35 sequence CPU modules, simultaneity of shared refreshed data cannot be guaranteed regardless of the configuration setting.

The “Non-simultaneous” option is provided for compatibility with the F3SP21, F3SP25 and F3SP35 CPUs. Select this option when these CPUs are replaced with the F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 and F3SP59 sequence CPU modules.

SEE ALSO

For details on index modification, see Section 1.10 in “Sequence CPU Instruction Manual – Instructions” (IM 34M06P12-03E).

■ Configuring Shared and Extended Shared Registers (R) for Multiple CPUs

Configure the range of shared and extended shared registers (R) to be used by each CPU in a multi-CPU system configuration where add-on CPU modules are installed. You can allocate any number of registers to each CPU in increments of 2.

Table 4.13 Configuration of Shared Registers (R)

Item		F3SP22, F3SP28, F3SP53 F3SP38, F3SP58, F3SP59	
		Default	Configuration Range
Device Capacities	Shared register (R)	0	1024 max. for all CPUs combined in increments of 2
	Extended shared register (R)	0	3072 max. for all CPUs combined in increments of 2

Extended shared registers (R) can only be used with sequence CPU modules (F3SP22, F3SP25, F3SP28, F3SP35, F3SP38, F3SP53, F3SP58 and F3SP59).

CAUTION

Assign the same range of shared and extended shared registers (R) for all CPU modules. No error will result, however, even if the range is not the same among the CPU modules. Rather, data in other CPU modules may appear wrongly assigned to register numbers or data in the CPU module may appear that way when referenced from other CPU modules.

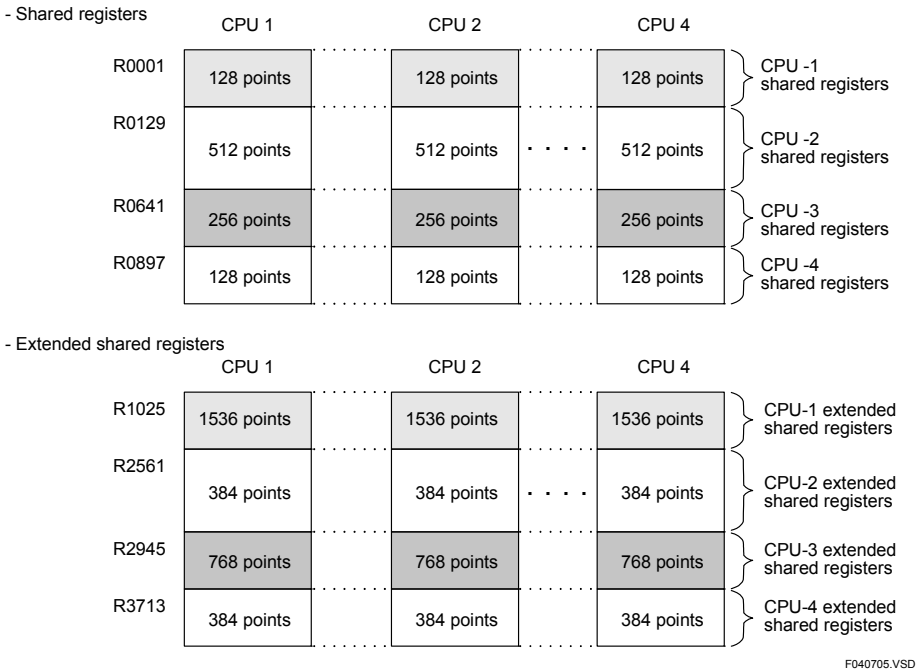


Figure 4.25 Example of Shared and Extended Shared Register (R) Allocation when Four Sequence CPU Modules Are Installed

CAUTION

Even if the specified range includes less than 1024 shared registers (R), the extended shared registers (R) always begin with the number R1025.

4.7.3 Setting Initial Data for Data Registers (D)

Using the configuration function, define the initial values of data registers (D) to be used at the beginning of program execution.

Specify the starting number and the number of data registers to be configured, followed by the initial data values. After this configuration, the preset initial data values are stored in the specified data registers when the program starts. This configuration is useful when a large volume of initial data needs to be set by a program or when the initial data needs to be saved. You can set initial data in a maximum of 1024 data registers.

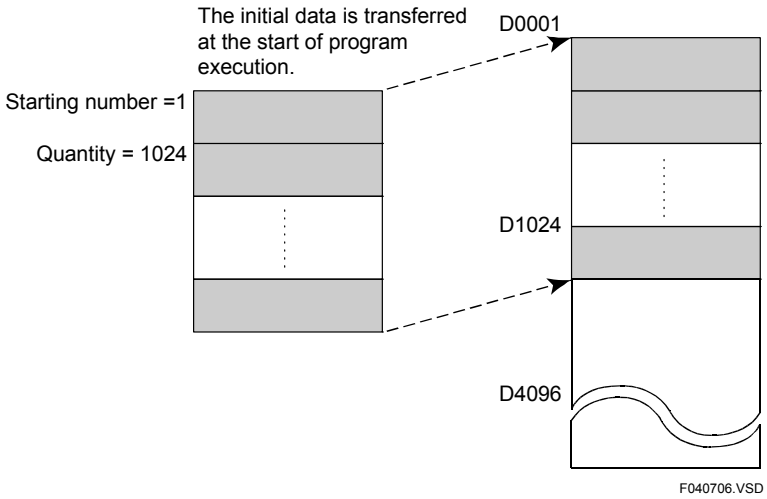


Figure 4.26 Setting Initial Data for Data Registers (D)

4.8 Special Registers (Z)

Special registers have specific functions, such as indicating the internal state of a programmable controller or detecting errors.

4.8.1 Sequence Operation Status Registers

Sequence operation status registers indicate the status of sequence operation.

Table 4.14 Sequence Operation Status Registers

Type	Sequence Operation Status Registers		
No.	Name	Stored Data	Description
Z001	Scan time (Run mode)	Latest scan time	Stores the latest scan time in 100-µs increments.
Z002	Minimum scan time (Run mode)	Minimum scan time	Allows the latest scan time to be read in 100-µs increments if it is shorter than the minimum scan time.
Z003	Maximum scan time (Run mode)	Maximum scan time	Allows the latest scan time to be read in 100-µs increments if it is longer than the maximum scan time.
Z004	Scan time (Debug mode)	Latest scan time	Stores the latest scan time in 100-µs increments.
Z005	Minimum scan time (Debug mode)	Minimum scan time	Allows the latest scan time to be read in 100-µs increments if it is shorter than the minimum scan time.
Z006	Maximum scan time (Debug mode)	Maximum scan time	Allows the latest scan time to be read in 100-µs increments if it is longer than the maximum scan time.
Z007	Peripheral-process scan time	Latest scan time	Stores the latest scan time in100-µs increments. (Tolerance: Scan time of one control process)
Z008	Minimum peripheral-process scan time	Minimum scan time	Allows the latest scan time to be read in 100-µs increments if it is shorter than the minimum scan time. (Tolerance: Scan time of one control process)
Z009	Maximum peripheral-process scan time	Maximum scan time	Allows the latest scan time to be read in 100-µs increments if it is longer than the maximum scan time. (Tolerance: Scan time of one control process)

CAUTION

- Do not write to a special register (Z), including those not listed in the table above (e.g., Z010 to Z016), unless otherwise stated. This is because they are used by the CPU module for the system. If you inadvertently write to these registers, a failure, such as a system shutdown, may result.
- You are not allowed to apply index modification to special registers (Z) in an attempt to specify them as the destination of data output. If you do so, an instruction processing error will result.
- In a ladder instruction for continuous data transfer or table-format data output (see examples below), you are not allowed to specify a special register (Z) as the output destination. If you do so, an instruction processing error will result.
  - **Instructions for continuous data transfer: Block Move Instruction (BMOV Instruction), Block Set Instruction (BSET Instruction), String Move Instruction (SMOV Instruction), etc.**
  - **Instructions for table-format data output: User Log Read Instruction (ULOGR Instruction), FIFO Write Instruction (FIFWR Instruction), etc.**

4.8.2 Self-diagnosis Status Registers

Self-diagnosis status registers indicate the results of self-diagnostics by the sequence CPU.

Table 4.15 Self-diagnosis Status Registers

Type	Self-diagnosis Status Registers		
No.	**Name**	**Stored Date**	**Descriptions**
Z017	Self-diagnosis error	Self-diagnosis error No.	Store the results of self-diagnosis.*
Z018		Self-diagnosis error block No.	
Z019		Self-diagnosis error instruction No.	
Z022	Instruction processing error	Instruction processing error No.	Store errors occurring during instruction processing.*
Z023		Instruction processing error block No.	
Z024		Instruction processing error instruction No.	
Z027	I/O comparison error	I/O comparison error No.	Store detailed information on I/O comparison errors.*
Z028		I/O comparison error block No.	
Z029		I/O comparison error instruction No.	
Z033 to Z040	I/O error	Slot no. with I/O error 16 ····· 2 1 0 ····· 1 0	Store, as a bit pattern, the slot number for which an I/O error has occurred. Z033: Main unit Z034: Subunit 1 Z035: Subunit 2 Z036: Subunit 3 Z037: Subunit 4 Z038: Subunit 5 Z039: Subunit 6 Z040: Subunit 7
Z041	Module recognition	Main unit	Slot number 16 ····· 1 0 ····· 1 0 0: No modules are recognized. Unable to read/write. 1: Modules are recognized.
Z042		Subunit 1	
Z043		Subunit 2	
Z044		Subunit 3	
Z045		Subunit 4	
Z046		Subunit 5	
Z047		Subunit 6	
Z048		Subunit 7	
Z089	Subunit Commu-nication error slot	Main unit	Slot number 16 ····· 1 0 ····· 1 0 Fiber-optic FA-bus module 0: Normal transmission line; Unspecified transmission line; or Loaded with a wrong module 1: Abnormal transmission line (“Subunit communication error” or “Sub unit transmitter switching has occurred)
Z090		Subunit 1	
Z091		Subunit 2	
Z092		Subunit 3	
Z093		Subunit 4	
Z094		Subunit 5	
Z095		Subunit 6	
Z096		Subunit 7	

* For information on error numbers (codes) to be saved in these special registers, see Table 8.2, “Details of Self-diagnosis.”

SEE ALSO

For details on the Z089 to Z096 special registers (Communication error slot), see "Fiber-optic FA-bus Module and Fiber-optic FA-bus Type 2 Module, FA-bus Type 2 Module" (IM 34M06H45-01E).

4.8.3 Utility Registers

Table 4.16 Utility Registers

Type	Utility Registers		
No.	**Name**	**Stored Data**	**Description**
Z049 (write-enabled)	Clock data	Last two digits of calendar year	Stores "year" as a BCD-coded value. e.g. 1999 as $0099 2000 as $0000
Z050 (write-enabled)		Month	Stores "month" as a BCD-coded value. e.g. January as $0001
Z051 (write-enabled)		Day	Stores "day of month" as a BCD-coded value. e.g. 28th as $0028
Z052 (write-enabled)		Hour	Stores "hour" as a BCD-coded value. e.g. 18:00 hours as $0018
Z053 (write-enabled)		Minute	Stores "minute" as a BCD-coded value. e.g. 15 minutes as $0015
Z054 (write-enabled)		Second	Stores "second" as a BCD-coded value. e.g. 30 seconds as $0030
Z055		Day of week ($0 to $6)	Stores "day of week" as a BCD-coded value. e.g. Wednesday as $0003
Z056	Constant scan time	Value of constant scan time	0.1 ms increments e.g. 10 ms as 100
Z057	Constant scan time	Value of constant scan time	1 ms increments e.g. 10 ms as 10
Z058	Scan monitoring time	Value of scan monitoring time	1 ms increments e.g. 200 ms as 200

- For CPU module F3SP□□-□S, you can set clock data using the Set Date instruction (DATE), Set Time instruction (TIME), Set Date String instruction (SDATE), and Set Time String instruction (STIME).

- For CPU module F3SP□□-□N/-□H, use the following procedure to set time data.

  (1) Write the clock data to special registers Z049 to Z054. (Use a MOV P instruction. If you use a BMOV or BSET instruction, an error will be generated).

  (2) Set special relay M172 to ON within the same scan as that in step (1) (use a DIFU instruction).

  (3) Set special relay M172 to OFF in the scan subsequent to that in step (2). Also stop writing the clock data to special registers Z049 to Z054 in that scan.

  Note that no change is made to the clock data and the data reverts to its original values if the values being set are incorrect.

- The accuracy of clock data is as follows.

  Maximum daily error: ±8 s (±2 s, when actually measured)

  The clock accuracy is reset to the maximum daily error of -1.2 s/+2 s, however, when the power is turned off and on again. In addition, it is possible to input a corrective value from the programming tool. If you input a precise corrective value, the clock data is corrected during the power-off-and-on sequence, thus offsetting the cumulative amount of error.

4.8.4 FA Link Module Status Registers

FA Link module status registers indicate the status of FA link.

Table 4.17 FA Link Module Status Registers

Type	FA Link Module Status Registers		
No.	Name	Stored Data	Description
Z075	Local station number		System 1 (FA Link)
Z076	Local station number		System 2 (FA Link)
Z077	Local station number		System 3 (FA Link)
Z078	Local station number		System 4 (FA Link)
Z079	Local station number		System 5 (FA Link)
Z080	Local station number		System 6 (FA Link)
Z081	Local station number		System 7 (FA Link)
Z082	Local station number		System 8 (FA Link)
Z065	Local station status	0: Initialization in progress 1: Offline 2: Online	System 1 (FA Link)
Z066	Cyclic transmission time		System 1 (FA Link) 1ms increments
Z070	Local station status	0: Initialization in progress 1: Offline 2: Online	System 2 (FA Link)
Z071	Cyclic transmission time		System 2 (FA Link) 1ms increments
Z257	Local station status	0: Initialization in progress 1: Offline 2: Online	System 3 (FA Link)
Z258	Cyclic transmission time		System 3 (FA Link) 1ms increments
Z262	Local station status	0: Initialization in progress 1: Offline 2: Online	System 4 (FA Link)
Z263	Cyclic transmission time		System 4 (FA Link) 1ms increments
Z267	Local station status	0: Initialization in progress 1: Offline 2: Online	System 5 (FA Link)
Z268	Cyclic transmission time		System 5 (FA Link) 1ms increments
Z272	Local station status	0: Initialization in progress 1: Offline 2: Online	System 6 (FA Link)
Z273	Cyclic transmission time		System 6 (FA Link) 1ms increments
Z277	Local station status	0: Initialization in progress 1: Offline 2: Online	System 7 (FA Link)
Z278	Cyclic transmission time		System 7 (FA Link) 1ms increments
Z282	Local station status	0: Initialization in progress 1: Offline 2: Online	System 8 (FA Link)
Z283	Cyclic transmission time		System 8 (FA Link) 1ms increments

SEE ALSO

For details on the FA link module status registers (Z), see the Special relays (M) /registers (Z) sections in “FA Link H Module, Fiber-optic FA Link H Module” (IM 34M06H43-01E).

4.8.5 Sequence CPU Module Status Registers

CPU module status registers indicate the status of a CPU.

Table 4.18 Sequence CPU Module Status Registers

Type	CPU Module Status Registers		
No.	**Name**	**Contents**	**Description**
Z105	Number of user log records		See Section 6.14, "User Log Management Functions," for information on user logs.
Z109	Sensor CB execution time	Refers to the length of time from when input refreshing is started for the sensor control block to when the program is executed and output refreshing is completed. (Unit: 10 µs)	
Z111	Maximum Sensor CB execution time	Refers to the maximum time taken to execute the sensor control block. (Unit: 10 µs)	
Z121 to Z128*	Module information	Module name and firmware revision number.	

Note: For example, the values for module”F3SP58-6S”, firmware Rev1 are as follows.
Z121 ”F3”
Z122 “SP”
Z123 “58”
Z124 “6S”
Z125 “/R”
Z126 “01”
Z127 “/ “
Z128 “ “

4.9 Index Registers (V)

Index registers are used to modify devices numbers.

You can use these registers in both basic instructions and application instructions to make index modifications.

Use these registers to address a device by adding the content of an index register to a device number, which is directly specified in an instruction.

SEE ALSO

For details on index registers, see Section 1.10.1, “Index Modification” in “Sequence CPU Instruction Manual – Instructions” (IM 34M06P12-03E).

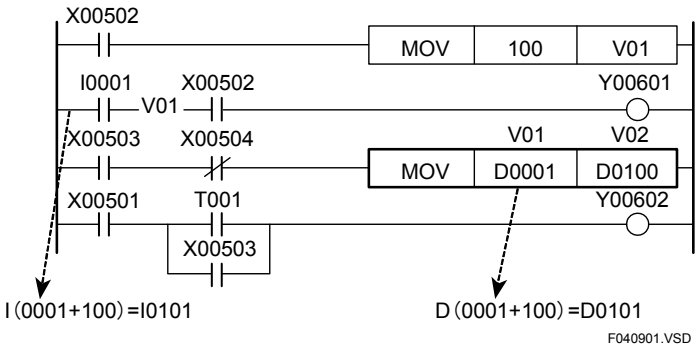


Figure 4.27 Index Registers

CAUTION

The sequence CPU module performs no check on whether an index modified device exceeds the device configuration range. If an index register is incorrectly specified, the device configuration range may be exceeded, resulting in inadvertent selection of a different type of device.

CAUTION

An index register can be set to any value between -32768 and 32767. Therefore, for devices such as file registers (B) whose size is larger than 32768, index modification cannot cover the entire device.

4.10 File Registers (B)

File registers (B) are used as extensions of data registers (D).

Each file register consists of one word.

Like data registers (D), you can read from or write to file registers on a word basis or 32-bit basis using application instructions.

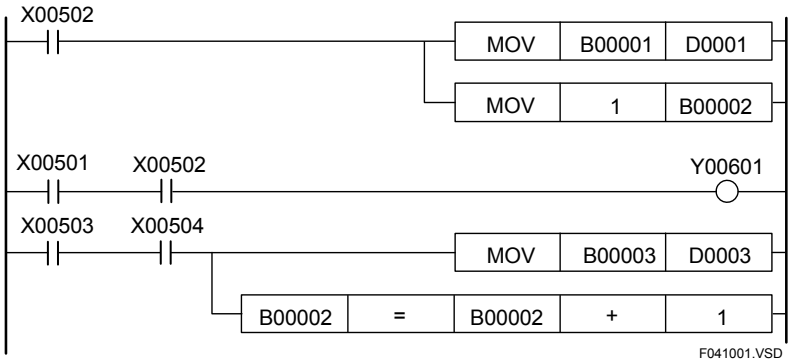


Figure 4.28 File Registers (B)

Unlike data registers (D), all file registers (B) retain their operation results when the power is turned off. A file register is cleared to "OFF (0)" if you:

- write the data value "OFF (0)" to the file register (B) using the programming tool WideField3 (or WideField2)

Unlike data registers (D), file registers are not cleared to "OFF (0)" even if you:

- execute a Clear Device command from WideField3 (or WideField2)
- clear the memory from WideField3 (or WideField2)

Blank Page

5. Programs

This chapter describes languages used for programming, program types and program memory.

5.1 Programming Language

Two types of programming language are available: structured ladder language and mnemonic language. In either case, the written program is read sequentially by the sequence CPU to perform operations according to the program's process details.

5.1.1 Structured Ladder Language

The structured ladder language is based on relay symbol representation and allows a programmer to do structured programming by breaking a program into functional parts.

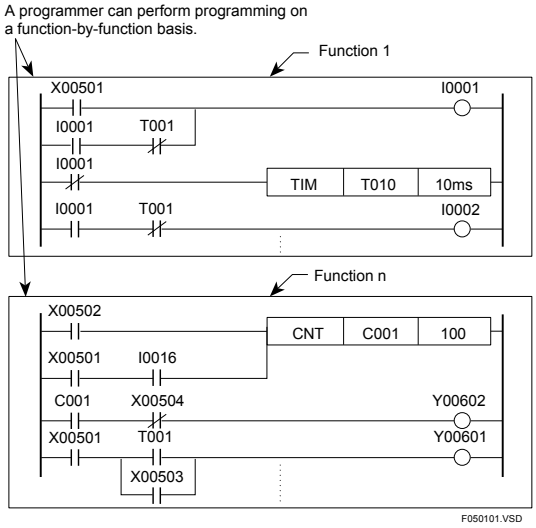


Figure 5.1 Structured Ladder Language

5.1.2 Mnemonic Language

The mnemonic language is designed for describing a program by breaking its process details into instruction, input parameter, and output parameter. Like the structured-ladder language, the mnemonic language allows the programmer to perform programming on a function-by-function basis.

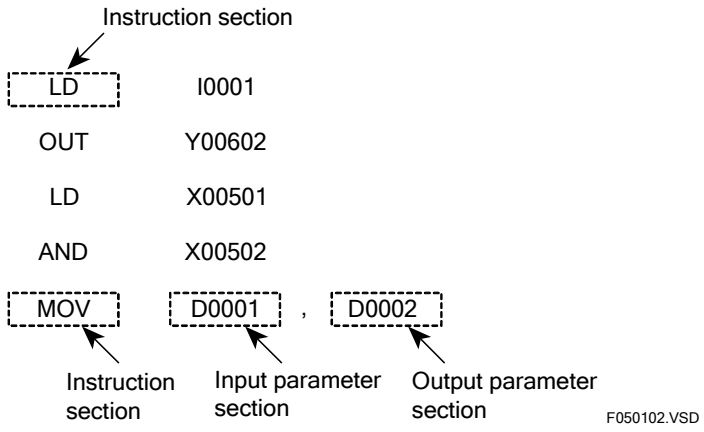


Figure 5.2 Mnemonic Language

5.2 Program Types and Configuration

There are two types of programs: blocks and executable programs.

5.2.1 Blocks and Executable Programs

■ Blocks

A block refers to a collection of circuits entered using WideField3 (or WideField2).

Parts of a program written on a function-by-function basis using the structured ladder language or mnemonic language are managed as blocks. As a program can be maintained or reused on block basis, program development becomes easier.

CPU modules F3SP□□-□N and F3SP□□-□H allow up to 10K steps per block.

CPU module F3SP□□-□S allows up to 56K steps per block (10K steps per block for F3SP22-0S and 30K steps per block for F3SP28-3S).

CAUTION

An individual block cannot be executed by the CPU.

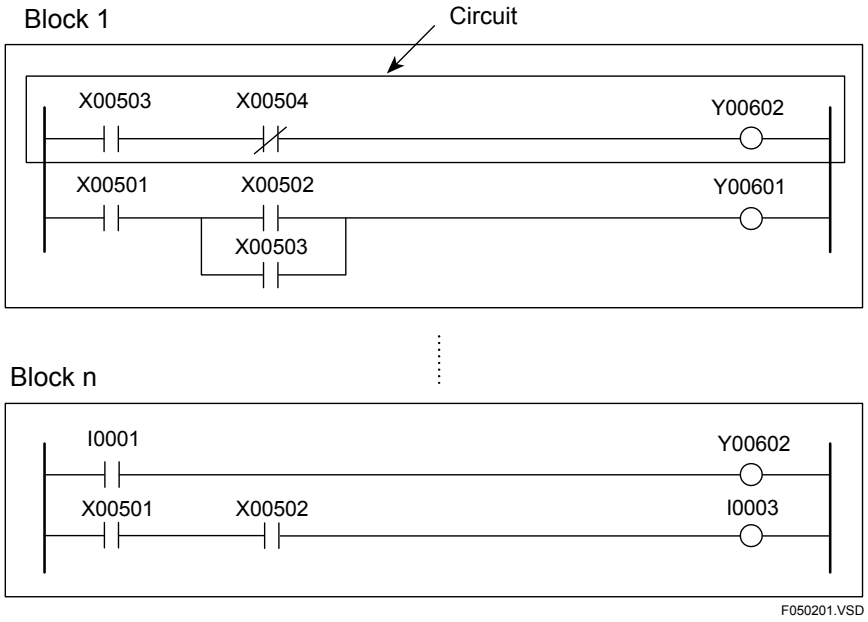


Figure 5.3 Blocks

■ Executable Program

An executable program refers to a program, which is stored in a format that allows it to be executed by the CPU. An executable program is composed by combining multiple blocks created using WideField3 (or WideField2). Each executable program can contain a maximum of 1024 blocks.

You can either execute all or selected blocks of an executable program. This simplifies program management.

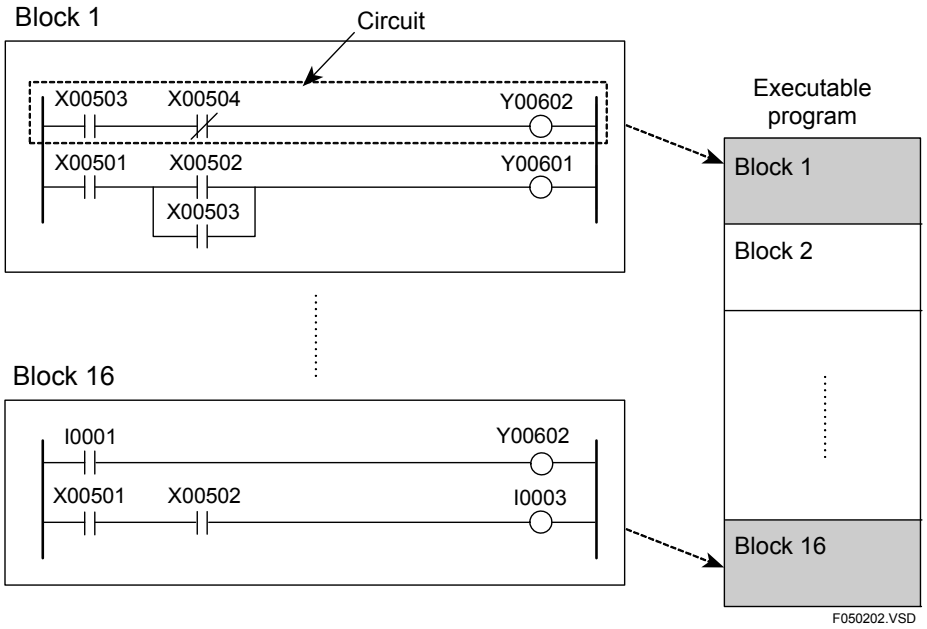


Figure 5.4 Example of an Executable Program

5.2.2 Component Programs of an Executable Program

An executable program contains a maximum of 1024 blocks. The sensor control block is regarded as a single, separate block. Programs that compose an executable program are classified into main routine programs, subroutine programs, interrupt programs and sensor control block programs, according to their functions.

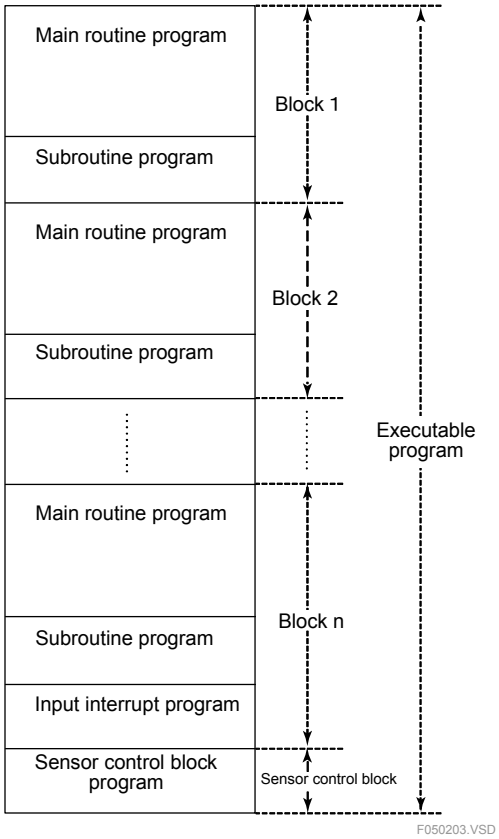


Figure 5.5 Component Programs of an Executable Program

■ Main Routine Program

A main routine program is always executed in each scan.

The main routine program is written using structured ladder language, and is composed of multiple blocks.

You can execute a main routine program by either executing all blocks of the program or executing only specified blocks.

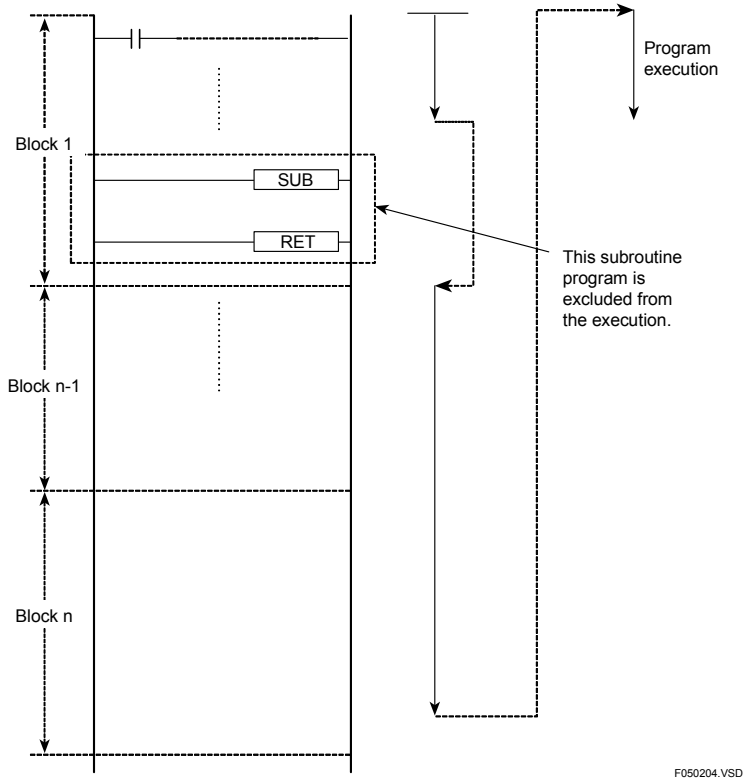


Figure 5.6 How a Main Routine Program Is Executed

■ Subroutine Program

A subroutine program is executed when a main routine program executes a CALL instruction. Use a subroutine program when you want to run a specific process two or more times within one scan. A subroutine program can be placed in any location in a block.

In the case where specified blocks are selected for execution, a subroutine program, which is called from a block being executed, will be executed even if it is located in a block, which is not selected for execution.

Subroutine program calls can be nested up to eight levels deep. (To nest a call is to call a subroutine from within another subroutine).

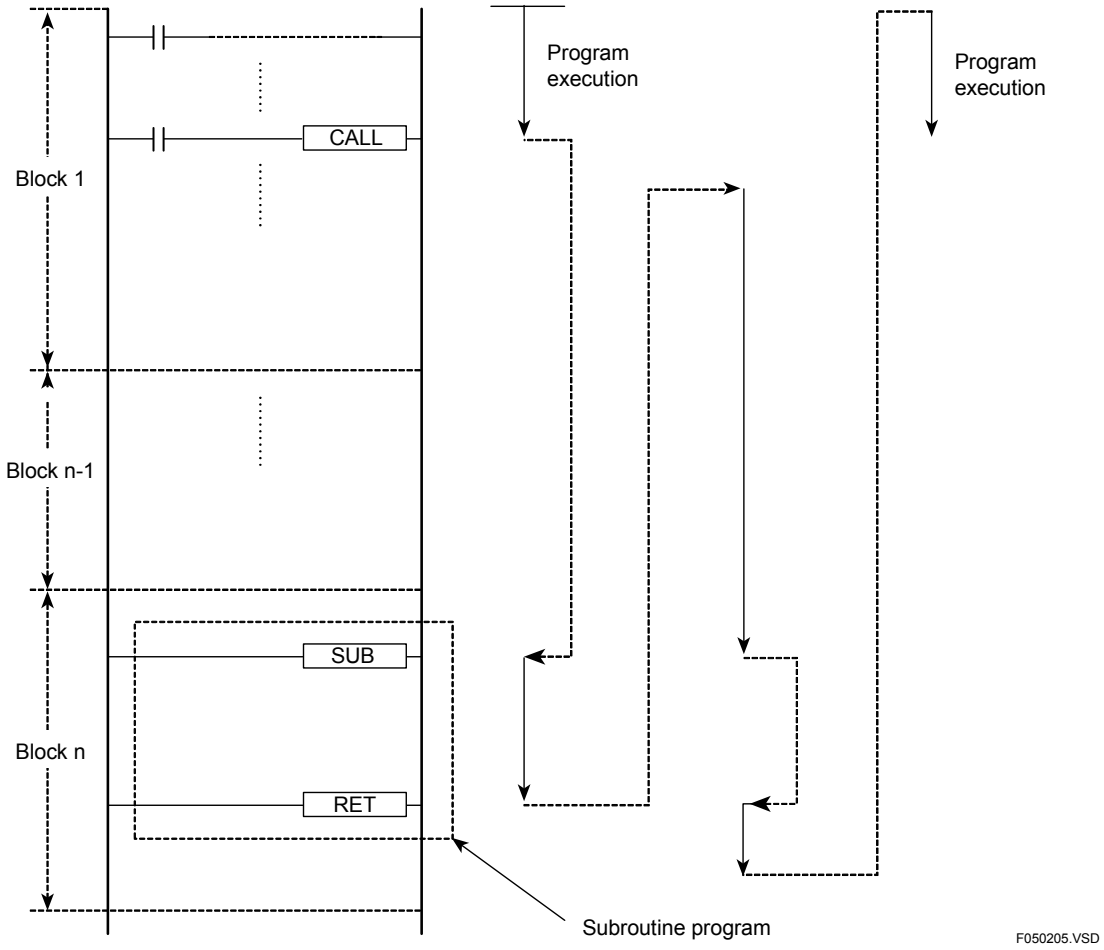


Figure 5.7 How a Subroutine Program Is Executed

■ Interrupt Programs

An interrupt program is executed when any cause of interrupt occurs.

A maximum of four interrupt programs can be included in a program.

The relationship between a cause of interrupt and an interrupt program is described as a parameter of the Interrupt (INTP) Instruction.

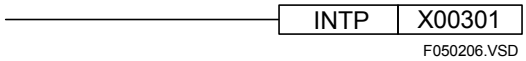


Figure 5.8 INTP Instruction

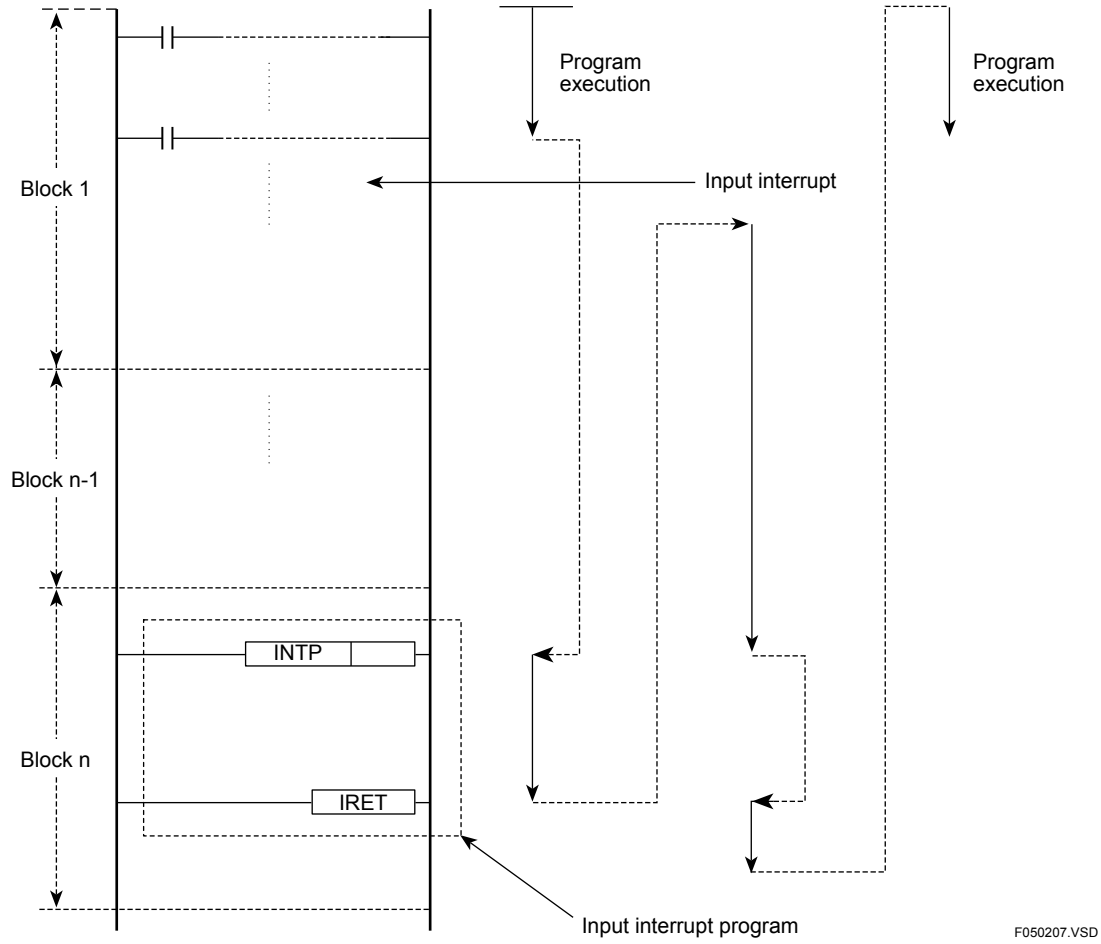


Figure 5.9 How an Interrupt Program Is Executed

■ Sensor Control Block

The sensor control block (SCB) is one block, which is executed at high-speed and at fixed intervals, separately from the normal scan.

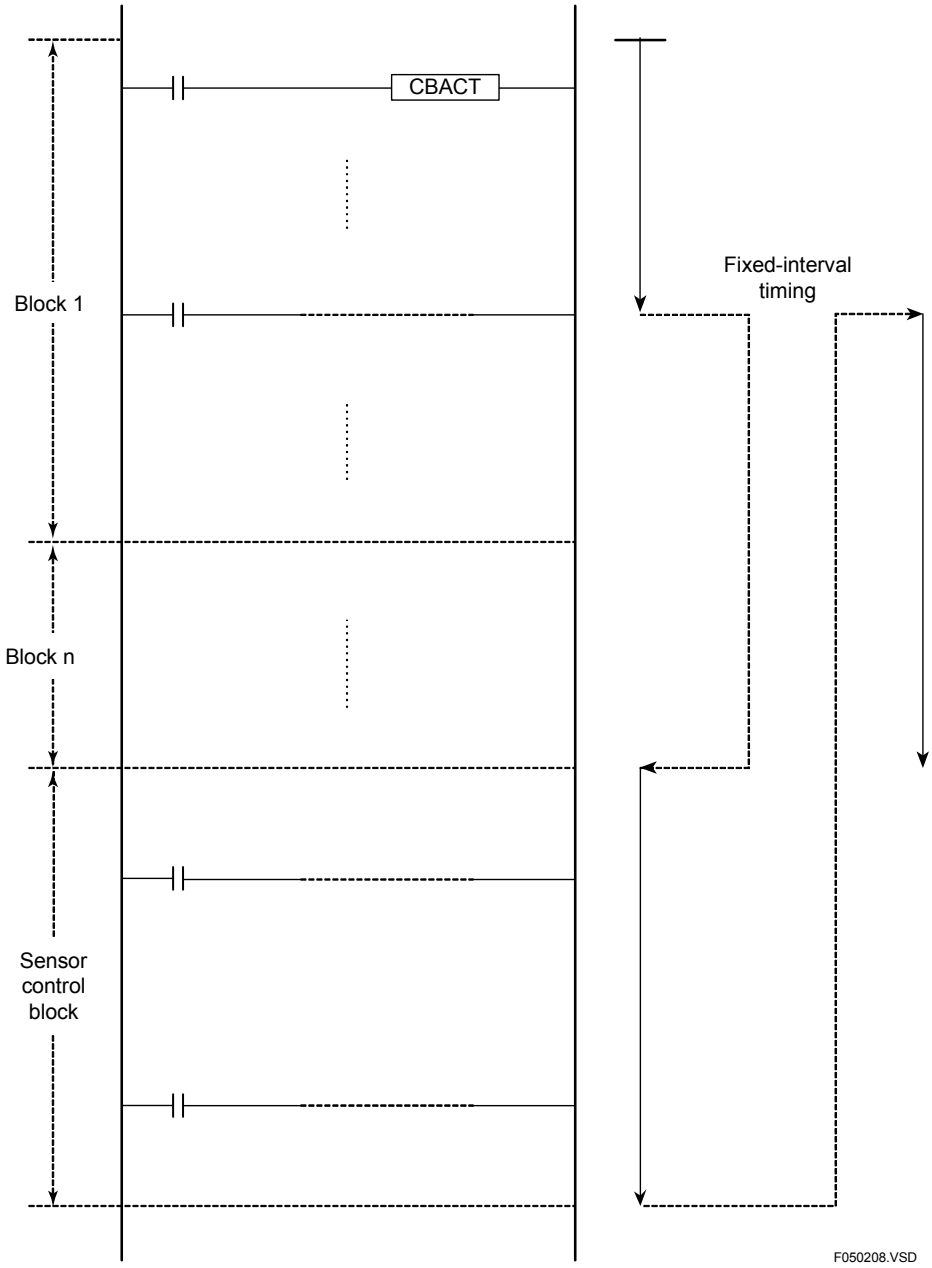


Figure 5.10 How the Sensor Control Block Is Executed

5.3 Program Memory

The program memory contains programs as well as information required for program execution and management. This section describes the structure of the program memory and its initial state with no program.

Table 5.1 Structure of Program Memory and Its Initial State

Component	Description	Initial State
Program management table	An area for storing information required for managing all programs including program name, step count, and block management information.	In the initial state, the program name is "PROGRAM, the " block name is "PROGRAM," and the number of steps is zero.
Program	An area for storing programs.	Contains a NOP instruction.
Configuration table*	An area for storing configuration information, such as device capacities and operation methods.	Contains the initial values discussed in Subsection 1.2.3, "Configuration."
I/O configuration table*	An area for storing configuration information such as I/O module setup and output mode (Hold/reset) in case sequence stops.	Contains the initial values discussed in Subsection 1.2.3, "Configuration."
Program control instructions table	An area for storing information required for managing the execution of program control instructions, such as JMP instructions and subroutine instructions.	Contains "0," indicating that there are no program control instructions such as JMP or subroutine instructions.
Timer/counter setpoint table	An area for storing timer and counter setpoints.	Contains "0," indicating that there are neither timers nor counters.
Utility	An area for storing information such as circuit comments and subcomments.	Contains "0".

*: See subsection 1.2.3, “Configuration,” for more information.

CAUTION

No program can be executed when the program memory is in its initial state.

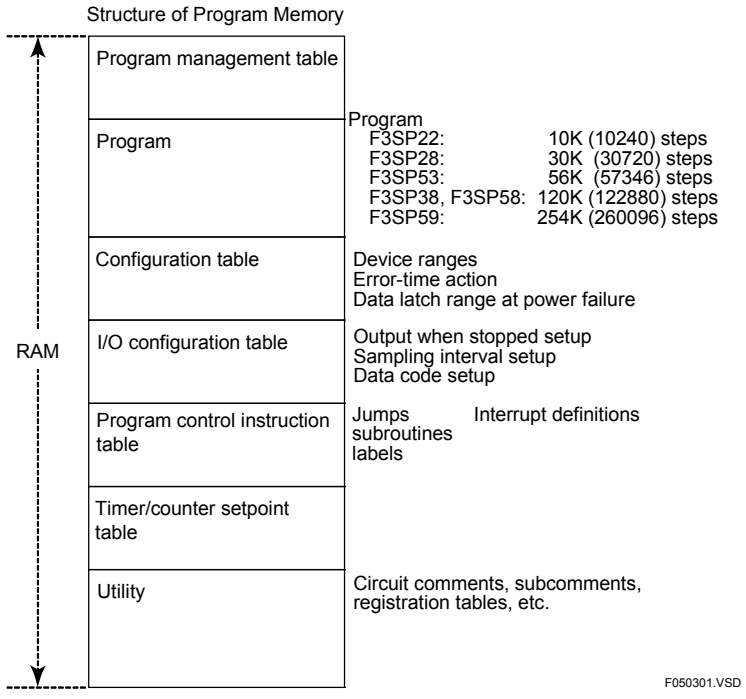


Figure 5.11 Structure of Program Memory

Blank Page

6. Functions

This chapter describes the functions provided by the sequence CPU module, such as the execution of specified blocks and debugging operations.

6.1 Function List

The following tables summarize the functions provided by the sequence CPU module and add-on CPU modules.

Table 6.1 Functions Provided by Sequence CPU Modules and Add-on CPU Modules

Functions of Sequence CPU Module	Function Overview	Page
Operation setup functions	Specify the operating mode of the sequence CPU module and its actions.	6-2
Constant scan	Executes a sequence program at certain time intervals.	6-4
Executing all blocks/specified blocks	Specifies how an executable program is processed. Specified blocks are executed using ACT and INACT instructions.	6-5
Debugging functions	Functions that support debugging, such as forced set/reset.	6-12
Program protection	Protect programs by means of password. These functions have two modes: executable program protection and block protection.	6-14
Online editing	Make on-line modifications or changes to a program in the program memory of the sequence CPU module.	6-16
Sampling trace functions	Acquire and displays states of multiple devices for up to 1024 scans.	6-24
Personal computer link functions	Perform communications equivalent to that of a personal computer link module, when a personal computer or a monitor is connected to the programming tool connector port.	6-27
Macro instructions	Allow the user to create and register new, customized instructions.	6-43
User log management functions	Allow the user to keep a log of, or record of, errors in the user's system, the way they occurred, the system's operating condition, and so on.	6-58
Sensor control functions	Execute a single block at high speed and at fixed intervals separately from the normal scan.	6-59
Partial download functions	Download specified blocks or macros only.	6-73
Functions for storing comments to CPU	Store circuit comments and subcomments to a sequence CPU module.	6-74
Functions for storing tag name definitions to CPU	Store tag name definitions to a sequence CPU module.	6-77
Structures	Represent a group of data items under a unified name.	6-78

Table 6.2 ROM Management (Writer) Functions

ROM Management (Writer) Functions	Function Overview	Page
File-to-ROM transfer function	Writes programs or data to the ROM pack	6-22
CPU-to-ROM transfer (ROM copy) function	Writes a program or data to the ROM pack	6-22
Compare file and ROM pack function	Compares the contents in the ROM to the program in WideField3 (or WideField2).	6-22
Clear ROM pack function	Erases ROM pack data.	6-22

Table 6.3 Device Management Functions

Device Management Functions	Function Overview	Page
Upload device data	Reads device information (data) from the sequence CPU module and saves it to a WideField3 (or WideField2) file.	6-42
Download device data	Reads device information (data) from a WideField3 (or WideField2) file and writes it to the sequence CPU module.	6-42
Edit device data	Edits device information (data) saved in a WideField3 (or WideField2) file.	6-42
Compare device data	Compares device information saved in the sequence CPU module with that saved in a WideField3 (or WideField2) file.	6-42

6.2 Operation Setup Functions

The operation setup functions set up the sequence CPU module operating mode and initializes programs and devices. You can set up operation by issuing a command from WideField3 (or WideField2), personal computer link module, or an add-on CPU module.

■ Run Mode

In Run mode, the CPU begins running a program from its first instruction, similarly to when the power is turned on. When the power is turned on or the operating mode is changed from Stop mode to Run mode, the CPU sets all devices to 0, except for latching-type devices, before executing the program. When the CPU switches to Run mode, functions that are available only in Debug or Stop mode are disabled.

■ Debug Mode

In Debug mode, the CPU begins running a program from its first instruction, similarly to when the power is turned on. When the operating mode is changed from Stop mode to Debug mode, the CPU sets all devices to 0, except for latching-type devices, before executing the program. Be sure to exit from Debug mode and enter Run mode after debugging and tuning.

■ Stop Mode

In Stop mode, the CPU stops running the program. The CPU either hold or reset external outputs depending on the settings of the configuration item “Output when stopped.” This function does not work when the CPU has already stopped running the program.

■ Clear Memory

Stop

This function deletes a program or programs and sets all devices except file registers (B) to 0.

You must stop running the program before using this function.

■ Clear Devices

Stop

This function sets all latching-type devices except file registers (B) to 0.

You must stop running the program before using this function. To clear file registers (B), use the edit device function of the device management functions to set all the file register (B) data to 0 and then write the data to the sequence CPU module using the write device function.

SEE ALSO

For details on the device management functions, see Section 6.12 “Device Management Functions.”

CAUTION

Observe the following precautions when using the functions described in this chapter:

- Some functions are only available in some but not all of the operating modes.
- The following marks are used when explaining a function to indicate that the function is available in the cited mode or modes.

  Run Debug Stop

  If no mark is indicated, it means that the function can be used in all operating modes.
- Some functions may lengthen the scan time.

  Be sure to disable such functions after use and before actual operation.

  Be especially careful when using any function that is enabled in Debug mode. Always disable the function and enter Run mode after debugging and tuning.
- Be sure to use the ROM writer functions when operating the ROM pack.
- The following mark is used when explaining each ROM writer function to indicate that the function is available in ROM Writer mode.

  ROM writer

6.3 Constant Scan

Run Debug

The constant scan function executes a program repeatedly at certain time intervals.

You can set the constant scan time, i.e., constant-scan time interval, to a value between 1 ms and 190 ms in 0.1 ms increments using the configuration function.

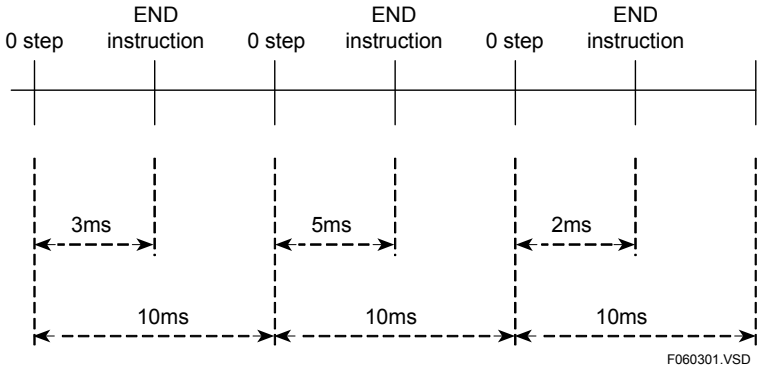


Figure 6.1 Operation Based on 10-ms Constant Scan

If the scan time of a sequence program is longer than the preset constant scan time, the constant scan setting is ignored and the program is executed using its own scan time.

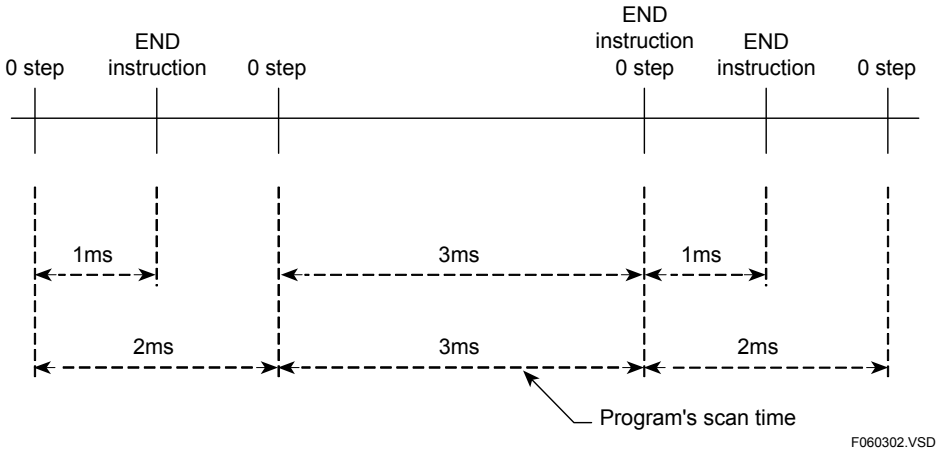


Figure 6.2 Operation Based on 2-ms Constant Scan

6.3.1 Setting the Constant Scan Time

You can set the constant scan time using the system configuration of project setting in WideField3 (or “Operation Control” of configuration of WideField2).

You can set the constant scan time to a value between 1 ms and 190 ms in 0.1-ms increments. To disable constant scan, select the option “Do not use” (default).

CAUTION

- The constant scan time must be shorter than the scan timeout interval.
- If the constant scan time is longer than the scan timeout interval, a scan timeout error occurs.

6.4 Executing All Blocks/Specified Blocks

Run Debug

Select the program execution mode ("All Blocks" or "Specified Blocks") using the executable program configuration of project setting in WideField3 (or “Operation Control” of configuration of WideField2).

6.4.1 Executing All Blocks

This mode executes all blocks of an executable program sequentially from block 1. The default program execution mode is “All Blocks.”

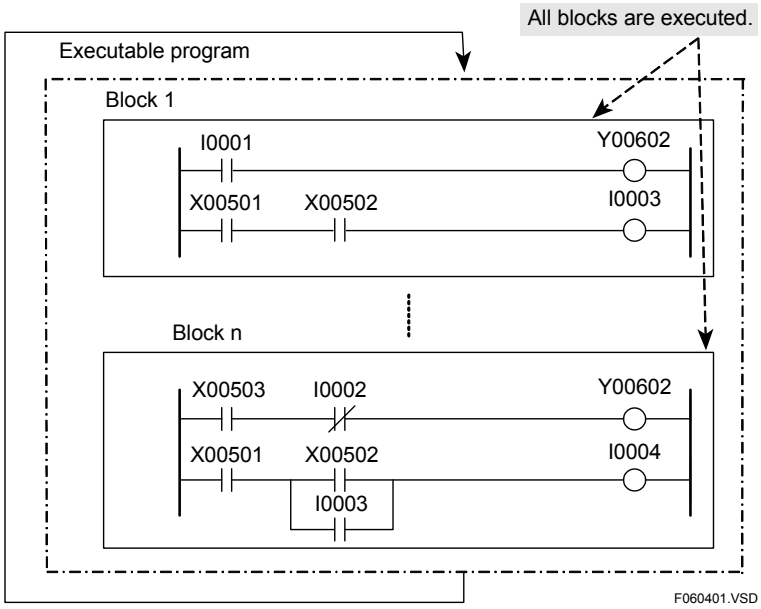


Figure 6.3 Executing All Blocks

6.4.2 Executing Specified Blocks

This mode allows you to specify selected blocks of an executable program for execution using ACT/INACT instructions.

In this way, you can control the execution of blocks, which are created on per function basis in modular programming.

Blocks to be executed are said to be "active" while blocks not to be executed are said to be "inactive." Use an ACT instruction to activate a block and an INACT instruction to inactivate a block. Whether each block is active or inactive is indicated by a special relay (M) given below:

- Special relays M2001 to M3024 for blocks 1 to 1024.

(Note that special relays M0001 to M0032 have the same values as special relays M2001 to M2032.)

The special relay for a block is set to “1” when the block is active and “0” when the block is inactive.

Active blocks are executed in ascending order of their block numbers. By default, only block 1 is active.

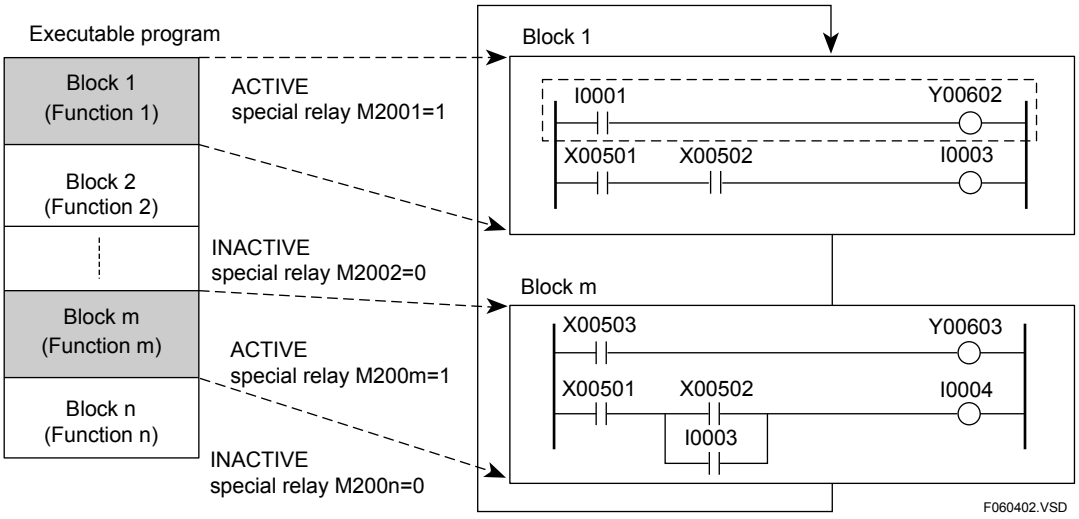


Figure 6.4 Execution of Specified Blocks (Executing block 1 and block m only)

6.4.3 Operation When Specified Blocks Are Activated

A block that is specified for activation by an ACT instruction is initialized at the end of that scan, and is actually started in the next scan.

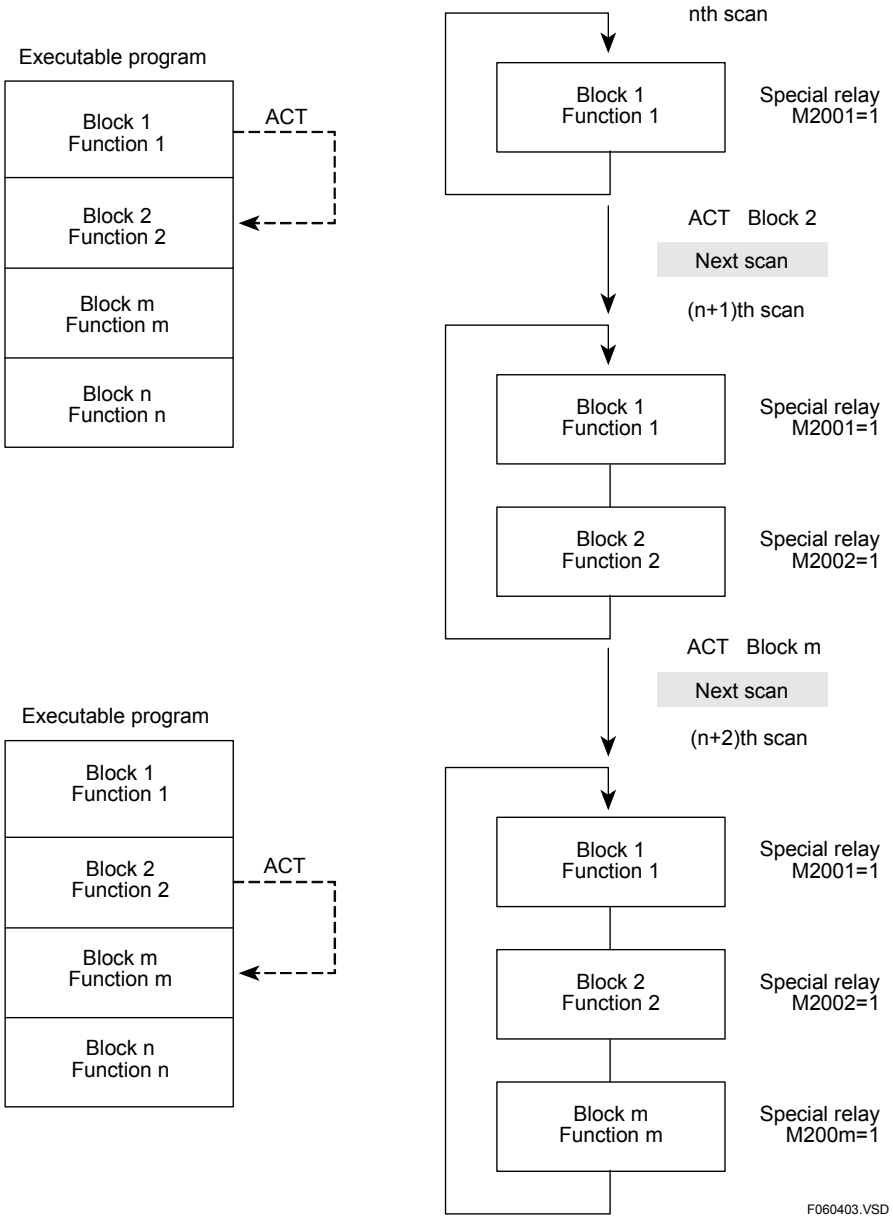


Figure 6.5 Operation When Specified Blocks are Activated

Devices that are used in a block, which is activated by an ACT instruction, are put into the following states by block initialization.

Table 6.4 State at Block Activation

Device	State at Block Activation
Timer (T)	Resets.
Continuous timer	Retains the value held before block activation.
Counter (C)	Retains the value held before block activation.
Destination of OUT instruction	Goes into an OFF state.
All other devices	Retains the states held before block activation.

Use a SET instruction for a device in a block whose output value is to be retained when the block is activated.

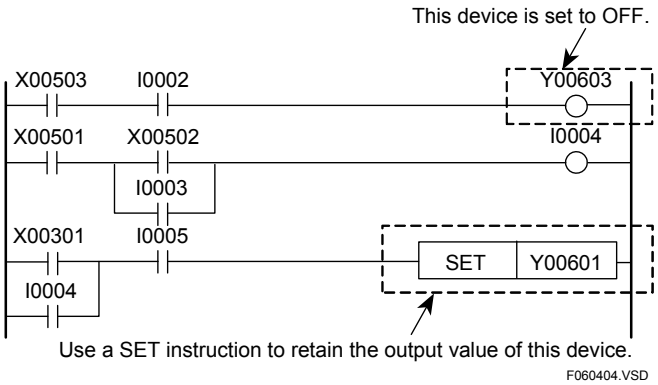


Figure 6.6 Example of Devices Initialized When a Block is Started

6.4.4 Operation When Specified Blocks Are Inactivated

A block that is specified for inactivation by an INACT instruction is initialized at the end of that scan, and is actually stopped in the next scan.

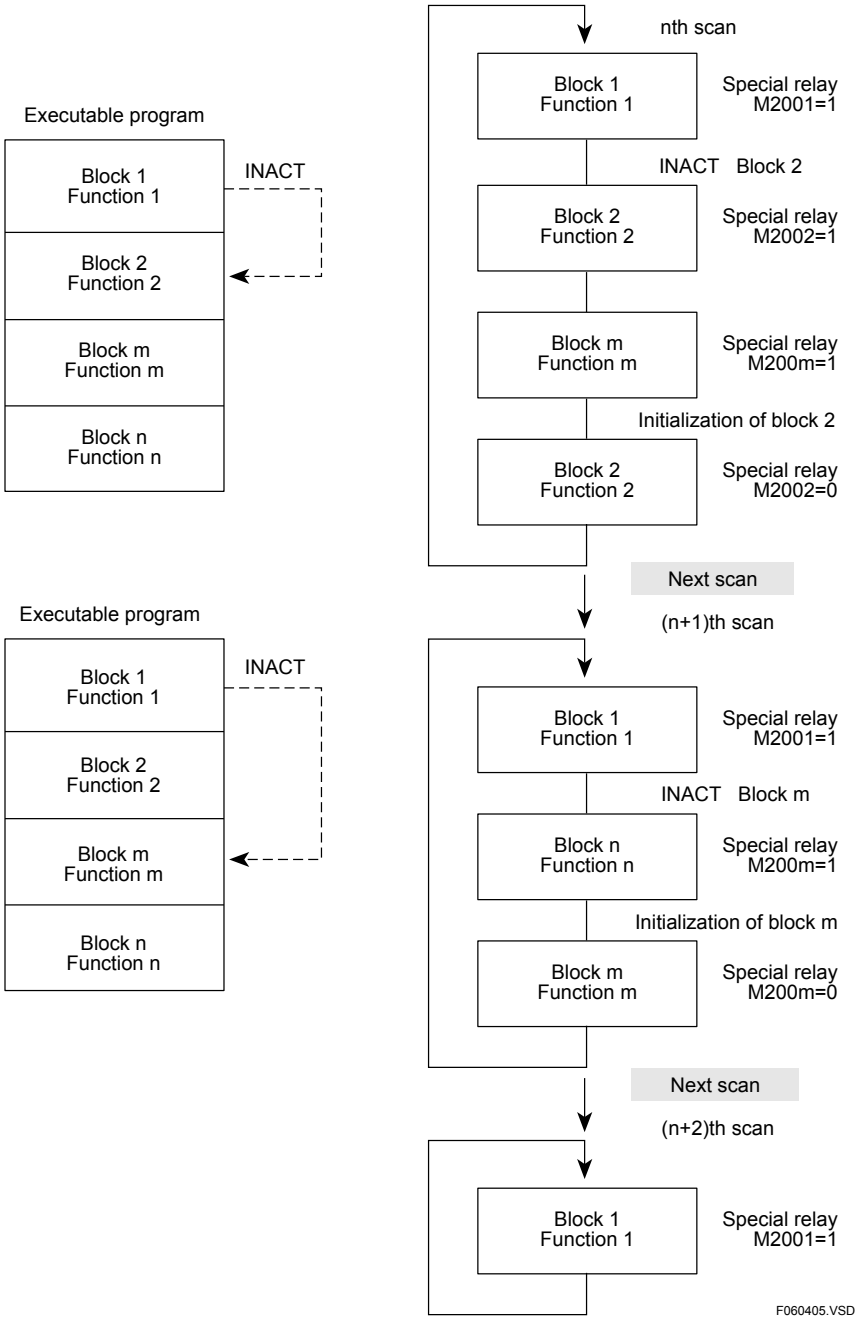


Figure 6.7 Operation When Specified Blocks Are Inactivated

Devices that are used in a block, which is stopped by an INACT instruction, are put into the following states by block initialization.

Table 6.5 State at Block Inactivation

Device	State at Block Inactivation
Timer (T)	Resets.
Continuous timer	Retains the value held before block inactivation.
Counter (C)	Retains the value held before block inactivation.
Destination of OUT instruction	Goes into an OFF state.
All other devices	Retains the states held before block inactivation.

Use a SET instruction for a device in a block whose output value is to be retained when the block is inactivated.

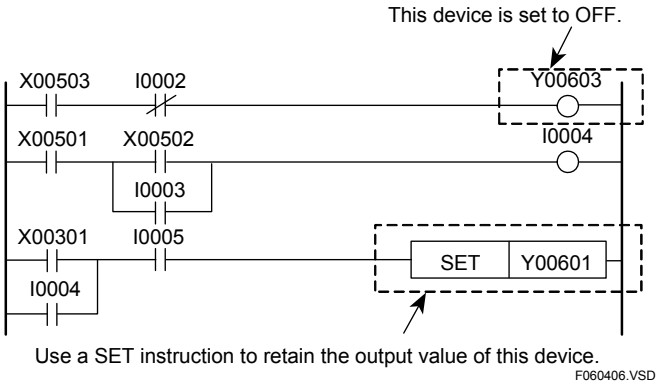


Figure 6.8 Example of Devices Initialized When a Block is Inactivated

6.4.5 Operation When Specified Blocks Are Executed

● Example Where Each Block Controls the Next Block to Be Activated

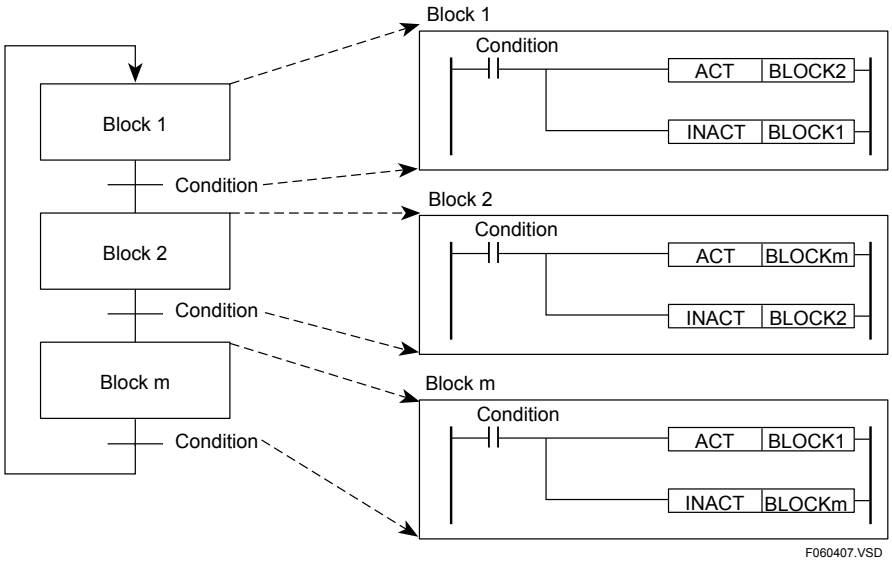


Figure 6.9 Example Where Each Block Controls the Next Block to Be Activated

● Example Where Block Activation is Controlled by a Scheduler

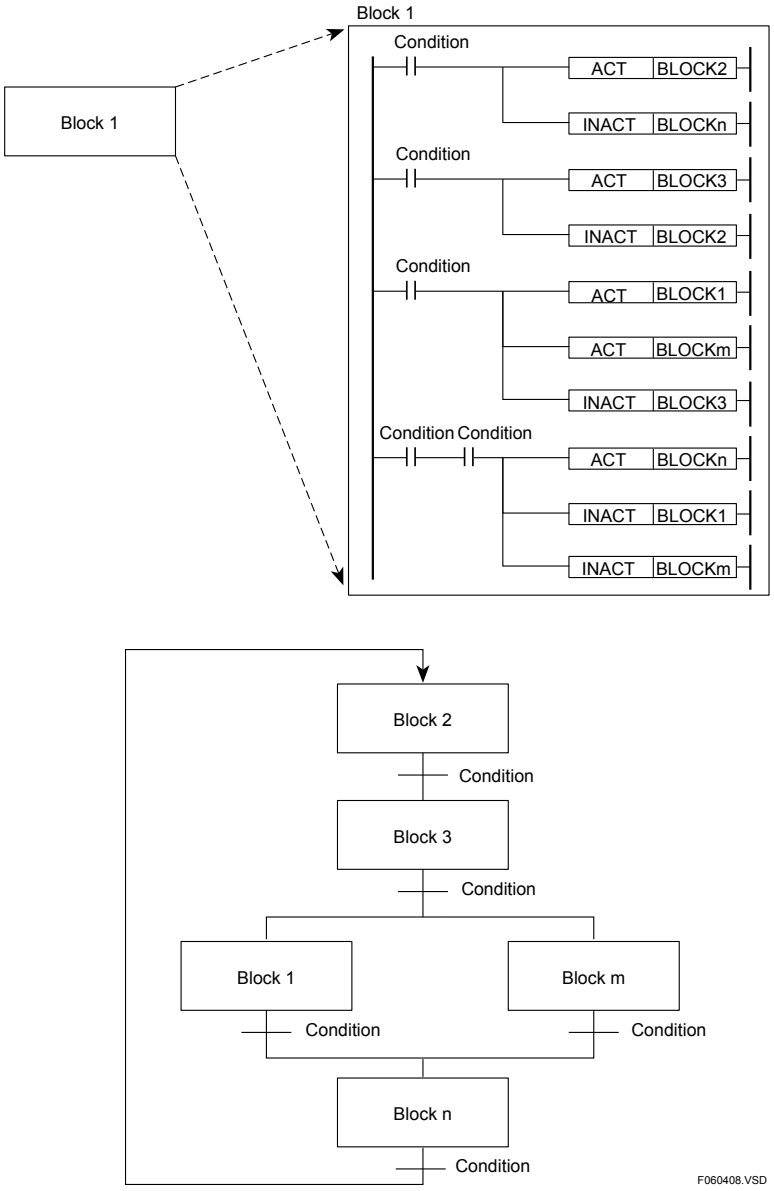


Figure 6.10 Example Where Block Activation is Controlled by a Scheduler

Create a scheduler using block 1 which is active by default.

6.5 Debugging Functions

This section describes the following functions: forced set/reset function for forcibly changing the status of a relay, functions for changing setpoints, current values and data values of registers, as well as the stop refreshing function for stopping I/O refreshing, link refreshing and shared refreshing.

6.5.1 Forced SET/RESET

Debug Stop

A forced SET/RESET forcibly sets a specified bit device to ON/OFF, regardless of program execution. You can apply forced set or forced reset to a maximum of 32 bit devices at one time. Only bit devices are supported (i.e., X, Y, I, E, L, T and C devices).

If a forced SET is applied to a timer (T) or a counter (C), the timer expires or the counter terminates.

A forced SET or forced RESET remains valid until you perform any of the following:

- Cancel the forced set or forced reset
- Change the operating mode to RUN mode
- Turn off the power
- Download a program or other data

CAUTION

Before turning off the power, cancel all instances of forced set or forced reset.

6.5.2 Changing Setpoints, Current Values and Data Values

Debug Stop

- **Changing Setpoints**

  You can change the setpoints of timers (T) and counters (C).

- **Changing Current Values**

  You can change the current values of timers (T) and counters (C).

  If you set a current value of “0”, a timer expires and a counter terminates.

- **Changing Word or Long-word Data Values**

  You can change the data values of word devices other than timers (T) and counters (C), such as data registers (D). If you specify a bit device such as an internal relay (I) instead of a word device, 16 or 32 bits of device data are changed, beginning with the first device address.

6.5.3 Stopping Refreshing

Debug | Stop

You can prevent external equipment (input relays (X) and output relays (Y)), FA link or FL-net systems (relays (L) and link registers (W)), as well as add-on CPU modules (shared relays (E) and shared registers (R)) from being refreshed by the results of program execution. This allows you to visually check I/O data on the monitor.

In the case of relays (input relays (X) and output relays (Y)) for external equipment, you can stop refreshing X input relays and Y output relays separately.

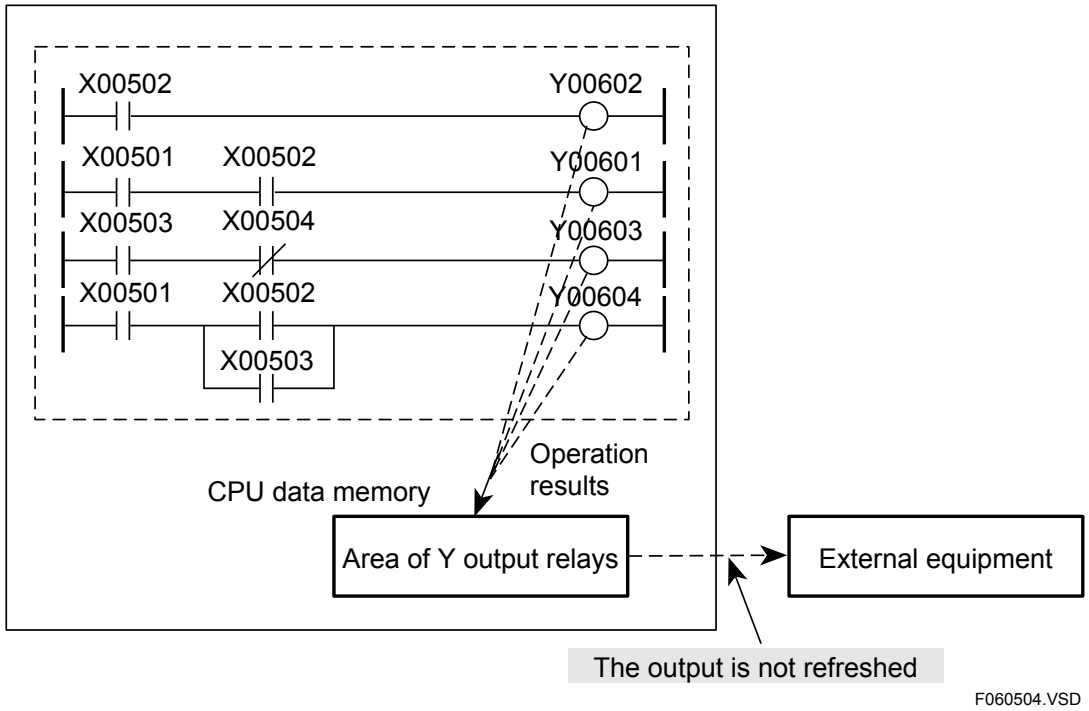


Figure 6.11 Stopping Output Refreshing

CAUTION

Refreshing of input relays (X) and output relays (Y) for external equipment, specified in the sensor control block, cannot be stopped.

6.6 Program Protection

You can protect your programs against unauthorized access for security reasons. There are two modes of protection: executable program protection and block protection. Protection is enabled by defining a password using WideField3 (or WideField2). A password must consist of eight alphanumeric characters, beginning with a letter. The protection information is saved with an executable program or block by WideField3 (or WideField2).

CAUTION

Program protection is only designed to prevent unauthorized read access. It does not protect against program deletion or CPU operation modification due to erroneous operations or writing.

6.6.1 Executable Program Protection

Executable program protection protects an entire executable program.

When this protection is enabled, all functions that act upon an executable program (downloading, uploading, monitoring, online editing, etc.) are prohibited.

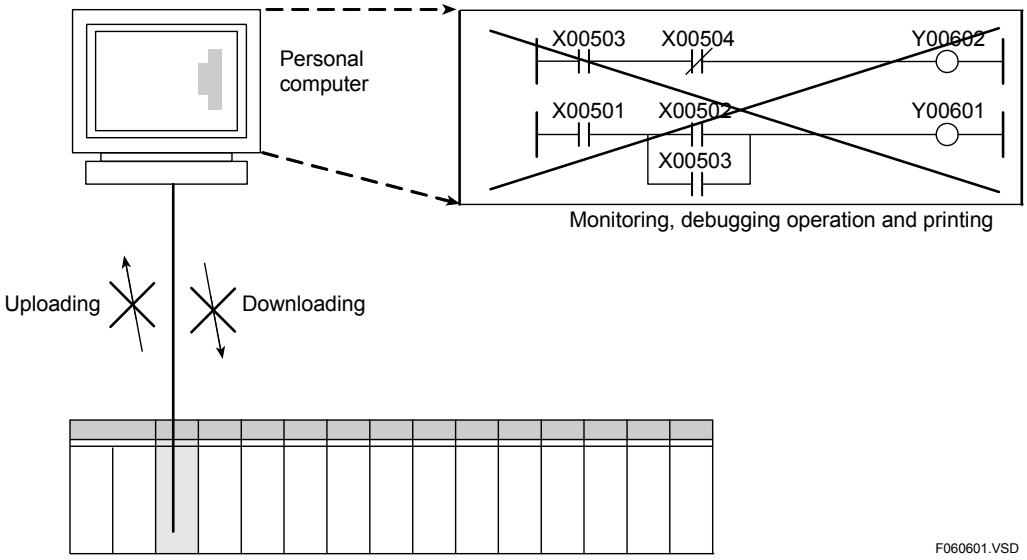


Figure 6.12 Executable Program Protection

When executable program protection is enabled, the following functions are prohibited:

Downloading, uploading, monitoring (circuit diagram monitoring, debug operation, changing timer (T)/ counter (C) setpoints, online edit), ROM writer functions, and printing.

6.6.2 Block Protection

Block protection protects programs on per block basis.

This protection mode is only designed to prevent unauthorized read access. In addition, only the specified blocks are protected. When block protection is enabled for a block, its circuit diagrams and instructions are not displayed in WideField3 (or WideField2).

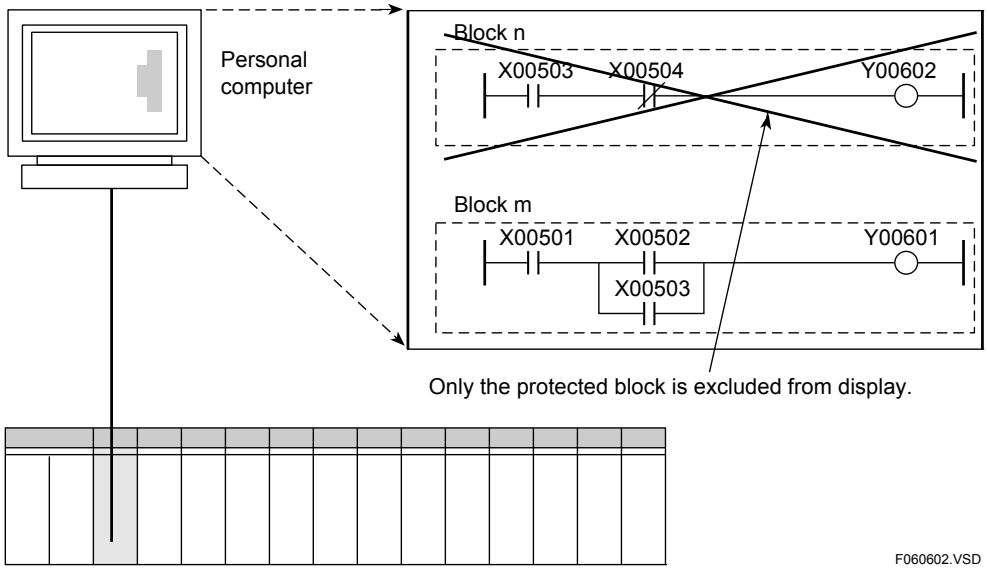


Figure 6.13 Block Protection

When block protection is enabled, the following functions are prohibited:

Monitoring (circuit diagram monitoring, debug operation, changing timer (T) /counter (C) setpoints, and online edit) and printing

6.7 Online Editing

Debug Stop

Online editing allows you to make modifications or additions to your program during program execution. This function is useful for making minor changes to the program during debugging or tuning. Modifications/changes made to the program are reflected in the program memory of the sequence CPU module at the end of a given scan.

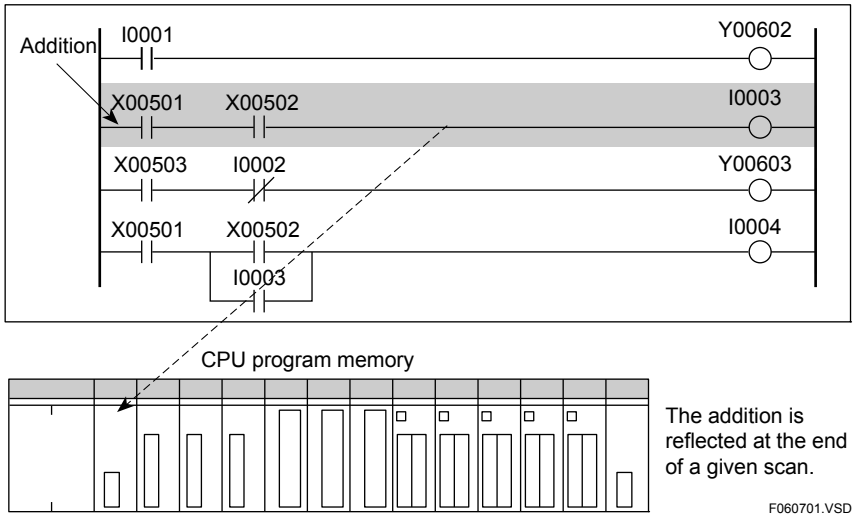


Figure 6.14 Online Editing

WARNING

- Do not perform online editing when machinery under control is in operation. When online edited data is written to the sequence CPU module, scan time may become much longer than usual. Scan time lengthens by as much as 10 ms for every 10K step increase in the program size. During this time, external refreshing or communications with external equipment are not allowed.
- Edited changes are reflected to the CPU module at the end of conversion, line deletion or online edit operations in WideField3 (or WideField2). Special considerations of sequence processing apply during this update process before all changes are reflected.
- If there is a differential type instruction in a circuit that is modified or added online, or in the circuit following a circuit that is modified, inserted or deleted online, beware that the instruction will be executed as if its preceding value is OFF. This means that the instruction may cause a differential output even if its input condition is always ON. (Differential type instructions include the following: LDU, LDD, UP, DWN, UPX, DWNX, DIFU, DIFD, FF, TIM, CNT, SFTR and input differential type instruction.)
- If a circuit that contains a timer instruction is modified or added, the time-out relay may stay ON when the timer input is OFF or the timer may not start running when the timer input is ON, depending on the preceding value. In this case, the rising edge of the next timer input enables normal operation.

CAUTION

(1) You are not allowed to modify the following instructions and circuits.
- Subroutine Entry (SUB) instruction and Subroutine Return (RET) instruction as well as circuits that contain any of these instructions.
- Interrupt (INTP) instruction and Interrupt Return (IRET) instruction for input modules, as well as circuits that contain any of these instructions.
- Structure Macro Instruction Call (SCALL) instruction, Structure Move (STMOV) instruction, as well as circuits that contain any of these instructions.

(2) Online editing affects peripheral processing.
Peripheral processing time may lengthen by approximately 200 ms, though this depends on the program size or the location in the program where modifications are made. During this time, the CPU does not perform shared refreshing, link refreshing or command processing.

6.8 Making Programs Resident Using ROM Writer Functions

This section describes how to make programs resident in ROM, setting initial values of devices to be resident in ROM, as well as ROM management (writer) functions and ROM writer mode.

6.8.1 Making Programs Resident in ROM

Programs that have been debugged and tuned can be made resident in the ROM pack. To make a program resident in the ROM pack, transfer the program to the ROM pack using the ROM writer functions of the sequence CPU module.

The items to be made resident in the ROM pack are programs themselves, as well as program management information, configuration information, control tables, timer (T)/counter(C) setpoint tables, and comment information.

All information required by the CPU to start program execution at power-on is made resident in the ROM pack.

The maximum limits on program steps that can be stored on various ROM packs depend on whether only programs are stored in the ROM pack, or tag name definitions are stored along with programs, as shown in the table below.

Table 6.6 Limitations on Selection of ROM pack

Sequence CPU Module	Storage	Program	Program + Tag Name Definition
F3SP28-3N	RK33-0N	30K steps	No tag name definition can be stored.
	RK73-0N		
F3SP53-4H	RK33-0N	56K steps	
	RK73-0N		
F3SP38-6N F3SP58-6H	RK33-0N	56K steps	
	RK73-0N	120K steps	
F3SP22-0S	RK33-0N	10K steps	56K steps
	RK73-0N		120K steps
F3SP28-3S	RK33-0N	30K steps	56K steps
	RK73-0N		120K steps
F3SP53-4S	RK33-0N	56K steps	56K steps
	RK73-0N		120K steps
F3SP38-6S F3SP58-6S	RK33-0N	56K steps	56K steps
	RK73-0N	120K steps	120K steps
	RK93-0N	120K steps	360K steps
F3SP59-7S	RK33-0N	56K steps	56K steps
	RK73-0N	120K steps	120K steps
	RK93-0N	254K steps	360K steps

Using configuration, you can make the following two types of data resident in ROM. These types of data are used to set initial values to be used by a program.

- Setpoints of 1,024 data registers' (D) worth of default data
- Either 32768 data registers' (D) or file registers' (B) worth of current values within the sequence CPU module. See Section 6.8.2, "Defining Current Values of Devices to Be Made Resident in ROM."

At power-on, data read from the ROM pack is stored in data registers (D) or file registers (B) specified with the configuration function. Data registers (D) and file registers (B) included in the data retention in case of power failure revert to their respective default values. If you edit both of the configuration items mentioned above for the same data register (D), only the setpoint of the second configuration item is effective.

TIP

Data retention in case of power failure is effective for devices not included in the configuration discussed above.

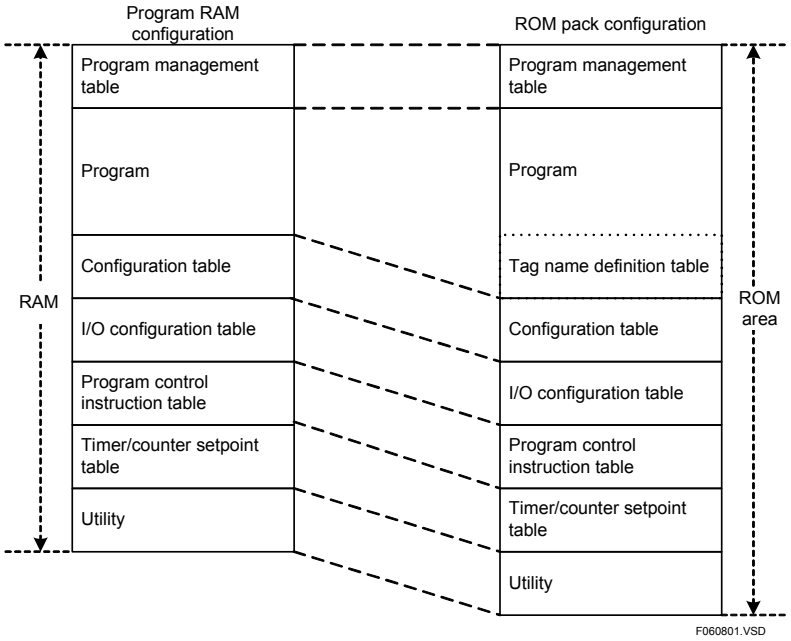


Figure 6.15 Contents of Program to Be Made Resident in ROM

■ Compatibility of a Sequence CPU Module with ROM Pack Data Written by another Sequence CPU Module

● For F3SP28-3S, F3SP38-6S, F3SP53-4S, and F3SP58-6S of Rev. 7 or earlier; and F3SP28-3N, F3SP38-6N, F3SP53-4H, F3SP58-6H, and F3SP59-7S

These sequence CPU modules can only execute programs written to ROM packs using the same CPU type as shown in the table below.

Example: The F3SP28-3N can only execute programs on ROM packs written using F3SP28-3N. Thus a ROM pack written by F3SP28-3S may not be installed and used with F3SP28-3N.

Table 6.6.1 Compatibility of a CPU with ROM Pack Data Written by another CPU (1/2)

CPU Compatible with Written ROM Pack Data	CPU Used to Write Program Data to ROM Pack
F3SP28-3N	F3SP28-3N
F3SP28-3S (Rev. 7 or earlier)	F3SP28-3S
F3SP38-6N	F3SP38-6N
F3SP38-6S (Rev. 7 or earlier)	F3SP38-6S
F3SP53-4H	F3SP53-4H
F3SP53-4S (Rev. 7 or earlier)	F3SP53-4S
F3SP58-6H	F3SP58-6H
F3SP58-6S (Rev. 7 or earlier)	F3SP58-6S
F3SP59-7S	F3SP59-7S

● For F3SP28-3S, F3SP38-6S, F3SP53-4S, and F3SP58-6S of Rev. 8 or later

These sequence CPU modules can execute programs written to ROM packs using the same CPU type or some other CPU type as shown in the table below.

Example: The F3SP28-3S can execute programs written to the ROM pack using F3SP28-3S or F3SP28-3N.

Table 6.6.2 Compatibility of a CPU with ROM Pack Data Written by another CPU (2/2)

CPU Compatible with Written ROM Pack Data	CPU Used to Write Program Data to ROM Pack
F3SP28-3S (Rev. 8 or later)	F3SP28-3S or F3SP28-3N*1
F3SP38-6S (Rev. 8 or later)	F3SP38-6S or F3SP38-6N*1
F3SP53-4S (Rev. 8 or later)	F3SP53-4S or F3SP53-4H*1
F3SP58-6S (Rev. 8 or later)	F3SP58-6S or F3SP58-6H*1

*1: A compatible CPU may take several extra seconds to start up from ROM pack data written by a different CPU type. The actual duration depends on the program size. Note that CPUs of different types may calculate the number of timer instruction steps and comment steps differently. Thus a ROM pack written by a CPU may result in a ROM pack error when executed on a compatible but different CPU type and cannot be accessed due to an excessive step count.

CAUTION

Do not use the CPU-to-ROM transfer function to directly write a program read to CPU from a ROM pack written by another CPU to the ROM pack.

Write the program to the ROM pack using the File-to-ROM transfer function after changing the CPU type by uploading the program with the programming tool, or using the CPU-to-ROM transfer function after downloading the program.

6.8.2 Defining Current Values of Devices to Be Made Resident in ROM

Define initial values to be made resident in data registers (D) or file registers (B) when program execution begins.

The data items to be defined are the type of device, starting number, and quantity of devices. This configuration enables the current values of the specified devices to be stored in the ROM pack when the “file-to-ROM transfer” or “CPU-to-ROM transfer (ROM copy)” function of the ROM writer functions is executed. You can determine whether or not to update the device data to be made resident in ROM with the current values when executing the “file-to-ROM transfer” or “CPU-to-ROM transfer” function. When program execution begins, the device data in the ROM pack is read and stored in the specified devices. This configuration is useful when you want to set a large volume of initial data or save initial data for a program. You can set initial values in a maximum of 32768 devices.

6.8.3 ROM Writer Functions and ROM Writer Mode

ROM writer

The sequence CPU module or an add-on CPU module can be operated by reading a program stored in the ROM pack. In the FA-M3 R series, you can achieve the same functions as those of a commercially available ROM writer, such as writing a program to the ROM pack, by using the sequence CPU module or add-on CPU module. These functions are called the ROM writer functions and include file-to-ROM transfer, CPU-to-ROM transfer, and file and ROM pack comparison. The ROM writer functions work in a dedicated mode different from the normal operating mode of the sequence CPU module. This dedicated mode is called the **ROM Writer mode**. The **ROM Writer mode** is maintained even when you turn on or off the power. At power-on, no programs are read from the ROM pack.

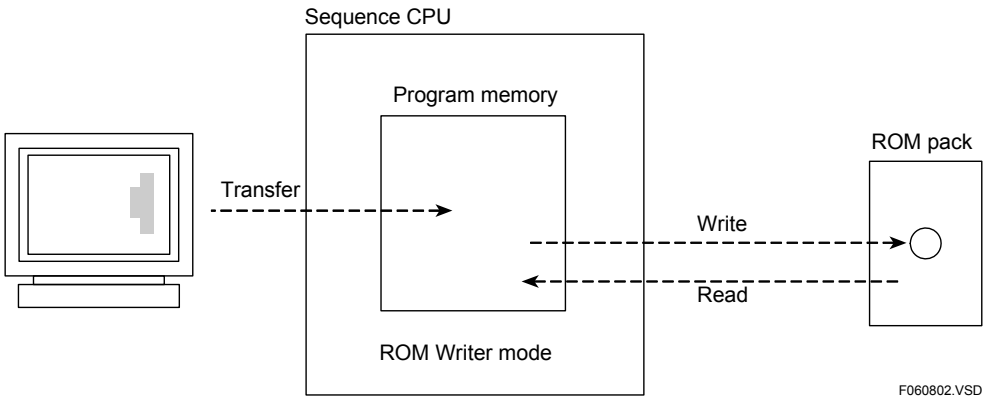


Figure 6.16 ROM Writer Functions and ROM Writer Mode

Using the ROM writer functions, you can save a debugged and/or tuned program to the ROM pack. To transfer the program to the ROM pack, use the ROM management function of WideField3 (or WideField2). The following details the ROM writer functions.

■ File-to-ROM Transfer Function

This function writes programs, data, and a tag name definitions to a ROM pack.

It first transfers a program to the CPU memory, and then writes the program to the ROM pack. You can specify whether to make the current values of devices resident in ROM and whether to write tag name definitions to the ROM pack. Tag name definitions, if to be written, are written directly to the ROM pack without first being transferred to the CPU memory.

■ CPU-to-ROM Transfer (ROM Copy) Function

This function writes a program or data in the CPU directly to the ROM without transferring it using the ROM management function of WideField3 (or WideField2). You can specify whether to make the current values of devices resident in ROM and whether to write tag name definitions to the ROM pack.

A debugged and/or tuned program or data in the CPU is not initialized when the CPU is changed to the ROM Writer mode. It is therefore possible to write the program or data directly to ROM. If tag name definitions are downloaded in the CPU, they are also written directly the ROM pack. This function is also used to write the same program to multiple ROM packs. You can write the program to multiple ROM packs by simply changing the ROM packs one after another. There is no need to transfer the program repeatedly.

■ Compare File and ROM Function

This function compares the content of the ROM pack with the program in WideField3 (or WideField2). If the contents do not match, the function shows the mismatches.

■ ROM Clearance Function

This function erases the content of a ROM.

CAUTION

- Change the CPU to the **ROM Writer** mode before using the ROM writer functions. You cannot use the **ROM Writer** functions in other modes.
- Be sure to disable the ROM Writer mode when you finish using the ROM writer functions. The CPU does not execute any sequencing functions if the **ROM Writer** mode remains active.
- Never try to switch off the module when the ROM pack is being overwritten or cleared. Otherwise the ROM pack may develop a permanent error and become no longer usable.

6.9 Exclusive Access Control

This section describes exclusive access control, a function for restricting operations on program, operating mode, or device data by other users during operation or debugging.

Exclusive access control is used to prevent a program, operating mode, or device data from being changed or a program or device data from being downloaded by other users, say during operation or debugging.

Once you acquire an exclusive access control, all modification- and control-related commands issued from other tools, sequence CPU modules or personal computer links are rejected until you release the control.

While you hold the exclusive access control, all modification- and control-related commands from other users remain disabled so you should release the access control as soon as you have completed the required processing.

If another user has already acquired an exclusive access control, it is not available to you.

The following exclusive access control functions are provided:

● Get exclusive access control

This function acquires exclusive access control.

● Release

This function releases exclusive access control.

● Forced Release

This function allows a tool or module that has no exclusive access control to force holder of the exclusive access control to release it.

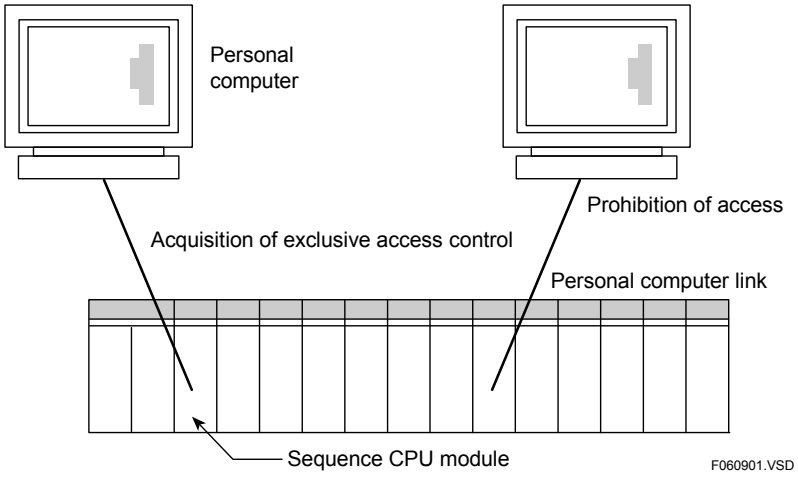


Figure 6.17 Exclusive Access Control

Once a user acquires exclusive access control, the system prohibits other tools or modules having no exclusive access control to perform the following operations:

Running or stopping a program, debugging, downloading, debug operation, use of debugging functions, writing to devices, and changing setpoints of timers (T)/counters (C).

6.10 Sampling Trace Functions

This section describes the sampling trace functions, which records state transitions of specified devices.

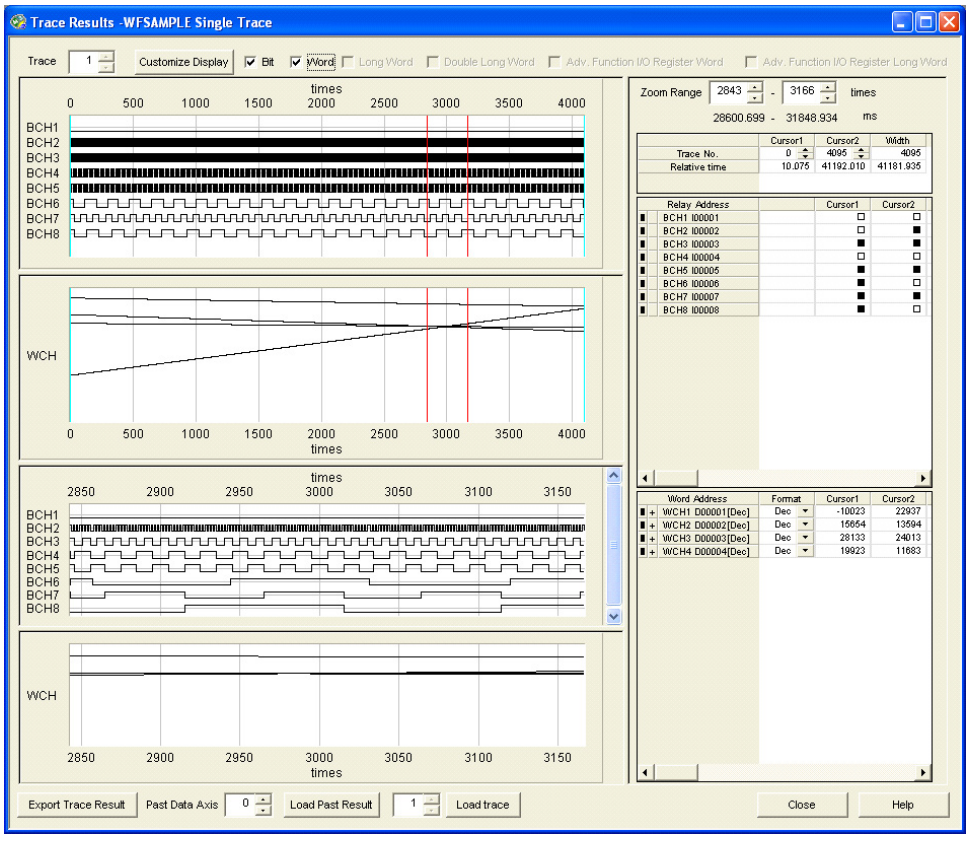
The sampling trace functions store the states and contents of devices selected to be sampled, sequentially in the sampling trace memory of the sequence CPU module.

Three sampling methods are available:

- TRC instruction sampling
- End-of-Scan Sampling
- Periodic sampling

You can define the trigger condition for sampling as the rising edge of a specified relay signal, the falling edge of a specified relay signal or data coincidence with a selected register device. The CPU monitors the trigger condition during scan end processing. If the trigger condition becomes true, the CPU takes 1024 samples, starting from a specified negative delay before or a specified positive delay after the condition becomes true.

Using WideField3 (or WideField2), you can configure the sampling trace functions, and subsequently view sampling trace results in time-chart format from WideField3, as shown in the figure below.



F061007.VSD

Figure 6.18 View of Sampling Trace Results (WideField3)

You can execute sampling trace in either Run or Debug mode. Re-executing sampling trace erases previous data. If you perform sampling trace setup using the configuration function, the CPU begins sampling immediately after power-on. If you perform sampling trace setup using the configuration function and then permanently store the setup to ROM, the CPU reverts to the ROM setup after a power-on-and-off sequence even if you have redefined the settings using the programming tool.

SEE ALSO

For details on how to configure the sampling trace functions, see, "FA-M3 Programming Tool WideField3" (IM 34M06Q16-□□E) or "FA-M3 Programming Tool WideField2" (IM 34M06Q15-01E).

How sampling is carried out is explained below.

● TRC Instruction Sampling

In TRC instruction sampling, the CPU samples the states and data of specified contacts whenever the Sampling Trace (TRC) instruction is executed. By executing the TRC instruction in a program, you can perform sampling at any point within a scan.

The CPU collects data when the input-condition relay of the Sampling Trace (TRC) instruction is set to ON. The CPU stores results of up to four cycles of sampling if the TRC instruction is executed multiple times within the same scan. Any fifth or subsequent Sampling Trace (TRC) instruction executions within a scan are ignored. Sampling trace results are stored at the end of a scan.

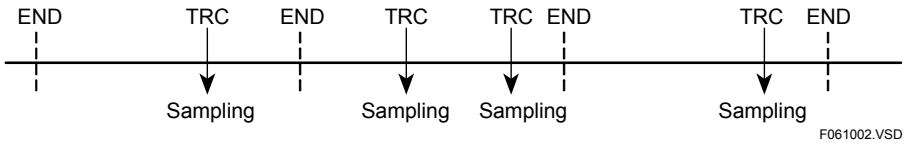


Figure 6.19 Sampling when the Sampling Trace (TRC) Instruction is Executed

● Scan Sampling

In scan sampling, the CPU samples the states and data of specified contacts at the end of a scan. It collects and stores the data each time the specified number of scans are completed.

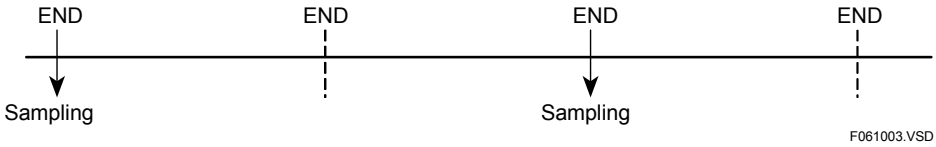


Figure 6.20 Scan Sampling at Two-scan Intervals

● Periodic Sampling

In periodic sampling, the CPU samples the states and data of specified contacts at fixed time intervals. It collects and stores the data after the specified period expires and before the next scan begins.

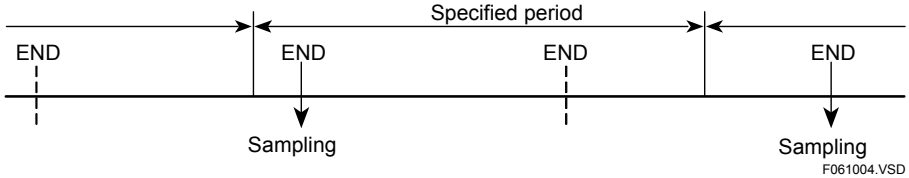


Figure 6.21 Periodic Sampling

CAUTION

The sampling trace functions check the trigger condition when an END processing in a program is performed. Therefore, if the trigger condition becomes true during program execution but becomes false again before processing of the END processing begins, sampling is not performed.

● Sampling when a Negative Delay Is Defined

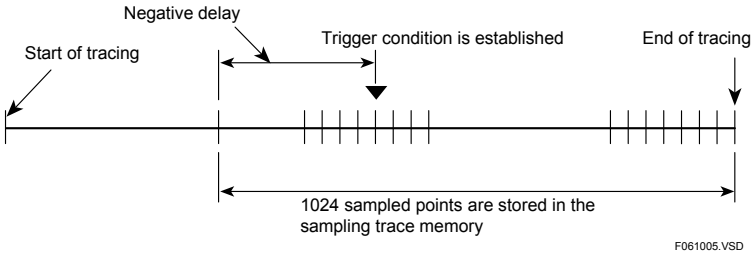


Figure 6.22 Sampling when a Negative Delay Is Defined

● Sampling when a Positive Delay Is Defined

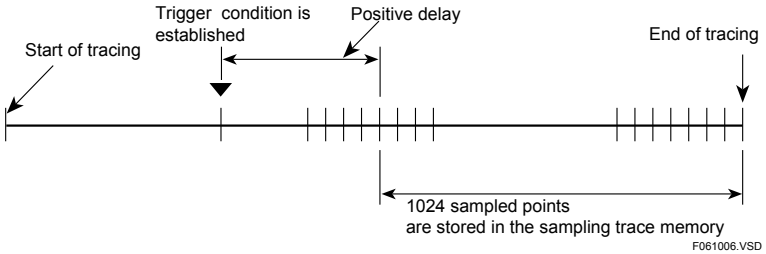


Figure 6.23 Sampling when a Positive Delay Is Defined

6.11 Personal Computer Link Functions

This section describes the personal computer link functions that allow a personal computer or a display device to be connected to a sequence CPU module.

The programming tool connector on the front of the CPU module functions in the same way as the RS232-C communication port on the F3LC11-1F personal computer link module. This means you can connect higher-level equipment, such as a personal computer or FA computer, or a monitor to the CPU module to perform one-to-one communication as you do with the personal computer link module. This feature is called the personal computer link functions.

You can monitor and configure devices, as well as start, stop, download and upload programs by entering commands from the higher-level computer.

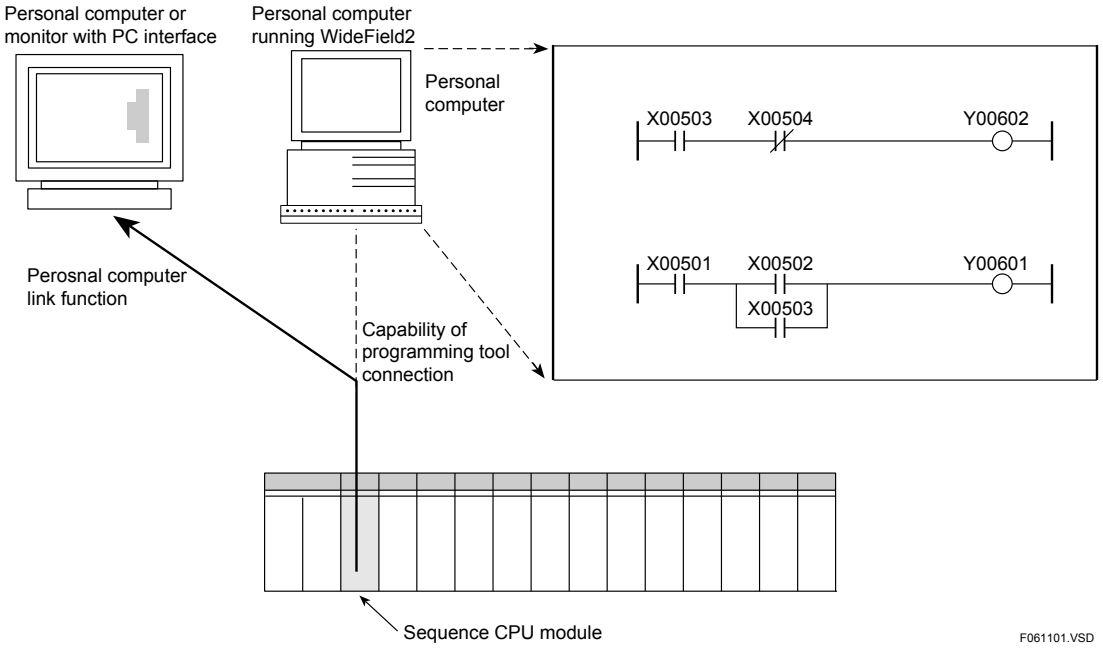


Figure 6.24 Personal Computer Link Functions

6.11.1 System Configuration

The figure below shows examples of system configuration using the personal computer link functions.

External equipment, such as a personal computer or monitor, is connected to the sequence CPU module of the FA-M3 by using the programming tool connector on the front of the FA-M3 and a dedicated programming tool cable.

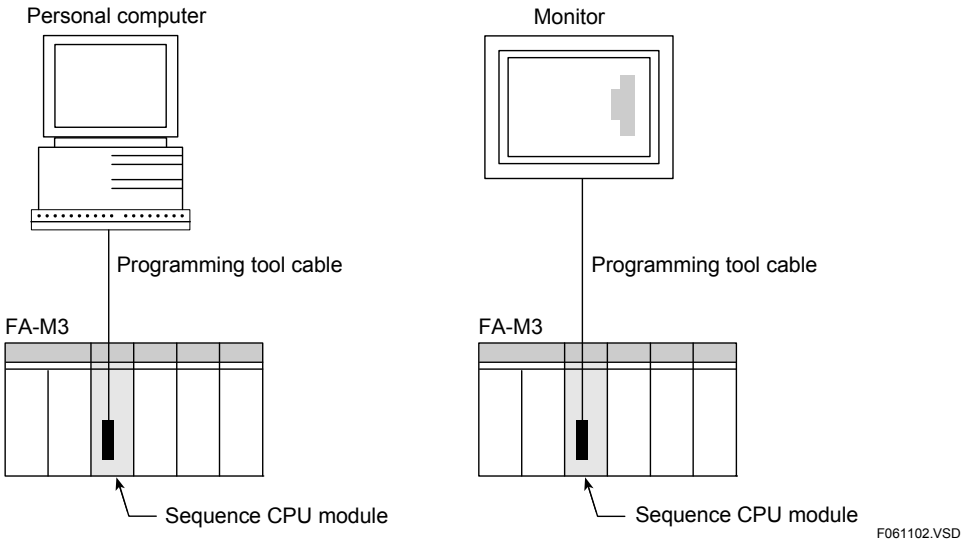


Figure 6.25 Examples of Connection between a Sequence CPU Module and External Equipment

Provide the programming tool cable with a ferrite core if you want to have the connected device compatible with the CE marking.

	Manufacturer	Product Series Name
Examples of ferrite cores	Kitagawa Industries K.K.	RFC series
	TDK Corporation	ZCAT series
	NEC TOKIN Corporation	ESD-SR series

6.11.2 Differences from Personal Computer Link Module

This subsection describes the differences between the F3LC1□-1F personal computer link module and the personal computer link functions of the sequence CPU module.

■ Function

The transmission rate and data format of the CPU's personal computer link functions differ from those of the personal computer link module. For details, see Subsection 6.11.4, "Setting Up the Personal Computer Link Functions."

Table 6.7 Transmission Rate and Data Format of CPU's Personal Computer Link Functions

Transmission Rate (bps)	Data Length	Parity	Stop Bits
9600	8 bits	Even	1 bit
9600	8 bits	None	1 bit
19200	8 bits	Even	1 bit
19200	8 bits	None	1 bit
38400	8 bits	Even	1 bit
38400	8 bits	None	1 bit
57600	8 bits	Even	1 bit
57600	8 bits	None	1 bit
115200	8 bits	Even	1 bit
115200	8 bits	None	1 bit

A dedicated programming tool cable is required to connect a personal computer or monitor to the CPU module. To set the transmission rate, data format, checksum, end character, and protection function, use the configuration item "Communication mode" (setup is by switches in the case of the personal computer link module). The event transmission function is not supported.

If the sequence CPU module receives an MDR module reset command as a PC line command, it resets only the communication port. The maximum number of personal computer link modules that can be installed remains the same even if the CPU's personal computer link functions are used.

■ Communications Protocol

A brief description of the communications protocol of the personal computer link functions is given below.

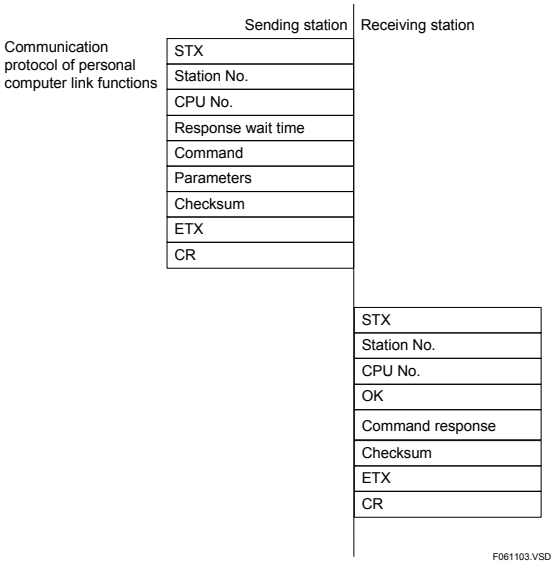


Figure 6.26 Communications Protocol of Personal Computer Link Functions

In personal computer link communication, the maximum size of text that can be transferred each time is 512 bytes.

6.11.3 Specifications of Personal Computer Link Functions

Table 6.8 Specifications of Personal Computer Link Functions

Item	Description	Setup *1
Interface	EIA RS-232-C compliant	
Transmission mode	Half-duplex transmission	
Synchronization	Start-stop synchronization	
Transmission rate (bps)	9600/19200/38400/57600/115200	✓
Data format	Start bit : 1	
	Data length : Fixed at 8 bits	
	Parity bit : None or Even	✓
	Stop bit : 1 bit (fixed)	
Error checking	Parity check	
	Checksum : Yes/No	✓
Control line (RS-232-C)	Not used.	
Xon/Xoff	Not used.	
Configurable item	Transmission rate, data format, checksum, end character and protection	✓
Protocol	Proprietary protocol	
End character	Yes/No	✓
Protection function *2	Yes/No	✓
Access range	Access to all control data, upload/download programs, CPU operation (Run mode)/stop (Stop mode), and read error logs	
Transmission distance	8 m max.	
External connection	Dedicated cable	

*1 The check mark ✓ indicates that a user can configure the item by using the configuration function. However, there are restrictions on the way the transmission rate and parity check are combined. See subsection 6.11.4, “Setting Up the Personal Computer Link Functions”, for more information.

*2 You can set the protection function to the Yes option to prevent inadvertent writing to the FA-M3.

CAUTION

The personal computer link functions use neither a control line nor Xon/Xoff characters. Be careful when using the function because a communication failure may occur at the higher-level equipment side, depending on the transmission rate.

6.11.4 Setting Up the Personal Computer Link Functions

This subsection describes the items you should define when using the personal computer link functions.

■ Transmission Rate and Data Format

You cannot set up the transmission rate and data format separately because they are shared by WideField3 (or WideField2) and the personal computer link functions.

To define these items, use WideField3 (or WideField2)or the configuration function.

The table below shows the available combinations for transmission rate and data format.

Table 6.9 Combinations of Transmission Rate and Data Format

Mode	Transmission Rate and Data Format			
	Transmission Rate (bps)	Data Length	Parity	Stop Bits
Communication mode 0	9600	8 bits	Even	1 bit
Communication mode 1	9600	8 bits	None	1 bit
Communication mode 2	19200	8 bits	Even	1 bit
Communication mode 3	19200	8 bits	None	1 bit
Communication mode 4	38400	8 bits	Even	1 bit
Communication mode 5	38400	8 bits	None	1 bit
Communication mode 6	57600	8 bits	Even	1 bit
Communication mode 7	57600	8 bits	None	1 bit
Communication mode 8	115200	8 bits	Even	1 bit
Communication mode 9	115200	8 bits	None	1 bit

The personal computer link functions are set to “communication mode 0” when the sequence CPU module is shipped from the factory, the CPU memory is cleared, or the functions are not configured.

CAUTION

- **Be careful when setting the transmission rate.**

  WideField3 (or WideField2) supports all of the communication modes listed above. However, you should first refer to the user’s manual of the personal computer that runs WideField3 (or WideField2) to check available transmission rates and data formats. Then, temporarily change the transmission rate of the personal computer link functions using WideField3 (or WideField2) to make sure the Sequence CPU module can communicate with the personal computer in the communication mode you want to use. Finally, configure the personal computer link functions according to that communication mode. The personal computer link functions automatically reverts to the previous transmission rate if communication is not established after a temporary change in the transmission rate.

  If you configure the personal computer link functions using a communication mode not supported by the personal computer, all communications with the sequence CPU module will fail. If this happens, install the sequence CPU module in the fifth or higher slot of the main unit, turn on the power, make sure the RDY indicator has come on, and then turn off the power. This clears the sequence CPU module memory completely, and reverts the CPU module to its factory settings.

CAUTION

- To use the personal computer link functions, enable it using the configuration item “Use personal computer link." If you do not select this option, communication with higher-level equipment may fail.

■ Checksum, End Character and Protection

Set up these items using the configuration item “communications setup.” By default, all these items are disabled.

SEE ALSO

For details on the configuration function, see, "FA-M3 Programming Tool WideField3" (IM 34M06Q16-□□E) or "FA-M3 Programming Tool WideField2" (IM 34M06Q15-01E).

6.11.5 Communication Procedure

To be able to perform communication, the transmission specifications, including the transmission rate and data format, must be consistent between the CPU module and a personal computer, FA computer or monitor.

Use the configuration function to set up the transmission specifications of the sequence CPU module. To set the transmission specifications of a personal computer or FA computer, use a communication software program. To set the transmission specifications of a monitor, follow its configuration procedure.

■ Communication Procedure

How to communicate with the FA-M3 using a BASIC program on a personal computer is briefly described below.

For details on the statements and functions to be used in the program, refer to the BASIC reference manual that is shipped with your personal computer.

1. Open the RS-232-C communication file by entering a command using the following syntax:

   OPEN “COM : ○○○○○” AS#△

   ○○○○○ : Enter communication parameters, such as the parity, data length, and the number of stop bits.

   △ : File number. This number is used for subsequent input to and output from the file.

2. Send a command to the FA-M3 using the following syntax.

   PRINT#△, String variable name (or string)

3. To receive a response from the FA-M3, enter a command using the following syntax:

   LINE INPUT#△, String variable name

   INPUT#△, String variable name

■ Overview of Communication

Communication control performed by the CPU module is based on the processing of commands and responses using a dedicated protocol.

At first, the higher-level computer (or monitor) has the transmission right. When the computer sends a command, the transmission right transfers to the CPU module. The CPU module then sends a response to the higher-level computer.

If the configuration item “Use personal computer link function” is enabled, the CPU module does not send any command to the higher-level computer.

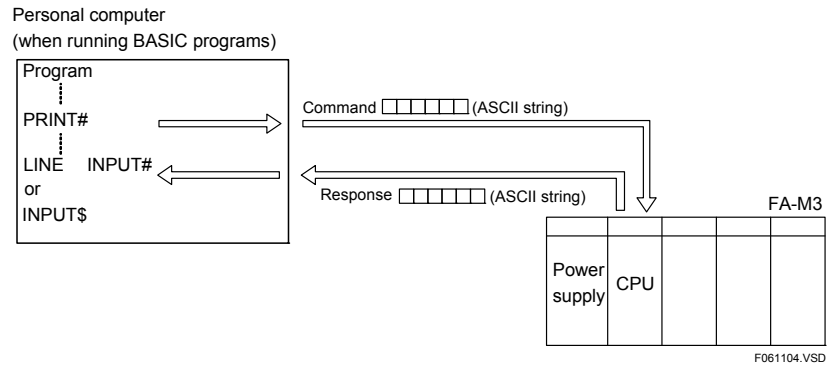


Figure 6.27 Interaction between Command and Response

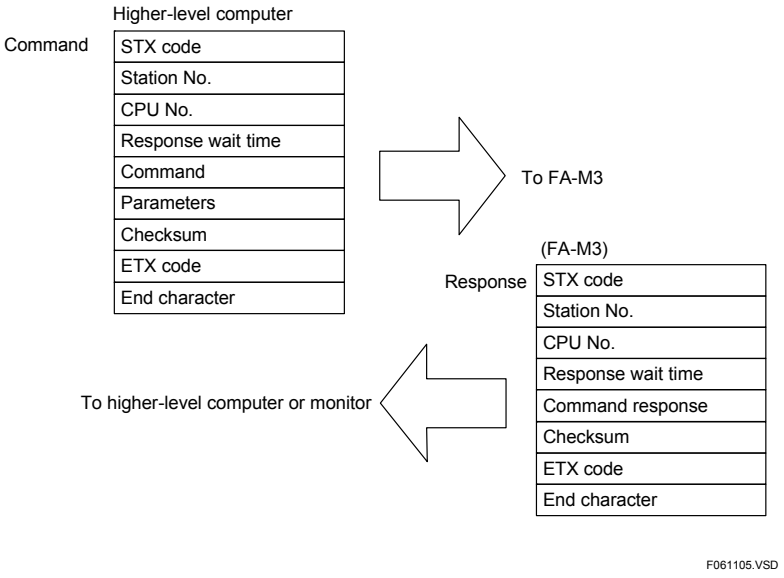


Figure 6.28 Brief Description of Command and Response Formats

6.11.6 Commands and Responses

SEE ALSO

For details on commands and responses, see "Personal Computer Link Commands" (IM 34M06P41-01E).

■ Command Format and Elements

The figure below shows the format of a command to be sent from the higher-level computer or monitor to the FA-M3.

No. of Bytes	Element	
1	STX code	
2	Station No.	
2	CPU No.	
1	Response wait time	
3	Command	
Variable-length	Parameters	
2	Checksum	Required only if the configuration item "Checksum" is set to "Yes"
1	ETX code	
1	CR code	Required only if the configuration item "End character" is set to "Yes"

F061106.VSD

Figure 6.29 Command Format and Elements

Only uppercase alphabetic characters from A to Z (ASCII codes $41 to $5A in hexadecimal) are used in commands and responses.

The individual elements are detailed below.

● STX (Start of Text) Code

This control code identifies the beginning of text. The corresponding character code is $02.

● Station No.

The station No. is fixed at 01 when the personal computer link functions of the sequence CPU module are used.

● CPU No.

Identifies the target sequence CPU module or add-on CPU module for a command using a number from 01 to 04.

01: Sequence CPU module

02: Add-on CPU module 1

03: Add-on CPU module 2

04: Add-on CPU module 3

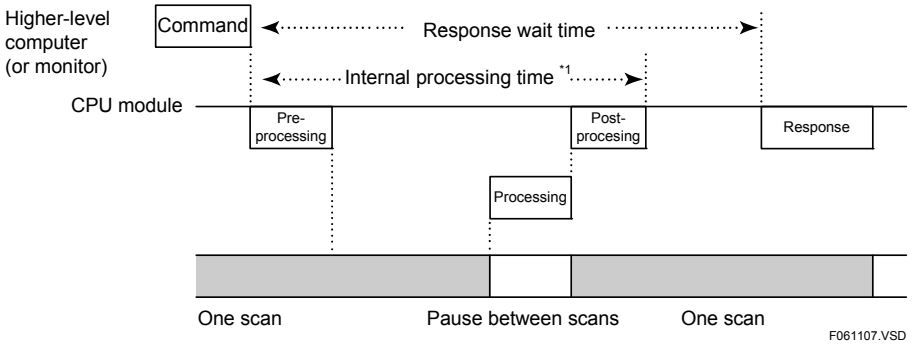
● Response Wait Time

You can specify the maximum waiting time (time delay of up to 600 ms) for a response following a command transmission.

Set a longer wait time if the communication software running on the higher-level computer is, say, a BASIC interpreter. Specify this time using one character ('0' to 'F') as shown below.

Table 6.10 Response Wait Time

Character	Response Wait Time (ms)	Character	Response Wait Time (ms)
0	0	8	80
1	10	9	90
2	20	A	100
3	30	B	200
4	40	C	300
5	50	D	400
6	60	E	500
7	70	F	600



*1: Even if the response wait time is set at 0, there is a delay of as much as the internal processing time.

Figure 6.30 CPU Operation during Response Wait Time

● Command

Using three letters, specify the type of access, such as reading or writing, from a higher-level computer (or monitor) to the sequence CPU module.

● Parameters

These include device name, number of devices, data, etc. The actual parameters vary depending on the command used. Some commands require no parameters.

● Checksum

A checksum can be added to the transmission text for data validation. You can select whether to add a checksum in the configuration.

If checksum is set to "Yes", a checksum must be appended to a command before transmission from the higher-level computer (or monitor) to the FA-M3. Moreover, a checksum is automatically appended to the response transmitted from FA-M3.

If checksum is set to "No", this element must not be appended to a command.

How the checksum is calculated is explained below.

- Add the ASCII codes of the characters following the STX character and preceding the checksum.
- Extract the low order byte of the sum and express its hexadecimal value as a character string (2 characters, 2 bytes) to obtain the checksum.

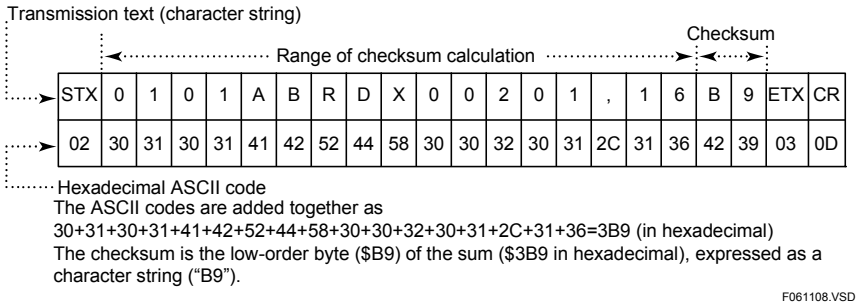


Figure 6.31 Checksum Calculation

● ETX (End of Text) Code

A control code indicating the end of text. The corresponding character code is $3.

● CR (Carriage Return) Code

A control code indicating the termination of text. The corresponding character code is $0D, which is the ASCII-code decimal numeral of 13. This code is required only if the configuration item “End character” is set to “Yes.”

■ Response Format and Elements

The format of a response that is sent from the FA-M3 to a higher-level computer (or monitor) is shown here. For details on individual elements and characters used, see "■ Command Format and its Elements" given earlier in this section.

● When Communication is Normal

No. of Bytes	Element
1	STX code
2	Station No.
2	CPU No.
2	OK
Variable-length	Command response
2	Checksum
1	ETX code
1	CR code

Appended to the response only if enabled accordingly in the configuration.

F061109.VSD

Figure 6.32 Response Format when Communication is Normal

When communication ends successfully, the string “OK” is returned along with a command response.

● When Communication is Abnormal

No. of Bytes	Element
1	STX code
2	Station No.
2	CPU No.
2	ER
2	EC 1
2	EC 2
3	Command
2	Checksum
1	ETX code
1	CR code

Appended to the response only enabled accordingly in the configuration.

F061110.VSD

Figure 6.33 Response Format when Communication Is Abnormal

When communication results in an abnormal end, the string “ER” is returned along with the codes EC1 and EC2.

EC1 = Error code

EC2 = Detailed error code

If the communication failure is due to an error in the CPU number, the received 2-byte CPU number is returned. If the failure is due to an error in the station number, no response is returned.

If an ETX code in a command is not received, no response may be returned. If this happens, be sure to perform a timeout process on the higher-level computer or monitor.

■ Error Code in a Response

A communication failure may occur when the sequence CPU module receives a command. In that case, the module returns the string “ER” and an error code as a response to the command.

The table below shows the error codes that may be returned as a response.

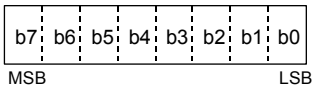
Table 6.11 Error Codes in a Response

Error Code (EC1)	Semantics	Possible Causes
01	CPU number error	- The CPU number is outside the range of 1 to 4.
02	Command error	- The command does not exist. - The command is not executable.
03	Device specification error	- The device name does not exist. * - A relay device is incorrectly specified for read/write access in word units.
04	Value outside the setting range	- Characters other than 0 and 1 are used for bit setting. * - Word setting is out of the valid range of 0000 to FFFF. - The specified starting position in a command, such as Load/Save, is out of the valid address range.
05	Data count out of range	- The specified bit count, word count, etc. exceeded the specifications range. * - The specified data count and the device parameter count, etc. do not match.
06	Monitor error	- Attempted to execute monitoring without having specified a monitor command (BRS, WRS).
08	Parameter error	- A parameter is invalid for a reason other than those given above. *
41	Communication error	- An error has occurred during communication. *
42	Checksum error	- Value of checksum differs. (Bit omitted or changed characters)
43	Internal buffer overflow	- The amount of data received exceeded stipulated value.
51	Timeout error	- No end-of-process response is returned from the CPU for reasons such as CPU power failure. (timeout)
52	CPU processing error	- The CPU has detected an error during processing. *
F1	Internal error	- A Cancel (PLC) command was issued during execution of a command other than a Load (PLD) or Save (PSV) command. - An internal error was detected.

*: See Table 6.14, "Detailed Error Codes," for more information.

If a parameter error occurs, the detailed error code field indicates the number of the faulty parameter. If a communication error occurs, the detailed error code field indicates details on the error.

Table 6.12 Detailed Error Codes

Error Code (EC1)	Meaning	Detailed Error Code (EC2) *
03	Device specification error	The EC2 field provides a hexadecimal representation of the number assigned to the faulty parameter. (The number is one, among the ordinal parameter numbers, at which an error has occurred first.)
04	Value outside the setting range	
05	Data count out of range	
08	Parameter error	(Example:) S T 0101ABRW 03 Y00501, 1, I0002, 0, I I0012, 1 X 1 2 3 4 5 6 7 ← Parameter numbers Erroneous device number In this example, the respective error codes take the values shown below. - EC1 = 03 - EC2 = 06
41	Communication error	 Each bit has the following meaning. b7: Reserved b6: Reserved b5: Framing error b4: Overrun error b3: Parity error b2: Reserved b1: Reserved b0: Reserved
52	CPU processing error	1☐: Self-diagnostic error 2☐: Program error (including parameter error) 4☐: Inter-CPU communication error 8☐: Device access error 9☐: Communication protocol error A☐: Parameter error B☐: Operating mode error, protected/exclusive access C☐: Device/block specification error F☐: Internal system error

* The EC2 error code has no meaning for any value of EC1 other than those listed above.

■ List of Supported Devices

Use commas (,) or spaces to separate parameters. A device name should be represented in six or seven characters (or bytes). Their abbreviations can be also used, however. For example, X00201 can be abbreviated as X201 and V00002 as V02 or V2.

The following example shows a case when you read the data of CPU1's five input relays, beginning with input relay X00201. The response wait time is assumed to be 100 ms.

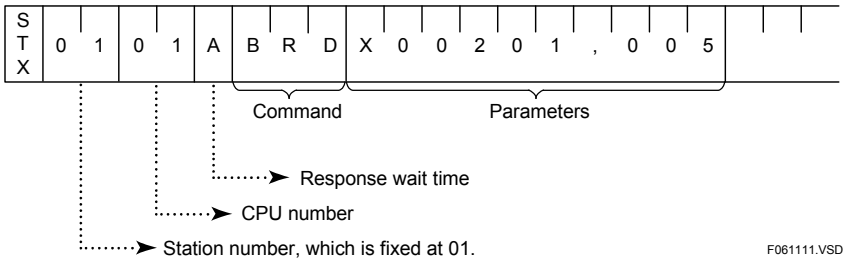


Table 6.13 List of Supported Devices

Device Name		Length	Read		Write	
			Bit	Word	Bit	Word
Bit device	Xnnnnn Input relay	6 bytes	Yes	Yes	No	No
	Ynnnnn Output relay	6 bytes	Yes	Yes	Yes	Yes
	Innnnn Internal relay	6 bytes	Yes	Yes	Yes	Yes
	Ennnnn Shared/extended shared relay	6 bytes	Yes	Yes	Yes	Yes
	Lnnnnn Link relay	6 bytes	Yes	Yes	Yes	Yes
	Mnnnnn Special relay	6 bytes	Yes	Yes	Yes*6	Yes*6
	Tnnnnn Timer	6 bytes	Yes*1	Yes*2	No	Yes*2
	Cnnnnn Counter	6 bytes	Yes*1	Yes*2	No	Yes*2
Word device	Dnnnnn Data register	6 bytes	No	Yes	No	Yes
	Rnnnnn Shared register	6 bytes	No	Yes	No	Yes
	Vnnnnn Index register	6 bytes	No	Yes	No	Yes
	Bnnnnnn File register *3	7 bytes	No	Yes	No	Yes
	Wnnnnn Link register	6 bytes	No	Yes	No	Yes
	Znnnnn Special register	6 bytes	No	Yes	No	Yes*6

*1 Specify:
a time-out relay as TUnnnn, and
an end-of-count relay as CUnnnn,

*2 Specify:
- the current value of a countdown timer as TPnnnn,
- the current value of a countdown counter as CPnnnn,
- the current value of a count-up timer*4 as TInnnn,
- the current value of a count-up counter*4 as CInnnn,
- the setpoint of a timer*5 as TSnnnn, and
- the setpoint of a counter*5 as CSnnnn.

*3 Only available with the F3SP22, F3SP25, F3SP28, F3SP35, F3SP38, F3SP53, F3SP58, and F3SP59 sequence CPU modules.

*4 The countdown type of timers and counters has been made available with the FA-M3 controller for such reasons as viewing them on a higher-level computer.
Current value of count-up type timer/counter =Setpoint-Current value of countdown type timer/counter

*5 The timer setpoint TSnnnn and counter setpoint CSnnnn are not available for a word writing command.

*6 In the case of F3SP22, F3SP28, F3SP38, F3SP53, F3SP58, and F3SP59 sequence CPU modules, writing to this device is not possible with any of BWR, BFL, WWR and WFL commands. Alternatively, use a BRW or WRW command.

■ Precautions for Communications

- You should include timeout handling on the higher-level computer to handle situations where a response is not returned due to say, an incorrect station number specified in the command.
- If the personal computer link functions are used to download a program, then you should not load another program from another source (personal computer link module, Ethernet interface module, etc.) at the same time. Otherwise, normal operation is not guaranteed.
- When writing to a shared device, the value may be immediately overwritten if another sequence CPU module is using the same device.
- If a power failure occurs when a monitor command is in use, it is necessary to set it again.
- The maximum text length that can be transmitted or received each time by the personal computer link functions is 512 bytes. However, the maximum size that can be received by a higher-level computer may be limited to 256 bytes in some cases. In such cases, make sure that the response text length does not exceed 256 bytes by reducing the number of devices to be read.

6.12 Device Management Functions

The device management functions enable you to upload, download, edit and compare device information/data of the sequence CPU module using WideField3 (or WideField2). You can specify the range of device data to be uploaded or downloaded.

You can also use these functions to perform initial setup of device data when, for example, replacing the CPU module.

The devices that you can configure using the device management functions are:

Internal relays (I), shared relays (E), time-out relays and current values of timers (T), end-of-count relays and current values of timers (C), data registers (D), shared registers (R), link registers (W), index registers (V) and file registers (B).

You cannot configure the following devices:

I/O relays (X/Y), setpoints of timers (T) and counters (C), special relays (M) and special registers (Z).

The device management function serves the following four purposes.

■ Upload Device Data

This function allows you to read device information/data from the sequence CPU module and saves it to a WideField3 (or WideField2) file. You can specify the range of devices to be saved.

■ Download Device Data

This function allows you to read device information/data from a WideField3 (or WideField2) file and writes it to the sequence CPU module. You can either download all device data from the file or download part of the data by specifying a range of devices.

■ Edit Device Data

This function allows you to edit device information/data in a WideField3 (or WideField2) file. You can view and change the current value of each device.

■ Compare Device Data

This function allows you to compare device information/data in the sequence CPU module with that in a WideField3 (or WideField2) file. You can make a comparison of all device data in the file or part of the data by specifying a range of devices. If any mismatch is found, the function shows the device name and content of the mismatch.

6.13 Macro Instructions

This section describes macro instructions.

Macro instructions allow reuse of created programs for increased programming efficiency. In addition, the use of macro instructions allows compact program codes structured by function, thus improving program readability and maintainability.

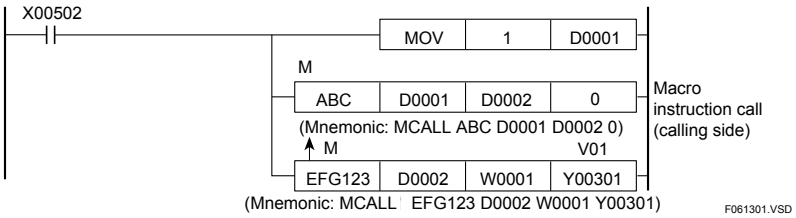
6.13.1 What Are Macro Instructions?

■ Overview

A macro instruction enables a process requiring multiple instructions/steps to be processed as a single instruction.

The figure below presents an overview of macro instructions.

How to code a macro instruction in ladder diagram editing



How to code a macro instruction entity "ABC" in ladder macro editing

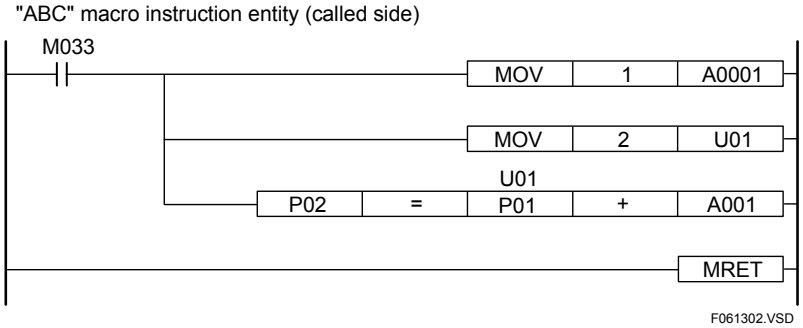


Figure 6.34 Examples of Macro Instructions

In the above figure, “ABC” and “EFG123” instructions are macro instructions. When the CPU encounters the “ABC” instruction, it executes the “ABC” macro instruction entity like a subroutine, using “D0001” and “D0002” as parameters. Macro instructions are created using ladder macro editing, separately from normal instructions created using ladder diagram editing.

The Macro Return (MRET) instruction represents the end of a macro instruction entity.

For details on parameters P01, P02, and U01 in the figure, see Subsection 6.13.3, "Devices Dedicated to Macro Instructions.”

SEE ALSO

For details on the Macro Return (MRET) instruction, see Section 3.13.4, "Macro Call (MCALL), Parameter (PARA), Macro Return (MRET)” of "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

■ Purpose

Using macro instructions offers the following two advantages.

● Increased Programming Efficiency

Like subroutines, macro instructions allow grouping of similar processing. Macro instructions differ from subroutines, however, on the following two points.

- **Parameters can be passed to macro instructions.**
  Subroutines require the use of instructions for passing parameters (e.g. MOV instructions) preceding a CALL instruction.
- **Macros can be handled as instructions.**
  A user need not be aware of the internal processing of a macro, except for its input and output parameters.

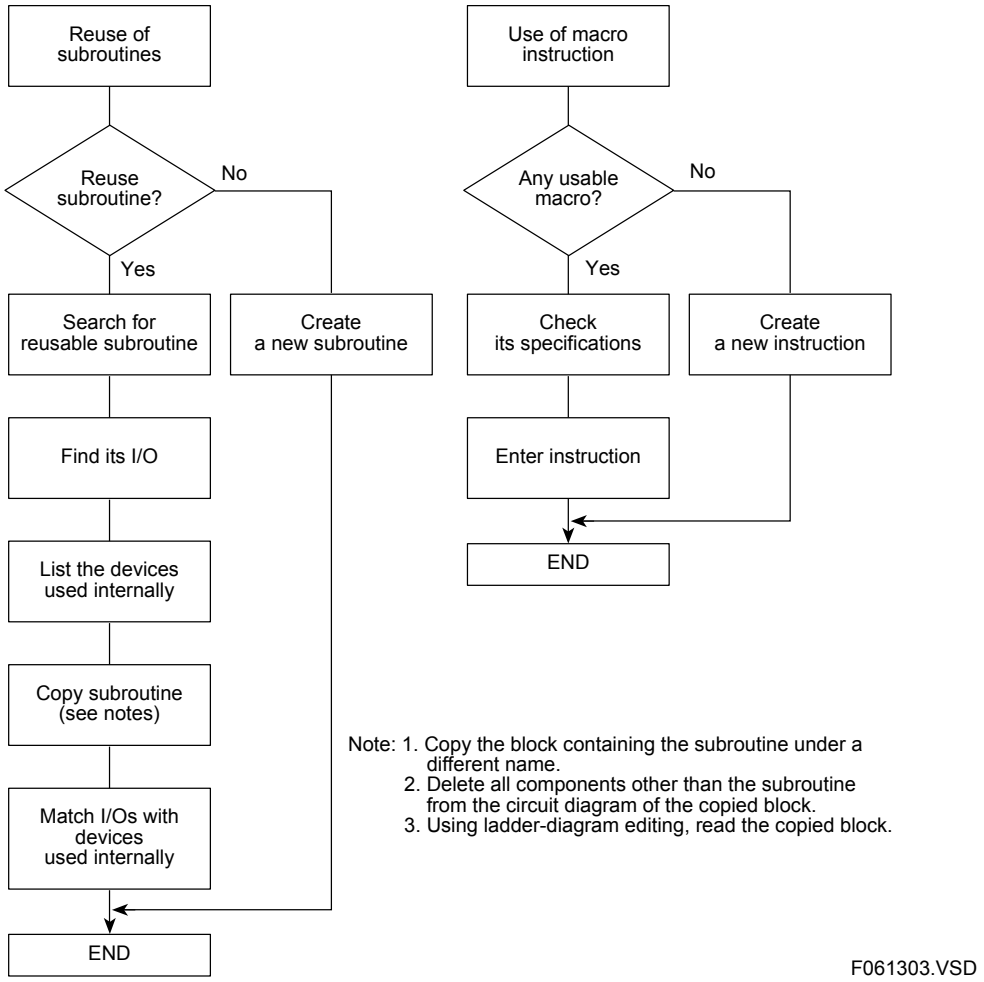


Figure 6.35 Differences between Subroutines and Macro Instructions

● Accumulation of Know-how

Know-how can be accumulated in the form of macro instructions for creation of customized FA-M3 controllers.

■ Types of Macro Instructions

There are three types of macro instructions, the availability of which depends on CPU types as follows:

Table 6.14 Macro Instructions and their Availability by CPU Type

Macro Instruction Type	F3SP28-3N F3SP38-6N F3SP53-4H F3SP58-6H	F3SP22-0S F3SP28-3S F3SP38-6S F3SP53-4S F3SP58-6S F3SP59-7S
Macro Call (MCALL)	Available	Available
Input Macro Instruction Call (NCALL)	Not available	Available
Structure Macro Instruction Call (SCALL)	Not available	Available

● Macro Call

Up to 16 parameters can be passed via a Macro Call instruction.

● Input Macro Instruction Call

The Input Macro Instruction Call instruction can be used as an input condition, just like the Load or Compare instruction. It can be used to represent complex or reusable input conditions in a single instruction.

Using an Output of Input Macro (NMOUT) instruction in an Input Macro Instruction call allows you to output the result of logical operations to the next instruction.

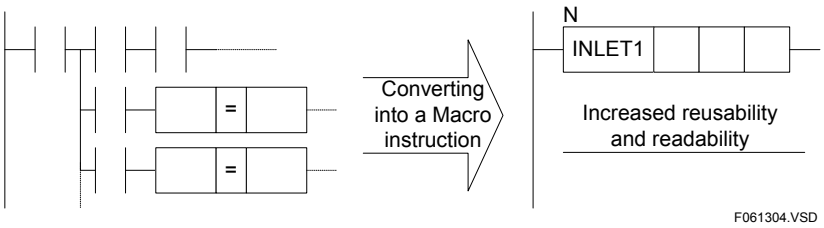


Figure 6.36 Benefits of Input Macro Instruction Call

● Structure Macro Instruction Call

The Structure Macro Instruction Call instruction passes multiple data items collectively in a structure to a macro instruction, and is especially useful in reducing the number of items to be passed to a macro instruction and providing better representation of a group of related data items.

6.13.2 Specification of Macro Instructions

■ Number of Macro Instructions

Macro instruction entities to be called are downloaded, along with user ladder programs, from a personal computer to the sequence CPU module using WideField3 (or WideField2). The table below lists the maximum number of macro instruction entities allowed in one executable program during downloading. A macro instruction can be called any number of times in a user ladder program.

Table 6.15 Maximum Number of Macro Instructions Allowed by CPU Type

Types	F3SP28-3N F3SP38-6N F3SP53-4H F3SP58-6H	F3SP22-0S F3SP28-3S F3SP38-6S F3SP53-4S F3SP58-6S F3SP59-7S
Macro Call (MCALL)	64	256 in total
Input Macro Instruction Call (NCALL)	0 (not available)	
Structure Macro Instruction Call (SCALL)	0 (not available)	

■ Size of Macro Instruction Program

The size of a macro instruction program is limited by the total size of that program and user programs combined.

■ Macro Instruction Execution Time

Table 6.16 Macro Instruction Execution Time

FUN NO.	Instruction	Mnemonic	F3SP22 F3SP28 F3SP38		F3SP53 F3SP58 F3SP59	
			When Executed (μs)	When Not Executed (μs)	When Executed (μs)	When Not Executed (μs)
996	Macro Call	MCALL	10.0	0.45	4.0	0.175
995	Parameter	PARA	6.0	0.27	2.5	0.105
998	Macro Return	MRET	5.0	—	2.0	—
981	Input Macro Instruction Call	NCALL	10.0	0.45	4.0	0.175
309	Output of Input Macro	NMOUT	2.4	0.18	1.0	0.070
985	Structure Macro Instruction Call	SCALL	34.0	3.17	15.7	1.265

■ Online Editing of Macro Instructions

You can use the online edit functions of WideField3 (or WideField2) to edit circuits containing macro instruction calls or input macro instruction calls. However, you can only use macro instructions that are already downloaded, and cannot create any new macro instruction. Circuits containing structure macro instruction calls cannot be edited online.

You can also use the online edit functions to edit macro instruction entities already downloaded, but you cannot edit any circuits following the Macro Return (MRET) instruction.

■ Making Macro Instructions Resident in ROM

You can make macro instructions resident in a ROM pack, just like programs. This is automatically done when you transfer a program to the ROM pack using the ROM writer function of the CPU module.

6.13.3 Devices Dedicated to Macro Instructions

Table 6.17 Devices Dedicated to Macro Instructions

Device	Symbol	Range	Number of Devices
Pointer register (P)	P	P01 to P16	16
Macro relay (H)	H	H0001 to H0512	512
Macro register (A)	A	A0001 to A0512	512
Macro index register (U)	U	U01 to U16	16
Structure pointer register (Q)	Q	Q01, Q02	2

■ Pointer (P) Registers

Pointer registers are used specifically to pass parameters to macro instructions. These registers can be used within macro instruction entities. Structure macro instructions use structure pointer registers instead of pointer registers.

The relationship between pointer registers (P) and macro instruction parameters is shown in the following figure.

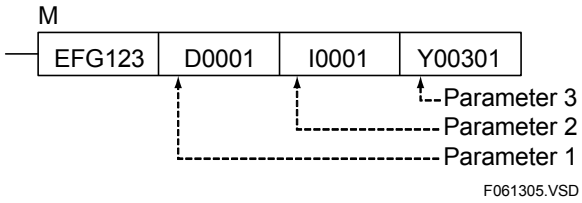


Table 6.18 Relationship between Pointer Registers and Macro Instruction Parameters

Operand No.	Pointer Register No.
1	P01
2	P02
3	P03
4	P04
⋮	⋮
16	P16

Parameters that can be directly passed using a macro instruction call

Parameters that can be passed using a parameter instruction

F061306.VSD

Within a macro instruction entity, you can read from and write to pointer registers using basic or application instructions, in the same way as for devices passed as parameters. You can also perform word/long word processing, index modification, and automatic BIN-to-BCD or BCD-to-BIN conversion on these pointer registers.

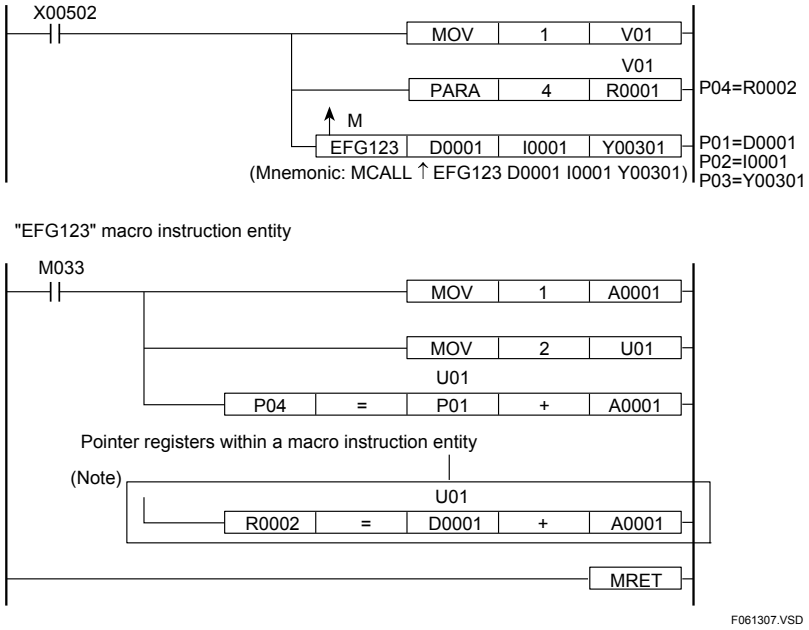
High speed processing of application instructions is not performed, however. More specifically, within a macro instruction entity, high speed processing is not performed for MOV, CAL, CMP, or logical operation instructions with pointer registers specified as parameters.

TIP

When an instruction using a pointer register (P) is to be executed repeatedly, you can first transfer the values of the pointer registers (P) to macro relays (H) and macro registers (A) and then rewrite the instruction to use these relays and registers instead. In this way, you can shorten the execution time.

SEE ALSO

For details on basic and application instructions, see Sections 2.1 and 3.1 of "Sequence CPU Instruction Manual - Instructions" (IM 34M06P12-03E).



Note: Pointer registers can be used within a macro instruction entity.

Figure 6.37 An Example Using Pointer Registers (P)

CAUTION

- If you pass a device with index modification as a parameter to a macro instruction, the instruction receives the index-modified device. In the example shown in the above figure, parameter R0001;V01 is the same as device R0002 because V01 = 1.
- Any index modification of a pointer register (P) is applied to the parameter that is passed. In the example shown in the above figure, P01;U01 is the same as device D0003 because P01 = D0001 and U01 = 2.

■ Macro Relays (H), Macro Registers (A) and Macro Index Registers (U)

These devices are dedicated to macro instructions. Within a macro instruction entity, you can read from and write to macro relays, macro registers or macro index registers using basic or application instructions, the same way as for internal relays (I), data registers (D) and index registers (V). These devices can be used within a macro instruction entity.

By using these devices in your macro entity, you need not know which devices are used in the macro instruction call. Needless to say, the values of these devices remain unchanged.

■ Structure Pointer Registers (Q)

F3SP□□-□S

Structure Pointer Registers are dedicated registers used for passing structure data to structure macro instructions. It is used within structure macro instruction entities.

The relationship between structure pointer registers (Q) and structure macro instruction parameters is shown in the figure below.

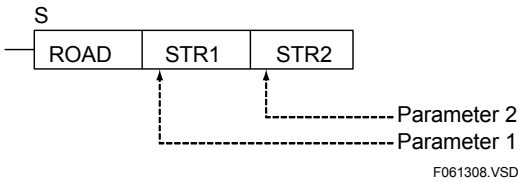


Table 6.19 Relationship between a Structure Pointer Register (Q) and Structure Macro Instruction Parameters

Operand	Structure pointer register number
1 (parameter 1)	Q01
2 (parameter 2)	Q02

Within a structure macro instruction entity, you can read from and write to structure data passed as parameters using basic or application instructions and referring to structure members using the "<structure pointer register number>.<structure member name>" syntax.

Word processing, long-word processing and automatic BIN-to-BCD or BCD-to-BIN conversion can be used with structure pointer registers, but index modification is not allowed.

High speed processing of application instructions is not performed, however. More specifically, within a structure macro instruction entity, high speed processing is not performed for MOV, CAL, CMP, or logical operation instructions with structure pointer registers (Q) specified as parameters.

TIP

When an instruction using a structure pointer register (Q) is to be executed repeatedly, you can first transfer the member data to macro relays (H) and macro registers (A) and then rewrite the instruction to use these relays and registers instead. In this way, you can shorten the execution time.

SEE ALSO

For details on basic and application instructions, see Sections 2.1 and 3.1 of the "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

SEE ALSO

For details on structures, see "FA-M3 Programming Tool WideField3" (IM 34M06Q16-□□E) or "FA-M3 Programming Tool WideField2" (IM 34M06Q15-01E).

6.13.4 Nesting Macro Instructions

Nesting macro instructions is to call another macro instruction or an input macro instruction when executing a macro instruction.

Calling a structure macro instruction within a structure macro instruction body is not allowed.

Calling another macro instruction or an input macro instruction from a structure macro instruction body is allowed but the PARA instruction cannot be used.

Nesting macro instruction calls beyond seven levels will cause an instruction processing error. The nesting depth is stored in special register Z106. A value of “0” is stored in the special register Z106 during non-nested execution of a macro instruction.

Table 6.20 Calls between Macros, Input Macros, and Structure Macros

Calling Side	Called Side	Availability
Block	Macro	✓
Block	Input macro	✓
Block	Structure macro	✓
Macro	Macro	△
Macro	Input macro	△
Macro	Structure macro	x
Input macro	Macro	△
Input macro	Input macro	△
Input macro	Structure macro	x
Structure macro	Macro	△
Structure macro	Input macro	△
Structure macro	Structure macro	x

✓ : Call is allowed (PARA instruction can be used).
△: Parameters passed using the PARA instruction are overwritten.
X : Call is not allowed.

CAUTION

Parameters 1 to 3 passed to macro instructions are saved when macro instructions are nested. However, parameters 4 to 16 passed using PARA (parameter) instructions are not saved. If a Parameter (PARA) instruction is executed in a called macro instruction, the relevant parameters are overwritten.

CAUTION

Errors generated in nested macro instructions are reported as errors of the first macro instruction.

SEE ALSO

For details on the Parameter (PARA) instruction, see Section 3.13.4, "Macro Call (MCALL), Parameter (PARA), Macro Return (MRET)” of "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

When nesting macro instructions, you may mistakenly overwrite macro devices, such as relays, registers and index registers, in a called macro instruction and thereby destroy their data. To avoid this problem, check the depth of macro instruction nesting stored in special register Z106 and use macro devices separately for each level of nesting depth (see the example below).

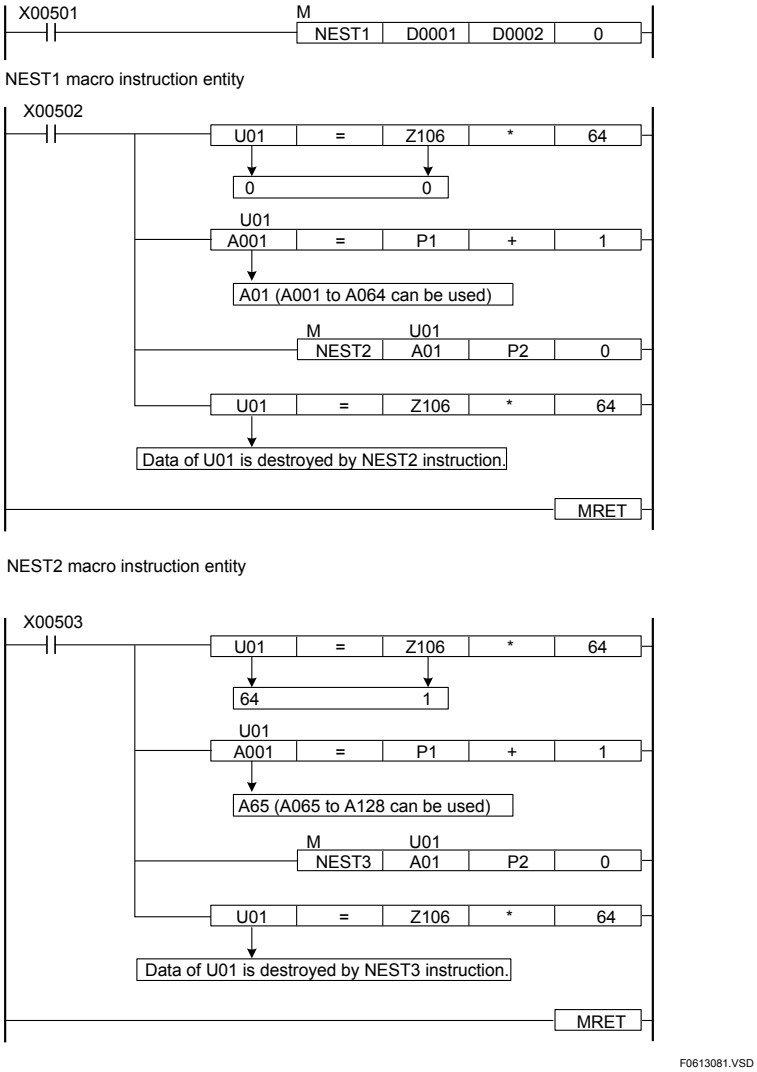


Figure 6.38 Example of Macro Device Separation when Nesting Macro Instructions

6.13.5 Handling Macro Instruction Errors

When creating a program using a macro instruction tool, an error is generated if:

- There are two or more macro instructions of the same name
- A macro instruction specified in a macro call (MCALL) is not found
- A macro instruction entity contains two or more macro return (MRET) instructions

An error is also generated and the special relay M201 for instruction processing errors is set to ON if:

- A macro return (MRET) instruction is executed before a macro call (MCALL) (special register Z022 contains the error code $2501)
- The depth of macro call nesting exceeds 7 levels (special register Z022 contains the error code $2502)

An error detected within a macro instruction entity is seen by the user as an error of the macro instruction. Thus, the user can know which parameters were passed to the macro instruction.

CAUTION

Any error detected by self-diagnosis (except for a memory checksum error) within a macro instruction entity is also seen by the user as an error of the macro instruction execution.

Table 6.21 Error Codes for Macro Instructions

Error Type	Error Name	Error Code	Description
Instruction processing	Macro instruction error	$2501	There is no return destination.
		$2502	The maximum nesting depth (seven levels) is exceeded.

6.13.6 Protecting Macro Instructions

You can protect macro instructions against unauthorized read access. The protection can be configured on per macro instruction basis by entering a password using WideField3 (or WideField2). A password must consist of eight alphanumeric characters, beginning with a letter. The protection information is saved in the management information area of a macro instruction file. A protected macro instruction can be edited, printed or monitored only if the password matches.

TIP

Executable program protection and block protection also apply to user-created ladder programs containing macro instructions. For instance, if executable program protection is enabled, downloading, uploading, monitoring, online-editing and other operations on the executable program are not allowed.

6.13.7 Debugging Operation

■ Forced Set and Forced Reset

You can force bit devices to turn ON or turn OFF in macro instructions (either in a macro call or within a macro instruction entity).

■ Partial Operation

Partial operation is not allowed within a macro instruction entity.

6.13.8 Input Macro Instructions

F3SP□□-□S

An Input Macro Instruction is a type of macro instruction that can be used as an input condition, just like the Load or Compare instruction. It can represent complex, reusable input conditions as a single instruction.

By calling the Output of Input Macro (NMOUT) instruction internally, an input macro instruction can also output the result of logical operation to the next instruction.

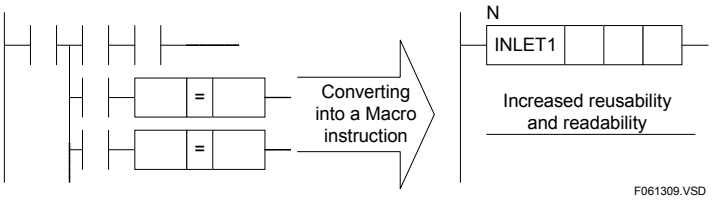


Figure 6.39 Benefits of Input Macro Instructions

■ How to Use

● Creating an Input Macro Instruction

Input macro instructions can be created like ordinary macro instructions.

Macro instructions called by the Input Macro Instruction Call (NCALL) instruction are called input macro instructions.

Thus, the same macro instruction entity can be either an input macro instruction (if called by NCALL) or a macro instruction (if called by MCALL).

● Calling an Input Macro Instruction

Use the Input Macro Instruction Call (NCALL) instruction to call an input macro instruction.

SEE ALSO

For details on the NCALL instruction, see "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

● Where to Code an Input Macro Instruction Call (NCALL) Instruction

You can code an NCALL instruction along with the **LOAD**, **AND**, or **OR** logical operator.

You cannot use it in place of an output instruction (at the right end of a ladder rung).

To call a macro instruction at the position of an output instruction, use the MCALL instruction instead.

● Passing Parameters to an Input Macro Instruction

Use the pointer register (P) to pass parameters to an input macro instruction, the same way as with ordinary macro instructions.

Up to three parameters can be directly coded in an NCALL instruction. To pass more than three parameters, use the Parameter (PARA) instruction. Be careful when using the PARA instruction, because it can be used by both macro and input macro instructions.

SEE ALSO

For details on pointer registers (P), see Section 6.13.3, "Devices Dedicated to Macro Instructions."

● Output of Logical Operation Result to the Power Rail

The NMOUT instruction is used to specify the logical operation result of an input macro instruction. The logical operation result to be output to the step following the Input Macro Instruction Call instruction depends on the status of the input parameter type of the NMOUT instruction.

Input Parameter	Logical Operation Output of Input Macro (device status = output)
Constant	OFF if 0, ON if otherwise
Relay device	OFF if 0, ON If 1
Register device	OFF if 0, ON if otherwise

If the NMOUT instruction is executed more than once, the last instruction takes precedence.

If no NMOUT instruction is executed, the logical operation result of an input macro is OFF.

SEE ALSO

For details on the NMOUT instruction, see "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

CAUTION

The NMOUT instruction takes effect only if executed within a macro instruction that has been called by NCALL (that is, an input macro). It is ignored if executed within a macro instruction that has been called by MCALL.

● Nesting Input Macros

Macro and input macro instructions when combined may be nested up to 8 levels.

6.13.9 Structure Macro Instructions

F3SP□□-□S

The Structure Macro Instruction passes a number of parameters collectively in a structure to a macro instruction. By using a structure, it simplifies data passing and improves representation in cases where there are many related parameters.

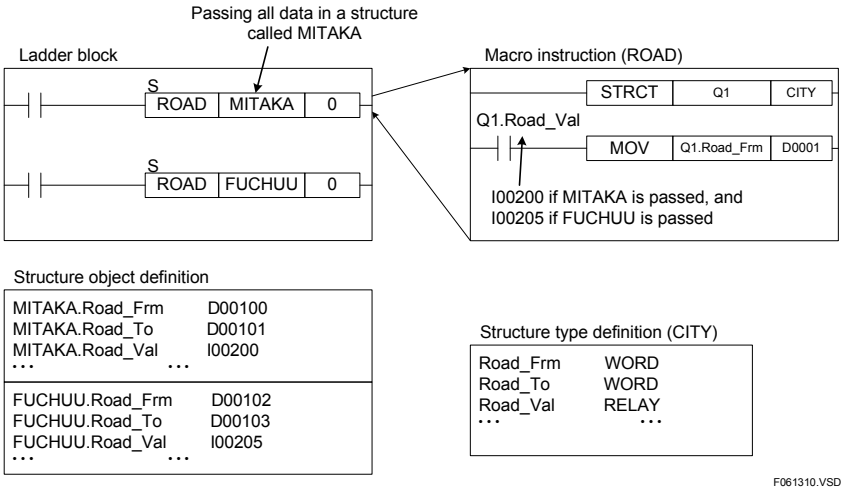


Figure 6.40 Benefits of Structure Macro Instructions

SEE ALSO

For details on structures, see "FA-M3 Programming Tool WideField3" (IM 34M06Q16-□□E) or "FA-M3 Programming Tool WideField2" (IM 34M06Q15-01E).

CAUTION

A structure macro instruction may not call another structure macro instruction.
A structure macro instruction may be called only by a block.

CAUTION

If the type of a structure passed using a structure macro instruction is different from the structure type declared by a structure pointer declaration instruction in the called structure macro instruction, the latter structure type is used during execution with no error generated.

CAUTION

Structure macros use pointer registers P4 to P8.

■ How to Use

● Structure Type Definition

Defines the name and members of a structure type.

● Structure Object Definition

Allocates actual registers to structure data.

● Creating Structure Macro Instructions

Structure macro instructions can be created just like ordinary macro instructions.

Macro instructions called by the structure Macro Instruction Call (SCALL) instruction are called structure macro instructions.

● Structure Type Declaration (STRCT) for Structure Macro Instructions

At the very beginning of a structure macro instruction, you must declare the type of the structure to be passed.

One structure type declaration is required if one structure is to be passed. Two structure type declarations are required if two structures are to be passed.

● Calling a Structure Macro Instruction

Use the structure Macro Instruction Call (SCALL) instruction to call a structure macro instruction.

SEE ALSO

For details on the SCALL instruction, see "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

● Passing Structure Data to a Structure Macro Instruction

When passing structure data, code the name of the structure in the macro instruction call, but use a structure pointer register (Q) within the macro instruction entity.

SEE ALSO

For details on structure pointer registers (Q), see Section 6.13.3, "Devices Dedicated to Macro Instructions."

● Nesting structure Macros

A structure macro instruction cannot call another structure macro instruction.

A structure macro instruction can call macro and input macro instructions, but cannot use the PARA instruction.

6.14 User Log Management Functions

The user log management functions keep a record of error events in a user system, including information on error occurrence and system operation status, when Save User Log instructions are executed. Stored user log records can then be read using instructions or WideField3 (or WideField2). These functions are useful for analyzing faults and understanding the operating conditions of machinery.

■ Handling User Logs

A maximum of 64 user log records per CPU can be saved by executing user log instructions in a program.
Four data items, namely, date of occurrence, time of occurrence, main code (one word) and subcode (one word), are saved in each user log record.
Up to sixty-four 32-character messages associated with individual main codes can be stored in the CPU. These stored messages can be retrieved along with main codes when log information is read.
The user log information area is maintained as a rotary buffer. If the maximum number of log records allowed is exceeded, existing log records are overwritten by new log records in chronological order.
Stored user log records can be read using WideField3 (or WideField2) or Read User Log instructions. You can check the special register Z105 for the number of stored user log records available.

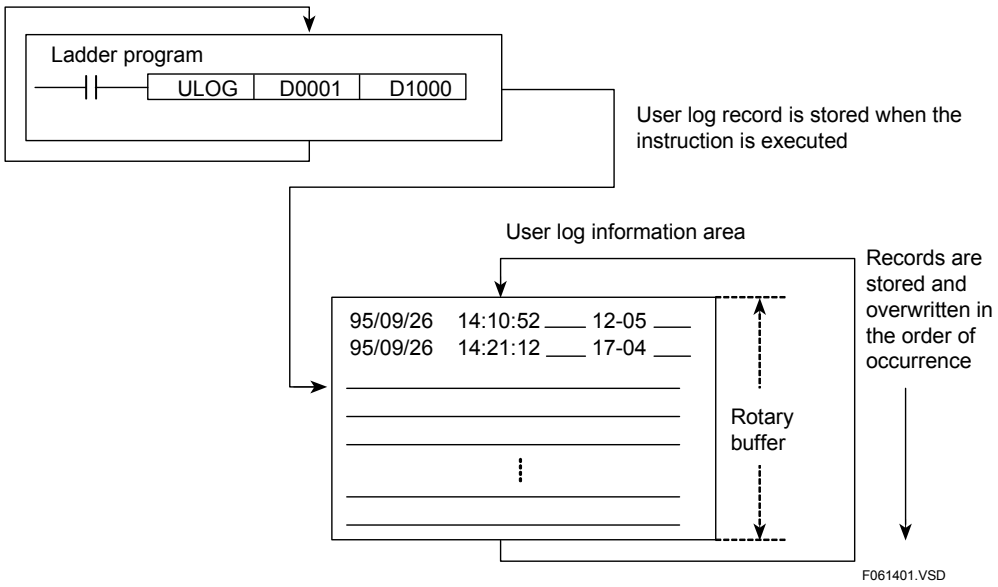


Figure 6.41 Handling User Logs

CAUTION

In some cases, WideField3 (or WideField2) may display two identical log records. This happens if a save user log instruction is executed when stored user log records is read using WideField3 (or WideField2). To solve the problem, redisplay log records when not executing a save user log instruction.

SEE ALSO

For details on the instructions related to user log, see Subsection 3.22.5, "Save User Log (ULOG), Read User Log (ULOGR) and Clear User Log (UCLR)" of "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

TIP

- You may assign any values from -32768 to +32767 to user log main codes. Messages for main codes 1 to 64 can be stored in a CPU module.
- You may also assign any values from -32768 to +32767 to user log sub codes.

6.15 Sensor Control Functions

To enable request responses at speeds of several hundred microseconds, a small PLC or sensor controller is often installed alongside the main PLC. The sensor control functions play the role of such a controller, enabling a program to be scanned at high speeds and fixed intervals independently and not affected by the main scan time, which tends to lengthen due to advanced functionality or high performance of the system.

Using the sensor control functions, you can execute one program block at high speeds and fixed intervals (200 µs minimum), independent of the regular scan. These functions are useful for control applications requiring higher machining accuracy.

6.15.1 Schematic Operation Diagram

The figure below shows the operation of a sensor control block.

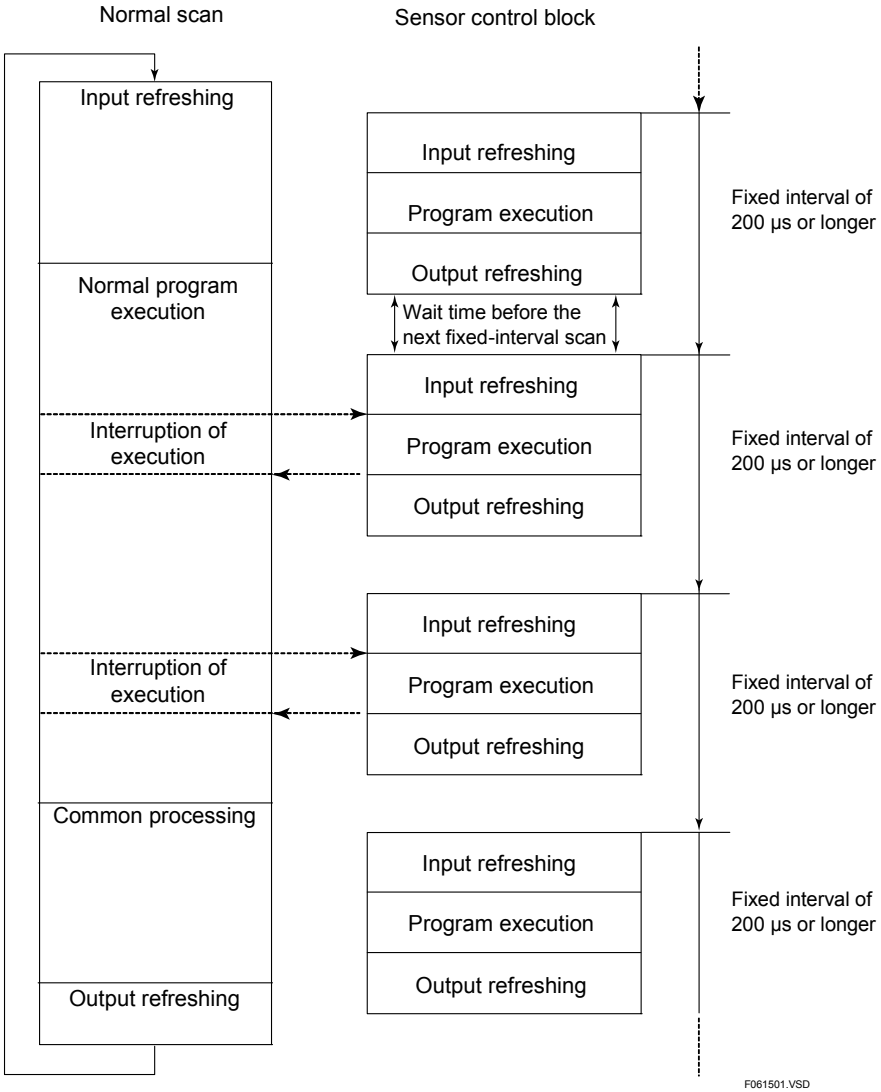


Figure 6.42 Schematic Diagram of Sensor Control Block Operation

6.15.2 Features

■ Features

The sensor control functions have the following features.

● High Speed

- The minimum interval of block execution is as short as 200 μs.
- The sequence CPU module operates as if it contains another sequence CPU module with the minimum scan time of 200 μs.
- The maximum I/O response delay is only 400 μs, i.e., twice the minimum interval of block execution.
- You can use it for a process requiring fast I/O response by isolating the process from the regular program.
- It allows the use of a wide choice of modules, including special modules for input/output.

● Fixed Interval

- The sensor control block is executed at fixed intervals.
- The sensor control block function runs even during instruction execution, refreshing or common processing of the normal scan.

6.15.3 Specifications and Restrictions

■ Specifications

Table 6.22 Specifications of Sensor Control Block

Item	Specifications
Number of sensor control blocks	1
Execution interval	200 μs to 25.0 ms, in 100 μs increments
Compatible modules	All types of modules*1
Unit of I/O-refreshed devices	Per word basis, i.e., in units of 16 relays or terminals.
Maximum number of I/O-refreshed words	4 to 512
Applicable instructions	All instructions, except for the Timer, special module High-speed Read (HRD) and special module High-speed Write (HWR) instructions
Applicable devices	All device types*2
Maximum program execution time	50 μs to 24.95 ms
Initial condition at normal program execution	Sensor control block at a stop
Interrupt timing	Configurable as "After instruction (execution)" or "Immediate (during instruction execution)." The default is "Immediate."
Priority of interrupts	Selectable from "Sensor CB interrupt has priority" (default) or "Input interrupt has priority."
Other functions	Start or stop by means of instruction or tool*3, or interrupt prohibition or prohibition cancellation by means of instruction.

*1 : (See "CAUTION" in "■ Compatible Modules".)
*2 : For details on writing to devices also used in normal scan, see Section 6.15.7, "Programming Precautions."
*3 : For start or stop by means of tools, see "Starting and Stopping Blocks" in WideField User's Manual (Online).

■ Compatible Modules

The sensor control block function can be used with all CPU modules. However, observe the precautions described below.

CAUTION

- **Precautions when setting “Terminal Usage” in Input/Output Setup**
  Using the configuration function of WideField3 (or WideField2), specify whether terminals of I/O modules are to be refreshed by the sensor control block or normal scan.

  All devices used in the sensor control block including input relays (X) and output relays (Y) are shared with the normal scan.

  When using an input or output module with the sensor control block, you must configure the input module on long word basis (i.e., 32 relays, 32 terminals, terminals 1 to 32 or terminals 33 to 64); and the output module on word basis (i.e., 16 relays or 16 terminals).

  Let's suppose you configured the input module incorrectly on word basis (for example, you set terminals 1 to 16 to the option “Used” [with normal scan] and terminals 17 to 32 to the option “Use with SCB”). Since input refreshing is performed on long word basis, input relays (X) used under a normal scan are refreshed by the Refresh instruction of the sensor control block when the normal scan is in progress. Consequently, the simultaneity of data is not guaranteed before and after the refreshing. Simultaneity of data is also not guaranteed for input relays (X) used in the sensor control block.

CAUTION

- **Relationship between shared refreshing and link refreshing**
  Shared and extended shared relays (E), shared and extended shared registers (R), link relays (L) and link registers (W) are not refreshed by I/O refreshing of the sensor control block. Instead, these devices are refreshed during common (synchronization) processing in a normal scan.

■ Maximum Number of I/O-refreshed Words

The maximum number of words that can be configured for refreshing depends on the type of unit (main unit or subunit) where the module is installed, the execution interval and the number of installed CPU modules (including BASIC CPU modules) as show in the table below.

You can calculate the number of words according to the following equation.

Number of I/O-refreshed words

= (Number of words of I/O modules in main unit to be refreshed)

+ (Number of F3XH04 modules in main unit to be refreshed that use pulse catch function)

+ (Number of words of I/O modules in subunit to be refreshed) x 4

+ (Number of F3XH04 modules in subunit to be refreshed that use pulse catch function) x 4

Table 6.23 Maximum Number of I/O-refreshed Words Handled by Sensor Control Block for each CPU

Execution Interval	Maximum Number of I/O-refreshed Words			
	One CPU	Two CPUs	Three CPUs	Four CPUs
200 µs	4	0	0	0
300 µs	8	4	0	0
400 µs	12	4	4	0
500 µs	16	8	4	4
1 ms	36	16	12	8
2 ms	76	36	24	16
3 ms	116	56	36	28
4 ms	156	76	52	36
5 ms	196	96	64	48
10 ms	396	196	132	96
20 ms	512	396	264	196
25 ms	512	496	332	248

Example 1:

Number of CPU modules installed : 2

Execution interval of sensor control block at CPU1 : 1 ms

Sum of input-refreshed and output-refreshed words for CPU1 : Should be kept below 16.

Execution interval of sensor control block at CPU2 : 500 µs

Sum of input-refreshed and output-refreshed words of CPU2 : Should be kept below 8.

The maximum number of I/O-refreshed words is proportional to the execution interval. You can calculate the maximum number for any execution interval not given in the above table by using the maximum numbers given for the execution intervals immediately above and below the required execution interval.

Example 2:

Number of CPU modules installed : 1

Execution interval of sensor control block : 600 µs

Maximum number of I/O-refreshed words : 20

If the maximum number of I/O-refreshed words is exceeded, the execution interval of the sensor control block cannot be maintained. This may result in a sensor CB scan timeout error. For details on the module operation in the event of a sensor CB scan timeout error, see subsection 6.15.6, “Error Handling.”

■ Maximum Program Execution Time

You should keep the program execution time of the sensor CB as short as possible. Otherwise, a program executed under a normal scan will be interrupted for a longer duration, and the time interval of the normal scan will become longer.

Calculate the maximum program execution time using the equation given below. If this maximum time is exceeded, a sensor CB scan timeout error may result because the CPU is unable to maintain the execution interval of the sensor control block.

For details on the module operation in the event of a sensor CB scan timeout error, see subsection 6.15.6, “Error Handling.”

Maximum program execution time (µs)

= (Maximum number of I/O-refreshed words discussed earlier
- Number of words actually refreshed)
x Number of CPU modules installed x 25 µs + 50 µs

Example 1:

Execution interval : 200 µs

Number of CPU modules installed : 1

Maximum number of I/O-refreshed words : 4 (from previous Table)

Number of words actually refreshed : 2

Maximum program execution time = (4 - 2) x 1 x 25 + 50 = 100 µs

The sensor control block, if composed of basic instructions only, is equivalent to a program with the following number of steps.

100/0.09 = 1111 steps = Approximately 1.1K steps
(for F3SP22, F3SP28 and F3SP38 CPU modules)

100/0.035 = 2856 steps = Approximately 2.8K steps
(for F3SP53, F3SP58 and F3SP59 CPU modules)

Example 2:

Execution interval : 1 ms

Number of CPU modules installed : 1

Maximum number of I/O-refreshed words : 36 (from previous Table)

Number of words actually refreshed : 6

Maximum program execution time = (36 - 6) x 1 x 25 + 50 = 800 µs

The sensor control block, if composed of basic instructions only, is equivalent to a program with the following number of steps.

800/0.09 = 8888 steps = Approximately 8.7K steps
(for F3SP22, F3SP28 and F3SP38 CPU modules)

800/0.035 = 42865 steps = Approximately 41.8K steps
(for F3SP53, F3SP58 and F3SP59 CPU modules)

6.15.4 Functions

The following sensor control functions are available.

Table 6.24 Sensor Control Functions

Functions	Description
Execution interval	Allows the execution interval of the sensor control block to be set by WideField3 (or WideField2).
Interrupt timing	Allows the interrupt timing of the sensor control block to be set by WideField3 (or WideField2) to either of the following options: - After instruction - Immediate
Priority of interrupts	Allows the precedence of sensor control block interrupt and input module interrupt to be set by WideField3 (or WideField2).
Activate/Inactivate	Allows the sensor control block to be activated or inactivated by a dedicated instruction.
Disable/Enable	Prohibits the sensor control block from being executed or cancels the prohibition by a dedicated instruction.
On-for-One-Scan-at-Sensor-CB-Start function	A special relay (M) that remains turned on for one scan when the sensor control block is activated.
Execution status	Reflects the Start or Stop status of the sensor control block in a special relay (M).
Execution time monitoring	Stores the processing time of the sensor control block in a special register (Z).

■ Execution Interval Setting and Accuracy

Using the configuration and interrupt setting function of WideField3 (or WideField2), set the execution interval of the sensor control block.

Table 6.25 Execution Interval Setpoints of Sensor Control Block

Item	Configuration Range
Setting range	200 µs to 25.0 ms. The setting range from 200µs to 900µs is only valid for the interrupt timing option of “Immediate"
Unit of setpoint	100 µs

The accuracy of an execution interval is 100 ppm.

■ Interrupt Timing

Using the configuration and interrupt setting function of WideField3 (or WideField2), set the timing for sensor control block interrupts during program execution. Two interrupt timing options are available.

Table 6.26 Sensor Control Block Interrupt Timing

Interrupt Timing	Description
After instruction	The CPU switches to the sensor control block after the completion of instruction execution. It does not, however, switch to the sensor control block during common processing or refreshing.
Immediate (default)	The CPU switches to the sensor control block during ladder instruction execution. It also switches to the sensor control block during common processing or refreshing.

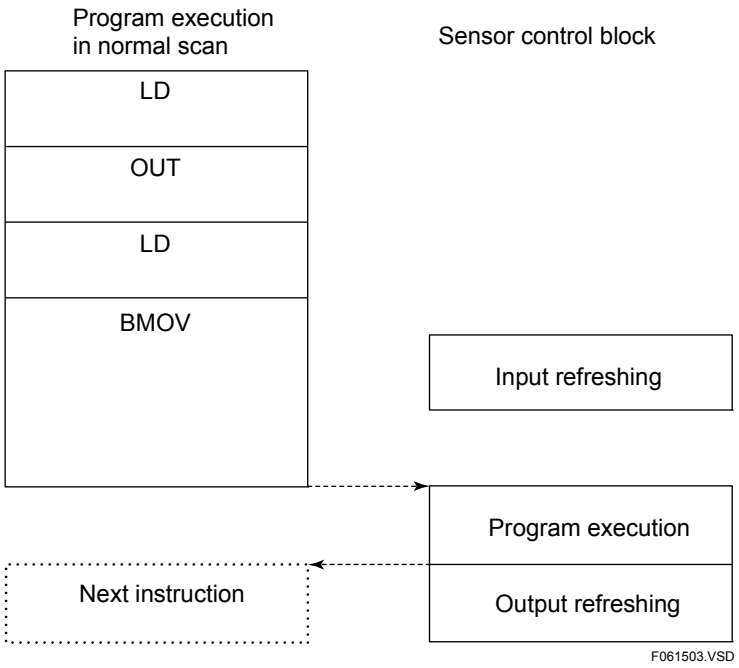


Figure 6.43 Interrupt by Sensor Control Block after Instruction Execution

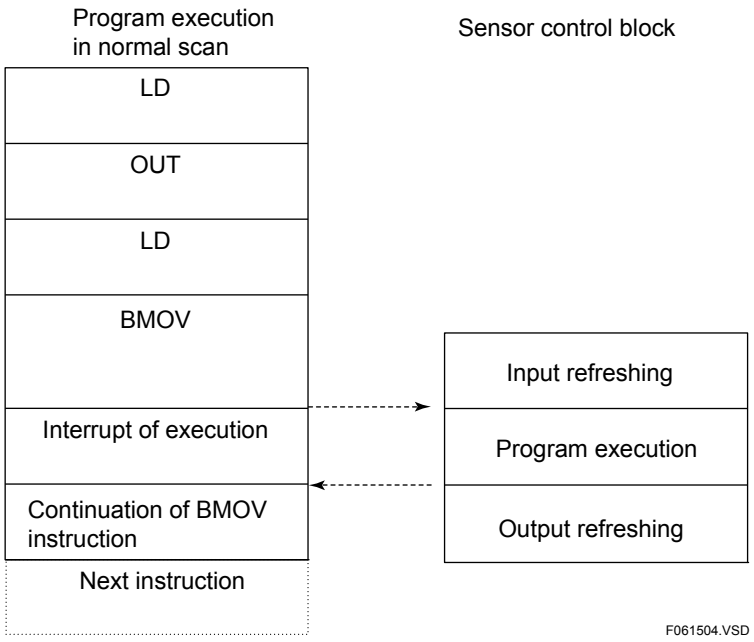


Figure 6.44 Immediate Interrupt by Sensor Control Block during Instruction Execution

The table below summarizes the differences between these two interrupt timing options.

Table 6.27 Differences between the Two Interrupt Timing Options

Item	When Interrupt Timing is Immediate	When Interrupt Timing is After Instruction Execution
Execution delay	Processing time of instruction being executed (Note 1) + Switchover processing time (Note 2), or Synchronization processing time (Note 3) + Common processing time (Note 3) + Input refreshing time (Note 3) + Switchover processing time (Note 2)	Switchover processing time only (Note 2)
Simultaneity of data	Guaranteed for each instruction	No simultaneity of multi-device data

Note 1: For details on instruction processing time, see the instruction list in the Appendix of "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).
Note 2: 9 to 30 µs for F3SP22, F3SP28 and F3SP38 CPU modules and 3 to 10 µs for F3SP53, F3SP58 and F3SP59 CPU modules
Note 3: See Section 7.1, "Description of Scan Time."

CAUTION

- **Setting execution interval when the interrupt timing is after instruction execution**

  If the interrupt timing is set to be after instruction execution, you should set the execution interval to 1 ms or longer.

  With this interrupt timing option, the CPU does not switch to the sensor control block during common processing or refreshing. Although the common processing time or refreshing time varies depending on the duration of synchronization processing, such as shared refreshing or link refreshing, you should set the execution interval to at least 1 ms. Otherwise, the execution interval of the sensor control block cannot be maintained and this may result in a sensor CB scan timeout error.

- **Debug operation when the interrupt timing is after instruction execution**

  If the interrupt timing is set to be after instruction execution, a sensor scan timeout error may occur when switching from **Run** to **Debug** mode or from **Debug** to **Run** mode, or when canceling a forced set/reset in **Debug** mode.

- **Simultaneity of data when the interrupt timing is immediate**

  Simultaneity of data for multiple devices is not guaranteed if the sensor control block is executed immediately during instruction execution.
  For example, consider the case shown in the previous figure where the sensor control block is executed during execution of a block transfer (BMOV) instruction in a normal scan. There is a risk that the source data that is partially transferred may be overwritten after the execution of the sensor control block or data transferred partially to the destination may be read by the sensor control block.

  Simultaneity of data is required when data of multiple devices is exchanged between a normal-scan program and the sensor control block program using a block transfer (BMOV) instruction, a long-word instruction with IEEE single-precision floating point data, or two or more instructions.

  If simultaneity of data is required when interrupt timing is configured as “Immediate (during instruction execution),” use any of the following means to ensure data simultaneity:

  1. Use a Disable Sensor Control Block (CBD) instruction and an Enable Sensor Control Block (CBE) instruction to prevent the sensor control block from being executed during exchange of multi-device data.
  2. Write an application program to perform flag control between the normal-scan program and the sensor control block using relays.

■ Priority of Interrupts

You can specify the priority of interrupts by configuration using WideField3 (or WideField2) for conflict resolution in the event that interrupt processing of an interrupt from an input module coincides with an interrupt from a sensor control block.

The table below lists the two options for “Priority of Interrupts”, along with how they work.

Table 6.28 Options for Priority of Interrupts

Priority of Interrupts	Functionality	
	When an interrupt from an input module occurs during execution of a sensor control block	When the time for executing a sensor control block arrives during input interrupt processing
Sensor CB interrupt has priority (default)	Executes the input interrupt process after executing the sensor control block.	Suspends the input interrupt process and resumes execution after executing the sensor control block.
Input interrupt has priority	Suspends the execution of the sensor control block and resumes execution after executing the input interrupt process.	Executes the sensor control block after executing the input interrupt process.

TIP

The sequence CPU applies the rule of interrupt execution timing (after completion of instruction execution or immediately during instruction execution) discussed earlier, even in the case where execution of the sensor control block or input interrupt process is suspended due to priority of interrupts.

■ Activating/Deactivating Sensor Control Block

You can activate the sensor control block using an Activate Sensor Control Block (CBACT) instruction, or stop the sensor control block using an Inactivate Sensor Control Block (CBINA) instruction. At the start of operation, the sensor control block defaults to the Stop status. To activate the sensor control block, execute an Activate Sensor Control Block (CBACT) instruction in a normal-scan program. An initial startup of the sensor control block takes place within 100 µs after the execution of a CBACT instruction.

Table 6.29 Instructions to Activate or Inactivate the Sensor Control Block

Instruction	Description
CBACT instruction	Activates the sensor control block.
CBINA instruction	Inactivates the sensor control block.

SEE ALSO

For details on individual instructions, see "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

TIP

When the sensor control block stops, the CPU holds or resets the data of output (Y) relays used or refreshed by the sensor control block according to the setting of the configuration item “Output when Stopped.” When the sensor control block is activated, the relays are always set to the Hold option.

TIP

The initialization processing of timers, counters and the destinations of OUT instructions does not apply to the sensor control block activated or inactivated by ACT/INACT instructions.

■ Enabling/Disabling Sensor Control Block

You can temporarily disable the sensor control block using a CBD (Disable Sensor Control Block) instruction or enable the sensor control block using a CBE (Enable Sensor Control Block) instruction. If the sensor control block is disabled, the CPU does not execute it until it is enabled even if it is time for executing the sensor control block. In this case, the CPU immediately begins executing the block as soon as it is enabled.

Table 6.30 Instructions to Disable and Enable Execution of the Sensor Control Block

Instruction	Description
CBD Instruction	Disables execution of the sensor control block.
CBE Instruction	Enables execution of the sensor control block.

SEE ALSO

For details on individual instructions, see "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

CAUTION

If the interval of execution disable is too long for the CPU to be able to execute the sensor control block at fixed intervals, a sensor CB scan timeout error will result. Consequently, the CPU stops executing the sensor control block. See subsection 6.15.6, “Error Handling” for more information.

■ On-for-one-scan-at-Sensor-CB-start Function

This function causes a special relay (M) to remain turned on for one scan during the first execution of the sensor control block when the sensor control block is activated.

Table 6.31 Special Relay (M) which Turns ON for One Scan at Sensor Control Block Startup

No.	Name	Status	Description
M097	On for One Scan at Sensor CB Start	ON : At block start OFF: In all other cases	Turns on for one scan when the sensor control block starts (at the first execution of the sensor control block).

TIP

The On-for-one-scan-at-Sensor-CB-start relay turns on when an Activate Sensor Control Block (CBACT) instruction is executed. It then turns off at the end of the first execution of the sensor control block.

■ Execution Status

The CPU stores in a special relay (M) the execution status of the sensor control block.

Table 6.32 Sensor CB Execution Status Special Relay (M)

No.	Name	Status	Description
M137	Sensor CB Execution Status	ON: Run OFF: Stop	Indicates the operating status of the sensor control block.

TIP

The status of the sensor control block is updated when an Activate Sensor Control Block (CBACT) instruction is executed, or during normal-scan input refreshing after error detection when an Inactivate Sensor Control Block (CBINA) instruction is executed.

■ Execution Time Monitoring

This function stores in a special register (Z) the time taken from when input refreshing for the sensor control block is started, followed by program execution, to when output refreshing is completed . The time indicates the actual execution time of the sensor control block during the preset execution interval.

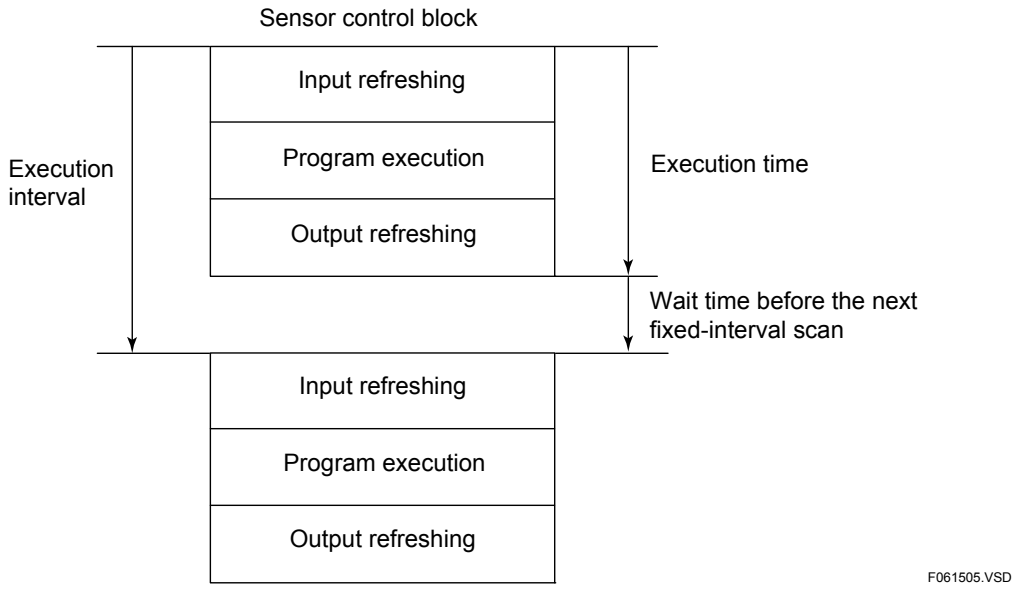


Figure 6.45 Schematic Diagram Showing Execution Time of Sensor Control Block

Table 6.33 Special Registers (Z) for Execution Time of Sensor Control Block

No.	Name	Description
Z109	Sensor CB Execution Time	Indicates the time taken from starting of input refreshing for the sensor control block through program execution to completion of output refreshing.
Z111	Maximum Sensor CB Execution Time	Indicates the maximum time taken to execute the sensor control block (Unit: 10 μs).

TIP

The measured execution time of the sensor control block is updated at each normal-scan input refreshing process.

6.15.5 Procedures for Using Sensor Control Functions

The table below summarizes the procedures for using the sensor control functions.

Table 6.34 Procedures for Using Sensor Control Functions

Procedure	Setup Item/Edit Item	WideField3 Function (WideField2)	Reference to Items Discussed Earlier
1	Defining I/O-refreshed words of sensor control block	Configuration	Maximum Number of I/O-refreshed Words
	Defining the execution interval	Input/Output Setup	Execution Interval Setting and Accuracy
	Defining the interrupt timing	Interrupt Setup	Interrupt Timing
	Defining the priority of interrupts		Priority of Interrupts
	Creating normal-scan programs	Block editing	Activating/Deactivating Sensor Control Block
	Creating sensor control programs		
2	Registering sensor control block	Component definition	

TIP

You can use the sensor control block, irrespective of whether the configuration item “Program Execution Mode” is set to “All Blocks” or “Specified Blocks.”

6.15.6 Error Handling

The table below summarizes the errors that may be reported when the sensor control functions are in use.

Table 6.35 Errors Related to Sensor Control Functions

Type of Error	Description
Sensor CB scan timeout error	The CPU fails to maintain the execution interval because it is exceeded by the sum of the fixed-interval I/O refreshing time and the execution time of sensor control programs.
Scan timeout	The CPU has insufficient time to execute a regular program because the execution time of sensor control programs is too long. Thus, the normal scan time exceeds the scan monitoring time.
I/O module error	An I/O module has failed during fixed-interval I/O refreshing in the sensor control block.

CAUTION

If multiple I/O module errors are detected during I/O refreshing in the sensor control block, the CPU only reports the first detected error by means of an alarm indicator.

Table 6.36 Sensor CB Scan Timeout Special Relay (M)

Item	Self-diagnosis Status		
No.	Name	Description	Status
M212	Sensor CB scan timeout	ON : Error OFF: Normal	Indicates that the CPU cannot maintain the execution interval of the sensor control block.

Table 6.37 Actions when an Error Is Encountered in Sensor Control Block

Type of Error Encountered	Action in Case of Error	
	Sensor Control Block	Effect on Normal Scans
Sensor CB scan timeout (Special relay M212 turns on.)	Stop	The program either runs or stops depending on the "Error-time Action" defined for Sensor CB scan timeout error in "Operation Control" of configuration.
Other errors	For errors configurable with "Operation Control" of configuration, the sensor control block either runs or stops depending on the "Error-time Action" defined. The block stops if the error is not configurable.	

TIP

If an error is encountered in the sensor control block, the error block number stored in a special register (Z) is the last block number of a normal scan program plus one.

Table 6.38 Action of Sensor Control Block when an Error Is Detected in Normal Scan

Type of Error	Action
All types of error	For errors configurable with "Operation Control" of configuration, the sensor control block either runs or stops depending on the "Error-time Action" defined. The block stops if the error is not configurable.

6.15.7 Programming Precautions

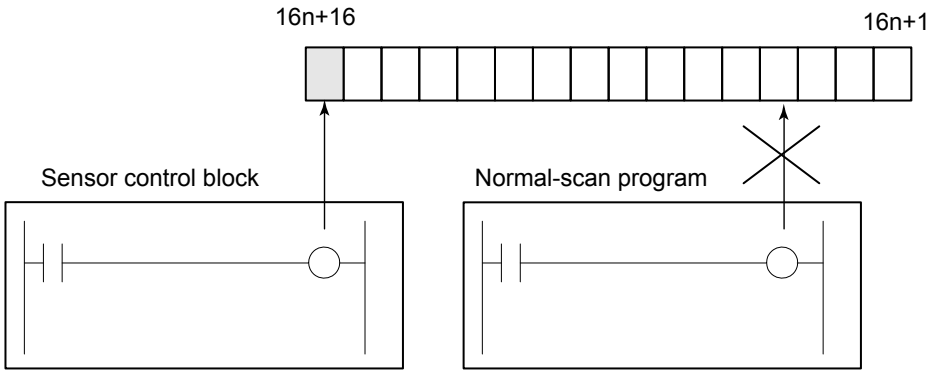
■ Instructions Not Applicable to Sensor Control Block

Instruction	Corrective Actions
Timer (TIM) instruction	Enable/disable timers in a regular block. Reading timer relays in a sensor control block is allowed, however.
Special module High-speed Read (HRD)	Use the special module Read (READ) instruction instead.
Special module High-speed Write (HWR)	Use the special module Write (WRITE) instruction instead.

■ Precautions when the Interrupt Timing Is Immediate

● Precautions when outputting data to relays

If you have already output data to any of the relays numbered 1 to 16 using an OUT, SET RST, DIFU or DIFD instruction in the sensor control block, do not also output data to any of these relays in a normal-scan program. Otherwise, the output instruction may not be processed correctly. This precaution is also true with other groups of 16 relays numbered 17 to 32, 33 to 48, 49 to 64, and so on. Do not output data to relays within the same group both in the sensor control block and in a normal-scan program.



Example: If a sensor control block controls I00032,
the normal-scan program must not control I00017 to I00031.

F061506.VSD

Figure 6.46 Precautions for Relay Output

● Simultaneity of multi-device data

Simultaneity of data for multiple devices is not guaranteed.

For example, consider the case shown in Figure 6.44 where the sensor control block is executed during execution of a block transfer (BMOV) instruction in a normal scan. There is a risk that the source data that is partially transferred may be overwritten after the execution of the sensor control block or data transferred partially to the destination may be read by the sensor control block.

Simultaneity of data is required when data of multiple devices is exchanged between a normal-scan program and the sensor control block program using a block transfer (BMOV) instruction, a long-word instruction with IEEE single-precision floating point data, or two or more instructions.

If simultaneity of data is required when interrupt timing is configured as “Immediate (during instruction execution),” use any of the following means to ensure data simultaneity.

1. Use a Disable Sensor Control Block instruction (CBD) and an Enable Sensor Control Block (CBE) instruction to prevent the sensor control block from being executed during exchange of multi-device data.
2. Write an application program to perform flag control between the normal-scan program and the sensor control block using relays.

● Data simultaneity of devices to be refreshed

Simultaneity of data is not guaranteed if, in the sensor control block, an access is made to I/O relays (X/Y) refreshed in a regular block or to shared/extended shared relays (E), shared/extended shared registers (R), link relays (L), or link registers (W).

The sensor control block is executed even during normal-scan input refreshing, output refreshing and common processing. If you read any of the above-mentioned devices in the sensor control block, a device value being refreshed may be read. Likewise, if you write to the device, the device value being refreshed may be overwritten. Consequently, simultaneity of data may be lost.

To prevent the sensor control block from being executed during normal-scan input refreshing, output refreshing and common processing, execute a Disable Sensor Control Block (CBD) instruction to disable the block at the end of a regular program and execute an Enable Sensor Control Block (CBE) instruction to enable the block at the beginning of the program.

6.16 Partial Download Functions

F3SP□□-□S

The partial download functions allow only specified blocks/macros to be downloaded to a CPU to replace corresponding blocks/macros of a program that has been downloaded earlier.

This reduces downloading time and improves debugging efficiency, especially in large-scale program development by a group of developers.

These functions are available only in STOP mode. It allows multiple blocks or macro instructions to be specified for downloading. Addition or deletion of block/macro instructions by partial downloading is not allowed.

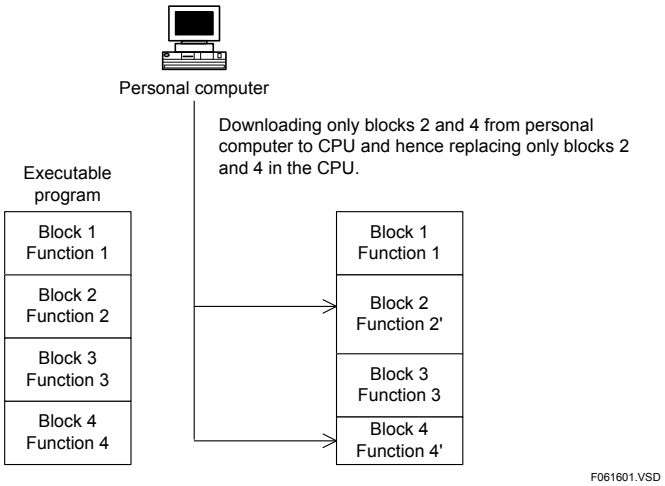


Figure 6.47 Partial Download Functions

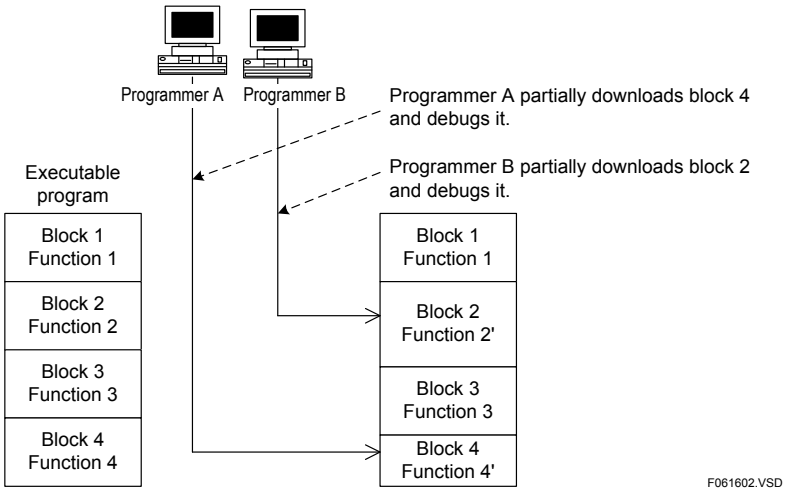


Figure 6.48 Partial Download Functions Used by Multiple Programmers

CAUTION

- If an error occurs at the time of partial downloading, the step count of the error block becomes 0. If you then upload this defective program to a personal computer, the step count for the corresponding block on the personal computer will also be 0. If you have to upload such a program, save it under a different project name.
- At the completion of partial downloading, program checking and optimization are performed and this may take some time.

6.17 Functions for Storing Comments to CPU

F3SP□□-□S

You can store circuit comments and subcomments to the CPU module.

Storing comments in the CPU module allows you to display them during circuit monitoring even if there is no project.

TIP

These functions can only store circuit comments and subcomments. To store I/O comments, use the functions for storing tag name definitions to CPU.

6.17.1 Performing Setup to Download Comments

You can select whether to store (download) circuit comments and subcomments to a CPU module but you cannot select to download only circuit comments or only subcomments.

In WideField3 (or WideField2), setup for downloading comments has to be performed in two places: in the local device properties and when you execute the download function.

Firstly, turn on the checkbox "Store Circuit Comment/Subcomment" in the storing to CPU settings in the local device properties for each relevant block (macro instruction).

Then, turn on the download comments checkbox when you execute the download program function. Turning on this checkbox downloads comments to the CPU module according to the block properties.

Turning off the circuit comments/subcomments checkbox when you execute the download program function will not download comments regardless of the block properties setup.

SEE ALSO

For details on block properties and program downloading, see "FA-M3 Programming Tool WideField3" (IM 34M06Q16-□□E) or "FA-M3 Programming Tool WideField2" (IM 34M06Q15-01E).

CAUTION

Note that if you turn off the checkbox for circuit comments/subcomments when you execute the download program function, comments will not be downloaded to the CPU module regardless of the local device properties setup.

6.17.2 Number of Steps Needed for Comments

Like program steps, circuit comments/subcomments also takes up program area. Thus, how much of the program area is consumed in terms of step count also depends on whether comments are downloaded to a CPU module.

■ Calculating the Step Count of Comments

● If comments are downloaded:

The step count of a circuit comment or a circuit subcomment is the sum of the comment offset (1 step) and step count of the character string, as given below.

Step count of a comment = comment offset (1 step) +
step count of the character string

The step count of the character string is calculated as follows:

Sum the step counts of all characters in a character string, using 0.25 steps for each single-byte character and 0.5 steps for each double-byte character, and round up to the nearest integer.

TIP

- Example: Assume that a comment is a character string consisting of four single-byte character and five double-byte characters.

  Summing the step counts of individual characters yields:

  4 x 0.25 (for single-byte characters) ＋ 5 x 0.5 (for double-byte characters) = 3.5 (steps)

  Rounding up to the nearest integer yields:

  Step count of the character string = 4 (steps)

  Adding 1 step for comment offset:

  Step count of the comment = 4 ＋ 1 = 5 (steps)

● If no comments are downloaded:

One step of program area is consumed for each comment (as comment offset).

CAUTION

One step of comment offset is added for each comment to the step count for a program even if the comments are not downloaded.

■ Checking the Program Step Count (including Comments)

The step count of a block (or macro instruction) containing comments is displayed on the status bar when the block is opened. The displayed step count includes the step counts of circuit comments/subcomments and block tag name definition specified to be downloaded to the CPU module in the local device properties window. If you select to only download the program, the displayed step count includes only the step count of the program.

6.17.3 Online Editing of Comments

If circuit comments/subcomments are stored in a CPU module, you can edit or delete them online but you cannot add new comments online.

CAUTION

If you have added circuit comments or subcomments using offline program editing, you should download the circuit comments and subcomments to the CPU module again.

6.18 Functions for Storing Tag Name Definitions to CPU

F3SP□□-□S

These functions store common, block, and macro tag name definitions along with a program to either the program memory of a sequence CPU module or the ROM pack.

If tag name definitions are to be stored in the program memory of a sequence CPU module, the sum of the program and tag name definition step counts must be within the capacity of the program. If tag name definitions are to be stored in the ROM pack, the sum of the program and tag name definition step counts must be within the capacity of the ROM pack.

Table 6.39 Program Capacity for Storing Tag Name Definitions

CPU Module	Where to Store	Program Only	Program plus Tag Name Definitions
F3SP22-0S	CPU memory	10K steps	10K steps
	RK33-0N		56K steps
	RK73-0N		120K steps
F3SP28-3S	CPU memory	30K steps	30K steps
	RK33-0N		56K steps
	RK73-0N		120K steps
F3SP53-4S	CPU memory	56K steps	56K steps
	RK33-0N		56K steps
	RK73-0N		120K steps
F3SP38-6S F3SP58-6S	CPU memory	120K steps	120K steps
	RK33-0N	56K steps	56K steps
	RK73-0N	120K steps	120K steps
	RK93-0N	120K steps	360K steps
F3SP59-7S	CPU memory	254K steps	360K steps
	RK33-0N	56K steps	56K steps
	RK73-0N	120K steps	120K steps
	RK93-0N	254K steps	360K steps

For the step count of tag name definitions, check the project properties, the block tag name definition properties for each block, or the macro tag name definition properties for each macro instruction.

You can separately specify whether to download common, block, and macro tag name definitions using the project properties, the local device properties for each block, or the local device properties for each macro instruction.

In addition, at the time you execute the download program function, you can choose to disable the downloading of tag name definitions, regardless of the properties setup.

If you specify not to download tag name definitions when you execute the download program function, any tag name definitions previously downloaded will be erased after the download.

If you edit tag name definitions online, the tag name definition files on the personal computer will be updated but not those in the program memory of the CPU module. If changes are made to the tag name definitions, download them to the CPU module again.

TIP

For programming efficiency, we recommend that you maintain the tag name definitions on the personal computer without storing them in the CPU module during debugging and program development, and download the tag name definitions to the CPU module after the programs are debugged.

6.19 Structures

F3SP□□-□S

A structure represents a group of data under a unified name. It improves device representation and program readability.

The instructions related to structures are:

- Structure Move (STMOV)
- Structure Pointer Declaration (STRCT)
- Structure Macro Instruction Call (SCALL)

For details on structures, see "FA-M3 Programming Tool WideField3" (IM 34M06Q16-□□E) or "FA-M3 Programming Tool WideField2" (IM 34M06Q15-01E). For details on the instructions, see "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

7. I/O Response Time Based on Scan Time

This chapter discusses examples of calculating the scan time and I/O response time.
It also explains such parameters as instruction execution time.

7.1 Description of Scan Time

As discussed earlier in Chapter 3, the sequence CPU module is designed so that two systems of processes, i.e., a system of control-related processes and a system of peripheral processes, run concurrently and independently. For this reason, the system of control-related processes whose main purpose is to execute programs and control-related processes is not affected by the system of peripheral processes whose purpose is to support communication and WideField3 or (WideField2). Thus, the system of control-related processes can run at extremely high speeds. Under normal conditions, the scan time of the sequence CPU module is equivalent to the time taken by the system of control-related processes. The following paragraphs explain the processing tasks and time of each of these systems.

● System of Control-related Processes

The latest, minimum and maximum of scan times taken by the system of control-related processes are stored in special registers Z001 to Z003 in that order.

Table 7.1 Scan Time of System of Control-related Processes

Item	Processing Task	Processing Time
Common processing	Self-diagnosis	Fixed at 0.2 ms
Program execution	Executes ladder programs. The scan time is calculated using the program execution time or output refreshing time, whichever is greater.	The scan time is the sum of the execution times of basic and application instructions. It varies depending on the execution time of each instruction word. For details, see Section 7.5, "Instruction Execution Time."
Output refreshing	Writes the contents of output relays (Y) to an output module.	12 µs x number of modules calculated on a 16-points basis*
Shared refreshing	Updates the contents of shared/extended shared relays (E) and shared/extended shared registers (R) when add-on CPU(s) are installed and shared refreshing is configured as a control-related process. In a single refreshing cycle, this task updates the contents of shared/extended shared relays (E) or shared/extended shared registers (R) included in the configuration for each CPU.	When an add-on CPU module is installed and shared refreshing is configured as a control-related process: - 0.003 x (number of shared relays set in the sequence CPU module for refreshing/32+number of shared registers set in the sequence CPU module for refreshing/2)+0.10 ms, if the sequence CPU module for which the devices are refreshed is F3SP22, F3SP28, F3SP38, F3SP53, F3SP58, F3SP59, F3SP66, F3SP67, F3SP71 or F3SP76. - 0.014 x (number of shared relays set in the sequence CPU module for refreshing/32+number of shared registers set in the sequence CPU module for refreshing/2)+0.10 ms, if the sequence CPU for which the devices are refreshed is other than those listed above.
	Not performed if no add-on CPU module is installed or shared refreshing is configured as a peripheral process.	0.00 ms
FL-net link refreshing	Updates the contents of link relays (L) and link registers (W) when FL-net interface module(s) is installed and FL-net link refreshing is configured as a control-related process.	0.0005 x (Number of link relays to be refreshed/16 + Number of link registers to be refreshed) + 0.30 ms
	Not performed if no FL-net interface module is installed or FL-net link refreshing is configured as a peripheral process.	0.00 ms
Input refreshing	Write the contents of input modules to CPU input relays (X).	6 µs x number of modules calculated on a 16-point basis*
Synchronization processing	Ensures synchronization of operation control related processing and the simultaneity of data between the system of control-related processes and the system of peripheral processes.	- When output relays (Y) are used: 8 µs x number of modules calculated on a 16-point basis* - When FA link modules are used: 0.003 x (number of relays used in FA link to be refreshed/16 + number of registers used in FA link to be refreshed) + 0.05 ms. - When an FL-net module is used and FL-net link refreshing is configured as a peripheral process: 0.0004 x (number of link relays to be refreshed/16 + number of link registers to be refreshed)+ 0.30 ms - When an add-on CPU is installed and shared refreshing is configured as a peripheral process: 0.002 x (number of shared relays set in CPU for refreshing/32 + number of shared relays set in the CPU itself/32 + number of shared registers set in CPU for refreshing/2 + number of registers set in the CPU itself/2) + 0.05 ms. - When an add-on CPU module is installed and shared refreshing is configured as a control-related process: 0.002 x (number of shared relays set in the CPU itself/32 + number of shared registers set in the CPU itself/2) + 0.05 ms.
Peripheral processing	Performs peripheral processes.	Minimum peripheral processing time (0.2 ms if not configured) or sum of program execution time + output refreshing time, whichever is greater.

*** Relationship between Types of I/O Module and Number of Modules Calculated on a 16-points Basis**

Type of I/O Module	Number of Modules Calculated on a 16-device Basis
4-point I/O module	1
8-point I/O module	1
14-point I/O relay	1
16-point I/O relay	1
32-point I/O relay	2
64-point I/O relay	4

● System of Peripheral Processes

The latest, maximum and minimum of scan times taken by the system of peripheral processes are stored in special registers Z007 to Z009 in that order.

Table 7.2 Scan Time of System of Peripheral Processes

Item	Processing Task	Processing Time
Shared refreshing	Updates the contents of shared/extended shared relays (E) and shared/extended shared registers (R) when add-on CPU module(s) is installed and shared refreshing is configured as a peripheral process. In a single refreshing cycle, this task updates the contents of shared/extended shared relays (E) and shared/extended shared registers (R) included in the configuration setting, for each CPU.	When an add-on CPU module is installed and shared refreshing is set as a peripheral process: - 0.003 x (number of shared relays set in CPU for refreshing/32 + number of shared registers set in CPU for refreshing/2) + 0.10 ms, if the CPU module for which the devices are refreshed is F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 or F3SP59. - 0.014 x (number of shared relays set in CPU for refreshing/32 + number of shared registers set in CPU for refreshing/2) + 0.10 ms, if the CPU module for which the devices are refreshed is other than those listed above.
	Not performed if no add-on CPU module is installed or shared refreshing is configured as a control-related process.	0.00 ms
FA-link refreshing	Updates the contents of link relays and registers when an FA link module is installed.	When an FA link module is installed: 0.015 x (number of relays used in FA link for refreshing/16+number of registers used in FA link for refreshing)+0.06 ms
	Not performed if no FA link module is installed.	0.00 ms
FL-net link refreshing	Updates the contents of link relays (L) and link registers (W) when FL-net interface module(s) is installed and FL-net link refreshing is configured as a peripheral process.	0.0005 x (Number of link relays to be refreshed/16 + Number of link registers to be refreshed) + 0.30 ms
	Not performed if no FL-net interface module is installed or FL-net link refreshing is configured as a control-related process.	0.00 ms
Tool service	Processes commands input from the WideField3 (or WideField2) connected to the sequence CPU module. Executes one command per service.	Varies with the type of command.
Link service	Processes commands input from a personal computer link module. Executes one command per service.	Varies with the type of command.
CPU service	Processes commands input from another CPU module. Executes one command per service.	Varies with the type of command.

7.2 Setting Scan Monitoring Time

This configuration item sets the scan monitoring time. You can set the time to any value from 10 ms to 200 ms, in 10 ms increments. By default, the time is set at 200 ms.

7.3 Examples of Scan Time Calculation

● When the CPU is F3SP22, F3SP28 or F3SP38

Module configuration : Four 32-point input modules
: Four 32-point output modules

User program : 5K steps
(5K steps for the ladder program below consisting of instructions only, where the average execution time of these instructions is assumed to be 0.09 µs)

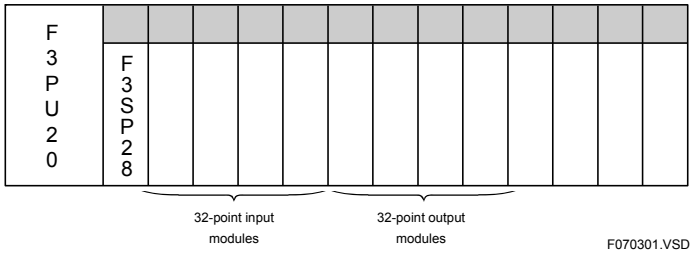


Figure 7.1 Module Configuration of F3SP22, F3SP28 or F3SP38 Sequence CPU

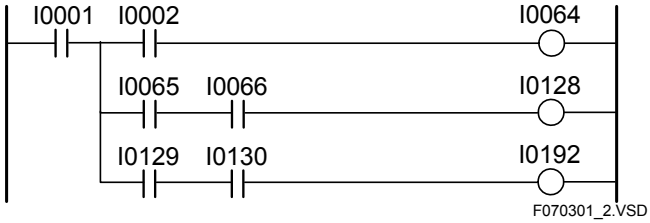


Figure 7.2 Program for F3SP22, F3SP28 or F3SP38

Table 7.3 Scan Time of F3SP22, F3SP28 or F3SP38 Sequence CPU

Item	Calculation	Processing Time
Common processing	Fixed 0.2ms	0.2 ms
Program execution	0.09µs × 5120 = 461 µs	0.5 ms
Output refreshing	Number of modules calculated on a 16-point basis: 2 × 4 = 8 12 µs × 8 = 96 µs	0.1 ms*
Shared refreshing	When no add-on CPU module is installed: 0.00 ms	0.00 ms
Input refreshing	Number of modules calculated on a 16-point basis: 2 × 4 = 8 6 µs × 8 = 48 µs	0.05 ms
Synchronization Processing	Number of modules calculated on a 16-point basis: 2 × 4 = 8 8 µs × 8 = 64 µs	0.06 ms
Peripheral processing	Minimum peripheral processing time, if not yet defined: 0.2 ms	0.2 ms*
Scan time, which is the sum of all time spans listed above		0.8 ms

*: The output refreshing time and the minimum peripheral processing time are excluded from scan time calculation because the sum of these time spans is smaller than the program execution time.

● When the CPU Is F3SP53, F3SP58 or F3SP59

Module configuration : Four 32-point input modules
: Four 32-point output modules

User program : 20K steps
(20K steps for the ladder program below consisting of instructions only, where the average execution time of these instructions is assumed to be 0.035 µs

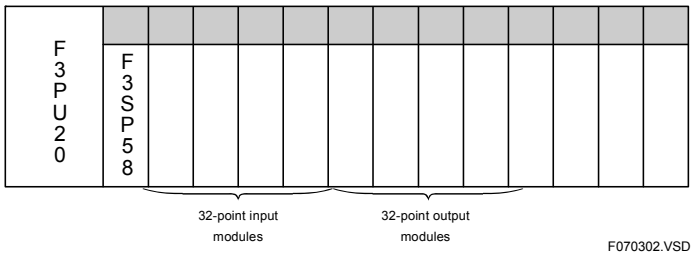


Figure 7.3 Module Configuration of F3SP53, F3SP58 or F3SP59 Sequence CPUs

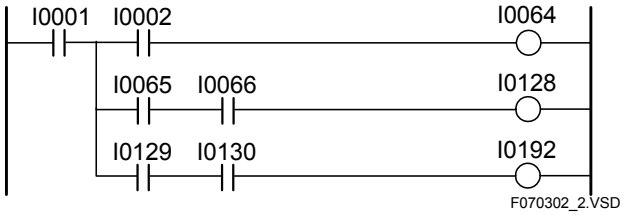


Figure 7.4 Program for F3SP53, F3SP58 or F3SP59

Table 7.4 Scan Time of F3SP53, F3SP58 or F3SP59 Sequence CPU

Item	Calculation	Processing Time
Common processing	Fixed at 0.2 ms.	0.2 ms
Program execution	0.035 µs × 20480 = 717 µs	0.7 ms
Output refreshing	Number of modules calculated on a 16-points basis: 2 × 4 = 8 12 µs × 8 = 96 µs	0.1 ms*
Shared refreshing	When no add-on CPU module is installed: 0.00 ms	0.00 ms
Input refreshing	Number of modules calculated on a 16-points basis: 2 × 4 = 8 6 µs × 8 = 48 µs	0.05 ms
Synchronization processing	Number of modules calculated on a 16-points basis: 2 × 4 = 8 8 µs × 8 = 64 µs	0.06 ms
Peripheral processing	Minimum peripheral processing time, if not yet defined: 0.2 ms	0.2 ms*
Scan time, which is the sum of all time spans listed above		1.0 ms

*: The output refreshing time and the minimum peripheral processing time are excluded from scan time calculation because the sum of these time spans is smaller than the program execution time.

7.4 Example of I/O Response Time Calculation

● Calculation of the minimum I/O response time

Input response time: 16 ms

Output response time: 1 ms

Scan time: 2 ms

Minimum I/O response time = Input response time + Scan time + Output response time

= 16 ms + 2 ms + 1 ms = 19 ms

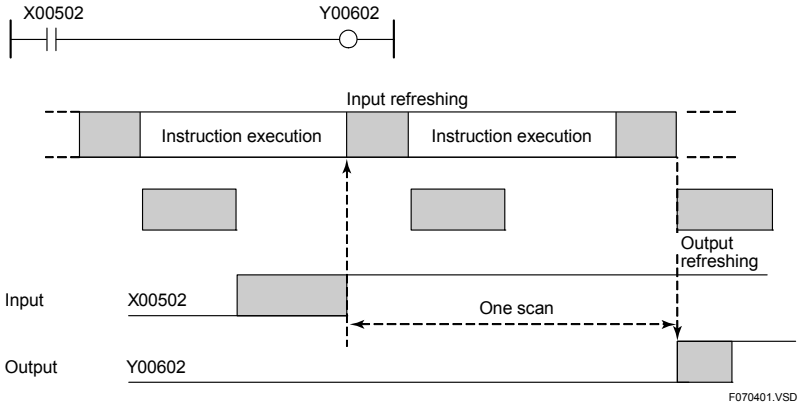


Figure 7.5 Minimum I/O Response Time

● Calculation of the maximum I/O response time

Input response time: 16 ms

Output response time: 1 ms

Scan time: 2 ms

Maximum I/O response time = Input response time + (Scan time x 2) + Output response time = 16 ms + (2 x 2) ms + 1 ms = 21 ms

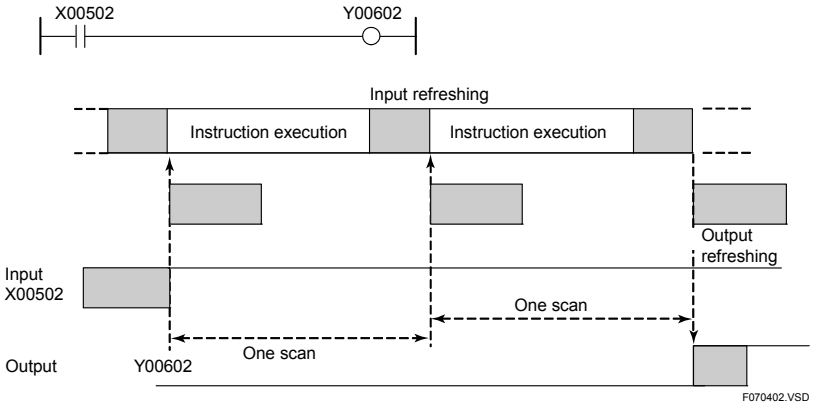


Figure 7.6 Maximum I/O Response Time

TIP

- The I/O response time refers to the total time taken to receive signal input from external input equipment, execute instructions and turn on external output equipment.
- Input response time refers to the time taken to load external input tag name using the input refreshing process.
- Output response time refers to the time taken to reflect the result of instruction execution in external output equipment using the output refreshing process.

7.5 Instruction Execution Time

SEE ALSO

For details on the execution time of each instruction, see Appendix 3, "List of Ladder Sequence Instruction" of “Sequence CPU Instruction Manual – Instructions (IM 34M06P12-03E).”

The instruction execution time varies somewhat depending on the contents of the input parameter or output parameter devices or the number of devices included in data transfer. The execution time lengths listed in the “List of Ladder Sequence Instructions” are typical. Use these values of the instruction execution time for reference purposes only when calculating the scan time. The instruction execution time varies somewhat with the conditions under which an instruction is executed as shown below. Use the instruction execution time (T) values given in the "List of Ladder Sequence Instructions" to calculate the instruction execution time under certain execution conditions.

Table 7.5 Calculation of Instruction Execution Time

Execution Conditions		Instruction Execution Time (µs)	
		F3SP22 F3SP28 F3SP38	F3SP53 F3SP58 F3SP59
Differential type instruction	When executed	T+0.18	T+0.07
	When not executed		
Relay (BIN format)	16 bits	T+2.5×N1	T+1.0×N1
	32 bits	T+3.5×N1	T+1.4×N1
I/O relays (X/Y) defined in BCD format	16 bits	T+3.5×N2	T+1.4×N2
	32 bits	T+4.5×N2	T+1.8×N2
Index modification	Basic instruction	T+1.0×N3	T+0.4×N3
	Application instruction	T+2.0×N4	T+0.8×N4

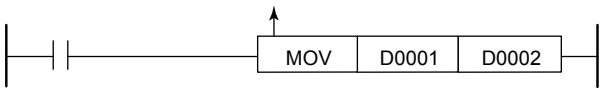
T : Instruction execution time given in “List of Ladder Sequence Instructions”.
N1 : Number of relay devices
N2 : Number of relay devices defined in BCD format
N3 : Number of index-modified devices

■ Examples of Calculation

Some examples for calculating the instruction execution time are given below.

For information on the execution time of an MOV instruction, see Appendix 3, “List of Ladder Sequence Instructions” of “Sequence CPU Instruction Manual – Instructions” (IM 34M06P12-03E).

(1) Differential Type Instructions

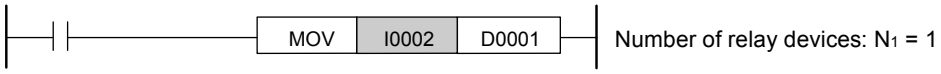


0.18+0.18=0.36 μs (for F3SP22, F3SP28 and F3SP38 sequence CPUs)
0.07+0.07=0.14 μs (for F3SP53, F3SP58 and F3SP59 sequence CPUs)

F070501.VSD

(2) Relays (BIN Format)

Use parenthesized execution time value from the “List of Ladder Sequence Instructions,” if any.

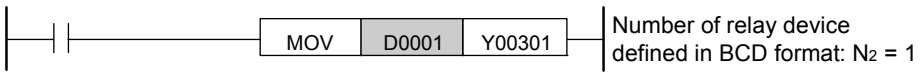


3.2+2.5=5.7 μs (for F3SP22, F3SP28 and F3SP38 sequence CPUs)
1.2+1.0=2.2 μs (for F3SP53, F3SP58 and F3SP59 sequence CPUs)

F070502.VSD

(3) I/O Relays (X/Y) Defined in BCD Format

Use parenthesized execution time value from the “List of Ladder Sequence Instructions,” if any.



3.2+3.5=6.7μs (for F3SP22, F3SP28 and F3SP38 sequence CPUs)
1.2+1.4=2.6μs (for F3SP53, F3SP58 and F3SP59 sequence CPUs)

F070503.VSD

(4) Index Modification

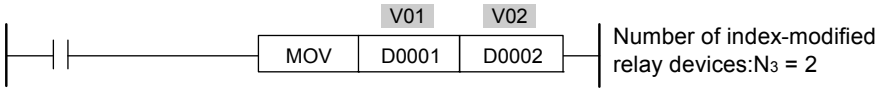
1. Basic Instructions

0.09+1.0=1.09 μs (for F3SP22, F3SP28 and F3SP38 sequence CPUs)
0.035+0.4=0.435 μs (for F3SP53, F3SP58 and F3SP59 sequence CPUs)

F070504.VSD

2. Application Instructions

   Use parenthesized execution time value from the “List of Ladder Sequence Instructions,” if any.



3.2+2.0x2=7.2 μs (for F3SP22, F3SP28 and F3SP38 sequence CPUs)
1.2+0.8x2=2.8 μs (for F3SP53, F3SP58 and F3SP59 sequence CPUs)

F070505.VSD

8. RAS Functions

This chapter describes the RAS functions of the sequence CPU module, such as the self-diagnosis and error logging functions that work if the module fails.

8.1 Self-diagnosis

The sequence CPU performs self-diagnosis on its device memory, instruction codes, and so on when the power is turned on or a program is being executed. The results of self-diagnosis are reflected in specific special relays (M) and registers (Z). If any failure is found during self-diagnosis, the CPU module updates the mode statuses of LED indicators and stops executing programs depending on the failure mode. The table below shows the classification of errors.

Table 8.1 Severity of Failure and CPU Module LED Indicator Status

Severity of Failure	LED Indicator Status	Failure Condition	Failure Mode	FAIL Signal Contact Output		Action of Output Module	
				Between FAIL1 and COM	Between FAIL2 and COM	Output modules other than F3YD64-1A, F3YD64-1M, F3WD64-□N, Y□□□□□ *2	F3YD64-1A, F3YD64-1M, F3WD64-□N, Y□□□□□ *2
Major failure	The green RDY lamp goes out.	The core hardware is disabled.	- CPU error - Memory crash	Shorted	Open	Default: RESET Configurable to HOLD or RESET on 16-terminal basis.	Always HOLD Not configurable
Moderate failure	The red ERR lamp comes on.	The user program cannot be started or run any further.	- Program error - I/O comparison error*1 - I/O module error*1 - Memory error - CPU error - Instruction error*1 - Scan timeout*1 - Startup error - Invalid instruction found - Excess number of I/O points - ROM pack error - Subroutine error*1 - Interrupt error*1 - Subunit communication error*1 - Sensor CB scan timeout *1 - Battery error	Shorted	Open	Default: RESET Configurable to HOLD or RESET on 16-terminal basis.	Default: RESET Configurable to HOLD or RESET on 16-terminal basis.
Minor failure	The yellow ALM lamp comes on.	An error has occurred but the program can continue execution.	- Momentary power failure - Inter-CPU communication error - Subunit transmitter switching has occurred	Open	Shorted	Operation continues	Operation continues

* 1: The table indicates the default severity level for this error, which can be reclassified as a minor failure by configuration.
* 2: Output relays (Y□□□□□) of the advanced modules.

For some of the failure modes, you can select the **Stop** or **Run** option to specify whether to stop or continue program execution if any of these failures occur. This selection can be made using the configuration function. This configuration item defaults to the **Stop** option for a moderate failure and to the **Run** option for a minor failure. Moderate failure modes set to the **Run** option are treated as minor failure modes, while minor failure modes set to the **Stop** option are treated as moderate failure modes.

SEE ALSO

The fail signal contact reports the error state to the external environment when an error occurs.

CAUTION

If you want the contacts of an output module to be held in case of a major or moderate failure in the sequence CPU module, set the “Output When Stopped” option of the configuration function to "Hold". Note that the module action is independent of the output module type.

Table 8.2 Details on Self-diagnosis (1/3)

Failure Mode		Special Relay that Turns ON	Special Registers that Store Error Codes, Etc.	Stored Error Code	Failure Description		Corrective Actions
Major failure	CPU error	—	—	—	The CPU malfunctions due to noise or for other reasons. Hardware failure	Hardware failure	1. Check the installation environment for possible problems, such as noise sources. If the failure recurs, replace the module.*3
Moderate failure	Startup error	M193	Z017 to Z019	$10nn	A failure has occurred during CPU initialization.	Hardware failure	1. Restrictions on module installation may have been violated. Check the modules according to Section A1.2, "Restrictions on Module Installation" of the "Hardware Manual" (IM 34M06C11-01E). 2. Check the installation environment for possible problems, such as noise sources. If the failure recurs, replace the module.
	SPU error			$11nn	The CPU for sequence computing has failed.	Hardware failure	1. Check the installation environment for possible problems, such as noise sources. If the failure recurs, replace the module.
	Memory error			$1201	A program checksum error has occurred.	Transient memory failure or hardware failure (See CAUTION at the end of these tables for information on how to discriminate between these failures).	1. The error may be due to a transient memory failure caused by effects of noise. Check the installation environment. Clear the memory by referring to CAUTION at the end of these tables and download the program again. If the failure recurs, replace the module.*1
				$1202	Inadvertent writing has been done to the M129 to M131 special relays for Run, Debug and Stop mode flags.	Application error	1. Check if there is any error in the values of index registers or in the parameters defining the number of devices in an instruction, which writes to multiple devices, such as a Block Move (BMOV) instruction.
					A device memory read/write check error has occurred.	Hardware failure	1. The error may be due to a transient memory failure caused by noise. Check the installation environment. Clear the memory by referring to CAUTION at the end of these tables and download the program again. If the failure recurs, replace the module.
				$1203	A system memory read/write check error has occurred.		
	Invalid instruction found			$1701 $1702	An invalid instruction has been encountered.		
	Program error			$2001	There is a mismatch in SUB, RET or JMP instructions.	Hardware failure	1. Verify that JMP, SUB and RET instructions are paired correctly. 2. The error may be due to a transient memory failure caused by effects of noise. Check the installation environment. Clear the memory by referring to CAUTION at the end of these tables and download the program again. If the failure recurs, replace the module.
	Excess number of I/O points			$2002	The number of I/O points has been exceeded.	Hardware failure	1. Restrictions on module installation may have been violated. Check the modules according to Section A1.2, "Restrictions on Module Installation" of the "Hardware Manual" (IM 34M06C11-01E).
	ROM pack error			$8203	The ROM pack is incompatible with the CPU	Mismatch between ROM pack and CPU Hardware; Hardware failure	1. The ROM pack whose data has been erased is not defective. Use it as is. 2. Data may have been written to the ROM pack under a wrong sequence CPU type. Try writing to the ROM pack again. The ROM pack or sequence CPU module may be defective if the same failure recurs. Replace the ROM pack or the sequence CPU module.
				$8204	A failure to read from or write to the ROM pack has been encountered.	Hardware failure	1. Try writing to the ROM pack again. The ROM pack or CPU module may be defective if the same failure recurs. Replace the ROM pack or CPU module.
	Battery error	M194	—	$1801	The backup batteries have failed.	Hardware failure	1. The error may be due to a transient memory failure caused by effects of noise. Check the installation environment. The module restarts with its factory settings when it is powered on after this error is detected. Download the program again. If the same failure recurs, replace the module.
	Subroutine error*2	M201	Z022 to Z024	$2201	The subroutine return (RET) instruction was not executed or there is no return destination.	Application error	1. Check if there is a jump out of or into the subroutine. 2. Check if a scan timeout has been detected within the subroutine.
				$2202	The maximum nesting depth of eight levels has been exceeded.	Application error	1. Check the depth of nesting when calling another subroutine in a given subroutine.

*1: You may recover from this error by turning the power off and then on again.
*2: The CPU module can be configured to stop or continue execution of the program for this error event.
*3: When a major failure occurs (with RDY off), an incorrect power-off time is recorded in the system log. If a power-off and an SPU error ($1104) are recorded simultaneously, it means that the RDY has turned off due to an SPU error.

Table 8.2 Details on Self-diagnosis (2/3)

Failure Mode		Special Relay that Turns ON	Special Registers that Store Error Codes, Etc.	Stored Error Code	Failure Description		Corrective Actions
Moderate failure	Interrupt error*2	M201	Z022 to Z024	$2301	The interrupt return (IRET) instruction was not executed or there is no return destination.	Application error	1. Check if there is a jump out of or into the input interrupt program. 2. Check if a scan timeout was detected within the input interrupt program.
				$2302	There are more than eight pending interrupts.	Application error	1. There are more than eight pending interrupts. Check the detailed process of each interrupt, the number of interrupts, their frequency, etc. Check whether there are more than eight pending interrupts after powering on and before executing the program.
	Instruction error /Macro instruction error*2			$2101	A parameter is invalid.	Application error	1. Check if any abnormal value is set in the instruction parameter.
				$2102	Data is invalid.	Application error	1. Check if any abnormal value, such as one based on division by 0, is set in the instruction parameter.
				$2103	There is an error in BIN-to-BCD conversion	Application error	1. An invalid value may have been set in BIN-to-BCD conversion. Check the parameter where the error has occurred.
				$2104	There is an error in the pointers of the FIFO table.	Application error	1. Check if data written to the FIFO table has exceeded its capacity. 2. Check if an attempt has been made to read data values from the FIFO table when there is none. 3. Check if the default settings of the FIFO table are correct. Also check if the table has been corrupted by any other part of the program.
				$2105	The value defining a boundary between devices has been exceeded.	Application error	1. Check if there is any error in the values of index registers or in the parameters defining the number of devices in an instruction for writing to multiple devices, such as a Block Move (BMOV) instruction.
				$2106	The FOR-NEXT loop is not consistent.	Application error	1. Check if there is a jump out of or into the FOR-NEXT loop. 2. Check if a scan timeout has been detected within the FOR-NEXT loop.
				$2107	The IL-ILC loop is not consistent.	Application error	1. Check if there is a jump out of or into the IL-ILC loop. 2. Check if a scan timeout has been detected within the IL-ILC loop.
				$2501	The macro return (MRET) instruction was not executed or there is no return destination.	Application error	1. Check if there is a jump out of or into the macro instruction. 2. Check if a scan timeout has been detected within a macro instruction.
				$2502	Macro call nesting is deeper than 7 levels.	Application error	1. Although a macro instruction may call another macro instruction (nesting), the nesting may not be more than 7 levels deep.
	I/O comparison error*2	M202	Z027 to Z029	$2401	-The condition of module installation is not consistent with the program. -The number of special module High-speed Read (HRD) instructions or special module High-speed Write (HWR) instructions exceeded the limit (error code: $2401).	Application error	1. There may be a mismatch between the I/O relay (X/Y) devices specified in the program and those contained in the installed I/O module. Check if the instruction parameter in question is consistent with the installed I/O module. 2. Check if the number of special module High-speed Read (HRD) instructions or the number of special module High-speed Write (HWR) instructions exceeded 64.
				$2402 (READ/WRITE)	$2402 (READ/WRITE)	Application error	1. There may be a mismatch between the slot number in READ/WRITE instructions used in the program and that of the installed I/O module. Check if the instruction parameter in question is consistent with the installed I/O module.

*2: The CPU module can be configured to stop or continue execution of the program for this error event.

Table 8.2 Details on Self-diagnosis (3/3)

Failure Mode		Special Relay that Turns ON	Special Registers that Store Error Codes, Etc.	Stored Error Code	Failure Description		Corrective Actions
Moderate failure	I/O comparison error*2	M202	Z027 to Z029	$2403 (HRD/ HWR)	$2403 Special module High-speed Read instruction (HRD)/ special module-High-speed-Write instruction (HWR)	Application error	1. There may be a mismatch between the slot number in a special module High-speed Read (HDR) Instruction or a special module High-speed Write (HWR) Instruction used in the program and that of the installed I/O module. Check if the instruction parameter in question is consistent with the installed I/O module.
	I/O module error*2,*3	M203	Z033 to Z040	Failed slot number	- There is a failure to read from or write to the I/O module. - There is a communication failure in the fiber-optic FA-bus module. - An attempt has been made to reset one of the other sequence CPU modules in a multi-CPU system.	Application error	1. Check if the subunit is turned off. 2. Check if there is any problem with the cable of the fiber-optic FA-bus module. 3. Do not reset the CPU modules individually. Rather, reset them all at once from the main CPU. 4. The I/O module may be defective. Replace it.
	Scan timeout*2,*3	M204	–	$1401	The scan monitoring time has been exceeded.	Application error	1. Check if the iteration-counter values of the FOR-NEXT loop are correct. 2. Check for the presence of an endless loop caused by JMP instructions. 3. Adjust the scan monitoring time according to the execution time of the application program.
	Subunit communication error*2	M210	Z089 to Z096	Failed slot number	There is a failure to read from or write to the subunit.	Open-circuited cable Loss of power to subunit Hardware failure	1. Check if the subunit is turned off. 2. Check if there is any problem with the cable of the fiber-optic FA-bus module. 3. The fiber-optic FA-bus module may be defective. Replace it.
	Sensor CB scan timeout*2	M212	–	$1402	The CPU fails to maintain the execution interval of the sensor control block as it is exceeded by the sum of its I/O refreshing time and execution time.	Application error	1. In the case of interruption by the sensor control block after completion of instruction execution, set the execution interval at 1 ms or longer, preferably at the largest possible value. 2. Check the number of input/output words in the sensor control block and the block's execution time and minimize both values. 3. Check and minimize any code section between CBD and CBE instructions, during which execution of the sensor control block is disabled.
	Momentary power failure	M195	–	$1302	The CPU indicates that a momentary power failure has occurred.	Momentary power failure	1. If this failure mode occurs too frequently, check the power supply for possible problems. If a UPS is in use, check that it has captured peak values of its supply voltage waveform. If the failure still occurs frequently while there is no problem with the wave-form, the power supply module or sequence CPU module may be defective. Replace it.
Minor failure	Inter-CPU communication error	M196	–	$400n	There is a communication failure in shared devices.	Hardware failure	1. There may be a failure in one of the other CPUs in a multi-CPU system. Do not reset the CPU modules individually. Rather, reset them all at once from the main CPU. If this failure mode recurs, replace the CPU modules.
	Subunit transmitter switching has occurred	M211	Z089 to Z096	Failed slot number	There is a problem with the paired cables attached to remote I/O modules in a loop configuration.	Open-circuited cable	1. Check if there is any problem with the cable of the fiber-optic FA-bus module. 2. The fiber-optic FA-bus module may be defective. Replace it.

*2: The CPU module can be configured to stop or continue execution of the program for this error event.

*3: Since debugging operation in Debug mode may extend the scan time, the scan monitoring time increases threefold.

SEE ALSO

For error log (system log) messages, see also "FA-M3 Programming Tool WideField3" (IM 34M06Q16-□□E) or "FA-M3 Programming Tool WideField2" (IM 34M06Q15-01E).

For details on the failure modes "subunit communication error (M210)" and "Subunit transmitter switching has occurred (M211)", see "Fiber-optic FA-bus Module and Fiber-optic FA-bus Type 2 Module, FA-bus Type 2 Module" (IM 34M06H45-01E).

CAUTION

You can clear the CPU memory and revert it to its factory settings by installing the sequence CPU module in the 5th or higher slot of the main unit and turning on the power. After confirming that the RDY lamp is lit, you may then turn off the power.

In the case of a transient memory error due to noise, download the application program again. If the error disappears, you can continue to use the module. If the error recurs, there may be a hardware failure. In this case, replace the sequence CPU module.

8.1.1 Setting Error-time Action (Operating Mode in Case of Error)

■ Setting Error-time Action (Operating Mode in Case of Error)

Using the configuration function, you can select "**Stop**" (stop execution for moderate failure) or "**Run**" (continue execution for minor failure) for some faults such as an instruction error. The table below summarizes the configuration items and their defaults.

If a failure for which you have selected the **Run** option actually occurs, the CPU fails to correctly perform such tasks as accessing the I/O module that caused the failure or processing instructions.

Configuration Item	Default
I/O module error	Stop (moderate failure)
I/O comparison error	
Instruction error	
Scan timeout	
Subroutine error	
Interrupt error	
Sensor CB scan timeout	
Subunit communication error	Run (minor failure)

SEE ALSO

For details on the failure mode “subunit communication error,” see "Fiber-optic FA-bus Module and Fiber-optic FA-bus Type 2 Module, FA-bus Type 2 Module" (IM 34M06H45-01E).

8.2 Updating Error Status Indicators after Correcting Moderate or Minor Failures

After eliminating the cause of a moderate or minor failure, initialize the states of special relays (M), special registers (Z) and LED indicators as instructed below.

■ Updating Error Status after Removing Moderate Failures

To reset special relays (M) and special registers (Z), as well as turn off the ERR LED indicator after eliminating the cause of a moderate failure, use the following procedure.

(1) Turn on the power again.

(2) Switch the sequence CPU to **Run** or **Debug** mode using WideField3 (or WideField2).

■ Updating Error Status after Removing Minor Failures

To reset special relays (M) and special registers (Z), as well as turn off the ALM LED indicator after eliminating the cause of a minor failure, use the following procedure.

(1) Turn on the power again.

(2) Switch the sequence CPU to **Run** or **Debug** mode using WideField3 (or WideField2).

(3) Perform “Cancel Alarm.”

9. Differences from F3SP25 and F3SP35 Sequence CPUs

This chapter describes the differences from F3SP25 and F3SP35 sequence CPUs. You must read this chapter before replacing sequence CPU module F3SP25 or F3SP35 with sequence CPU module F3SP22, F3SP28, F3SP38, F3SP53, or F3SP58 –□N/□H.

9.1 Comparison of Functional Specifications

Item	Specification						
	F3SP25	**F3SP35**	**F3SP28-3N F3SP53-4H**	**F3SP22-0S F3SP28-3S F3SP53-4S**	**F3SP38-6N F3SP58-6H**	**F3SP38-6S F3SP58-6S**	**F3SP59-7S**
Number of I/O points	4096 max.	8192 max., including remote I/O points	4096 max., including remote I/O points		8192 max., including remote I/O points		8192 max., including remote I/O points
Number of internal relays (I)	8192	16384	16384		32768		65535
Number of link relays (L)	8192	8192	8192		16384		16384
Number of timers (T)	1024	2048	1024		2048		2048
Number of data registers (D)	8192	8192	16384		32768		65535
Number of file registers (B)	32768	32768	32768		262144		262144
Number of link registers (W)	8192	8192	8192		16384		16384
Program size	20K steps max.	100K steps max.	10K steps max (F3SP22) 30K steps max (F3SP28) 56K steps max. (F3SP53)		120K steps max.		254K steps max.
Number of program blocks	128 max.	1024 max.	1024 max.		1024 max.		1024 max.
Number of program macros	128 max.	64 max.	64 max.	256 max.	64 max.	256 max.	256 max
Number of basis instructions	25	25	33		33		33
Number of application instructions	307	307	312	328	312	328	328
Other functions	—	—	Sensor control functions		Sensor control functions		Sensor control functions

Item		Specification			
		F3SP25	**F3SP35**	**F3SP22 F3SP28 F3SP38**	**F3SP53 F3SP58 F3SP59**
Instruction execution time	Basic instruction	0.12 to 0.24 μs per instruction	0.09 to 0.18 μs per instruction	0.045 to 0.18 μs per instruction	0.0175 to 0.07 μs per instruction
	Application instruction	0.24 μs min. per instruction	0.18 μs min. per instruction	0.18 μs min. per instruction	0.07 μs min. per instruction

TIP

To gain access to file register (B) using a personal computer link command, refer to the conventions given below.

Module	Restriction	Convention
Personal computer link (F3LC11-1N, F3LC11-2N)	Yes	Accessible file registers are B1 to B99999.
Personal computer link (F3LC11-1F, F3LC12-1F)	No	The device name of file registers (B) must be seven bytes long.
Ethernet interface		
CPU (personal computer link functions)		

- To gain access to the sensor control block using personal computer link module, refer to the conventions given below.

Module	Restriction	Convention
Personal computer link (F3LC11-1N, F3LC11-2N)	Yes	A sensor control block number is determined by adding “1” to the end of a number assigned to regular blocks. You cannot access the blocks numbered from 100 to 1024.
Personal computer link (F3LC11-1F, F3LC12-1F)	No	A sensor control block number is determined by adding “1” to the end of a number assigned to regular blocks.
Ethernet interface	Yes	A sensor control block number is determined by adding “1” to the end of a number assigned to regular blocks. You cannot access the blocks numbered from 100 to 1024.
CPU (personal computer link functions)	Yes	A sensor control block number is determined by adding “1” to the end of a number assigned to regular blocks. You cannot access the blocks numbered from 100 to 1024.

9.2 Configuration

Item		Specification		
		F3SP25	F3SP35	F3SP22, F3SP28, F3SP38, F3SP53, F3SP58, F3SP59
Device capacities	Internal relay (I)	8192 units for both types of relays combined	16384 units for both types of relays combined	No configuration is required for these types of relays.
	Shared Relay (E)			
	Data register (D)	8192 units for both types of registers combined.		No configuration is required for these types of registers.
	Shared register (R)			
	100-µs timer	No		16 max
	1-ms timer	16 max.		F3SP22, F3SP28 and F3SP53: 2048 max. (for all timers combined) F3SP38, F3SP58, F3SP59: 3072 max. (for all timers combined) .
Error-time action (Operating mode in case of error)	Sensor CB scan timeout	No		Yes
Input/output setup	Terminal usage (Module used/Not used)	Use/Not used Configurable on a module basis.		Configurable on 16-point basis, including Use/Not used/Use with Sensor CB (Note) .
	Input sampling interval	16ms/1ms; configurable on a module basis.		16 ms/1 ms/250 µs/62.5 µs/Always; configurable on 16-point basis.
	Output when stopped (Reset/hold external outputs when sequence stops)	Configurable on a module basis.		Configurable on 16 points.
Sensor CB	Execution interval	No		Configurable from 200 µs to 25.0ms in 100 µs increments.
	Timing of interrupt	No		After instruction execution /Immediate, during instruction execution.
Input interrupt	Timing of interrupt	No (always after instruction execution)		After instruction execution / Immediate, during instruction execution
Priority of interrupts		No		Sensor CB interrupt has priority / input interrupt has priority
Peripheral processing time		No		Configurable from 100 µs to 190 ms in 100µs increments.
Shared refreshing (inter-CPU-module communications method)	Shared refreshing range (partial disabling of shared refreshing)	No		Run/Stop; configurable for shared relays (E), shared registers (R), extended shared relays (E), and extended shared registers (R) of each CPU module.
	Simultaneity of shared refreshed data	No (No simultaneity in all cases)		Yes/No
	Shared refreshing mode (defined as control-related process)	No		Peripheral process/ Control process
FL-net setup	Common data refreshing mode	No		Peripheral process / Control process
	Common data refreshing range	No		All nodes / Some nodes

Note: Sensor CB = Sensor control block

CAUTION

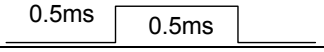
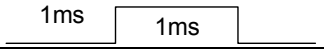
For F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 and F3SP59, you can determine by configuration (Yes/No options) whether there is simultaneity with the data of shared devices. The “No” option of this configuration item is designed for the interchangeability of the F3SP21, F3SP25 and F3SP35 modules. Select this option when replacing these modules with the F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 or F3SP59 modules.

CAUTION

If your sequence CPU is F3SP22, F3SP28, F3SP38, F3SP53, F3SP58, or F3SP59, set the CPU’s output relays (Y) to be refreshed to the option “Unused, ” by a direct Refresh (DREF) instruction in a program. If you set them to the option “Used” or “Used in CB (sensor control block) , ” the values one scan earlier may be overwritten with the values output by the DREF instruction because of the timing of output refreshing that is executed concurrently with the instruction.

9.3 Special Relays (M) and Special Registers (Z)

The following special relays (M) have been added to the list of utility relays.

Item	Utility		
No.	Name	Function	Description
M047	1-ms clock		Generates a clock pulse with a 1-ms period.
M048	2-ms clock		Generates a clock pulse with a 2-ms period.
M097	On for one scan at Sensor CB start	ON : When the block is activated. OFF: In all other cases	Turns on for one scan when the sensor control block starts (at the first execution of the sensor control block) .

Item	Utility		
No.	Name	Function	Description
M137	Sensor CB execution status	ON : Run OFF: Stop	Indicates the status of a sensor control block

Item	Utility		
No	Name	Function	Description
M212	Sensor CB scan timeout	ON : Error OFF: Normal	The CPU fails to sustain the execution interval of the sensor control block

Item	Utility		
No.	Name	Function	Description
M3521 to M3774	Node Participation Status	1: Participating 0: Not participating	FL-net system 1*1
M3777 to M4030	Upper Layer Operation Signal Error	1: Error 0: Normal	FL-net system 1*1
M4033 to M4286	Operation Status	1: Run 0: Stop	FL-net system 1*1
M4289 to M4542	Common Memory Data Valid	1: Valid 0: Invalid	FL-net system 1*1
M4561 to M4814	Node Participation Status	1: Participating 0: Not participating	FL-net system 2*2
M4817 to M5070	Upper Layer Operation Signal Error	1: Error 0: Normal	FL-net system 2*2
M5073 to M5326	Operation Status	1: Run 0: Stop	FL-net system 2*2
M5329 to M5582	Common Memory Data Valid	1: Valid 0: Invalid	FL-net system 2*2

*1: If both FL-net and FA link are installed, FL-net systems are allocated smaller system numbers.
*2: If both FL-net and FA ink are installed, FL-net systems are allocated larger system numbers.

SEE ALSO

For details, see "FL-net (OPCN-2) Interface Module" (IM 34M06H32-02E).

The following special registers (Z) have been added to the list of utility registers.

Item	Utility	
No.	Name	Description
Z109	Sensor CB execution time	Refers to the length of time from when input refreshing is started for the sensor control block to when the program is executed and output refreshing is completed. (Unit: 10 μs)
Z111	Maximum Sensor CB execution time	Refers to the maximum time taken to execute the sensor control block.(Unit:10 μs)

9.4 CPU Module to CPU Module Communication Method

● Data Sharing

The availability of the simultaneity of data in CPU-to-CPU communication between shared relays (E) and shared registers (R) and between extended shared relays (E) and extended shared registers (R) is as shown in the following table.

Availability of Simultaneity of Data among CPU Modules		CPU B	
		F3SP25 F3SP35	F3SP22 F3SP28 F3SP38 F3SP53 F3SP58 F3SP59
CPUA	F3SP25 F3SP35	×	×
	F3SP22 F3SP28 F3SP38 F3SP53 F3SP58 F3SP59	×	✓

There is no simultaneity of data in CPU module to CPU module communication between shared relays (E) /registers (R) and extended shared relays (E) /registers (R).

● Stopping Shared Refreshing Partially

You can exclude a particular CPU or CPUs from shared refreshing.

● Defining Shared Refreshing as a Control-related Process

You can determine whether shared refreshing is performed as a control-related process or a peripheral process.

9.5 High-speed Processing of Application Instructions

The F3SP25 and F3SP35 sequence CPUs do not support high -speed processing for Application Instructions that use any of the devices listed below.

In contrast, the F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 and F3SP59 sequence CPUs support the high-speed processing of Application Instructions where these devices are used.

- Extended shared relays (E)
- Extended shared registers (R)
- Link registers for FA link system 3 or later (W)
- File registers (B)

9.6 Instructions

The following instructions have been added to the list of instructions available with the F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 and F3SP59 sequence CPU modules.

Instruction Name	Description
LDU	Load Differential Up
LDD	Load Differential Down
UP	Logical Differential Up
DWN	Logical Differential Down
UPX	Logical Differential Up Using Specified Device
DWNX	Logical Differential Down Using Specified Device
INV	Inverter
FF	Flip-Flop
CBACT	Activate Sensor Control Block
CBINA	Inactivate Sensor Control Block
CBD	Disable Sensor Control Block
CBE	Enable Sensor Control Block
FTIMR	Read Free Run Timer

10. Difference between F3SP□□-□S and F3SP□□-□N/-□H

This chapter describes the difference in function and specifications between the F3SP28/38/53/58-□N/□H (old models) and the F3SP28/38/53/58/59-□S (new models) CPU modules.

10.1 Partial Download Functions

The new models are provided with partial download functions, which allow only specified blocks or macros to be downloaded to a CPU, overwriting corresponding blocks or macros of a program that has been downloaded earlier.

SEE ALSO

For details on the partial download functions, see Section 6.16, "Partial Download Functions."

10.2 Storing Comments or Tag Name Definitions in CPU

The new models allow circuit comments/subcomments or tag name definitions to be stored in the memory of a CPU module or the ROM pack.

SEE ALSO

- For details on the download comments function, see Section 6.17, "Functions for Storing Comments to CPU."
- For details on the download tag name definitions function, see Section 6.18, "Functions for Storing Tag Name Definitions to CPU."
- For details on the ROM pack function, see Section 6.8.3, "ROM Writer Functions and ROM Writer Mode."

10.3 New Instructions and Instruction Related Functions

- Structures

  The new models introduce a new concept of structures to handle device addresses. Some instructions are added to handle structures.

SEE ALSO

For details on structures, see "FA-M3 Programming Tool WideField3" (IM 34M06Q16-☐☐E) or "FA-M3 Programming Tool WideField2" (IM 34M06Q15-01E).

- Indirect specification

  With the new models, indirect specification is available to address devices. Thus some instructions are added to handle this new function.

  Using indirect specification allows you to address all file registers (B), including those which cannot be accessed using index modification.

  You can also use indirect specification to address devices other than the file registers (B).

SEE ALSO

For details on indirect specification, see Section 1.10.2, "Indirect Specification," of "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

- Index modification by constant

  With the new models, index modification by constant is available.

SEE ALSO

For details on index modification by constant, see Section 1.10.1, "Index Modification," of "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

- Nesting of interlock areas (program steps between IL and ILC)

  Interlock areas may be nested up to 8 levels.

SEE ALSO

For details on interlock, see Section 2.18, "Interlock (IL), Interlock Clear (ILC)," of "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

- New instructions

  The following instructions are added in the new models.

Table 10.1 List of Added Instructions

Mnemonic	Instruction
SET@	Indirect Address Set
ADD@	Indirect Address Add
MOV@	Indirect Address Move
LDW	Load Specified Bit
OUTW	Out Specified Bit
SETW	Set Specified Bit
RSTW	Reset Specified Bit
NCALL	Input Macro Instruction Call
NMOUT	Output of Input Macro
SCALL	Structure Macro Instruction Call
STMOV	Structure Move
STRCT	Structure pointer Declaration
DATE	Set Date
TIME	Set Time
SDATE	Set Date String
STIME	Set Time String

SEE ALSO

For details on these new instructions, see "Sequence CPU Instruction Manual – Instructions" (IM 34M06P12-03E).

10.4 Changes in Specifications

Table 10.2 List of Changes in Specifications

Items	CPU Types	
	F3SP28-3N F3SP38-6N F3SP53-4H F3SP58-6H	**F3SP28-3S F3SP38-6S F3SP53-4S F3SP58-6S F3SP59-7S**
Number of steps per block	10K steps	56K steps
Number of macro instructions	64	256
Number of circuits/subcomments	3000 per program	30000 per block
Number of timer steps	2	4
Compatible ROM pack	RK33/RK73	RK33/RK73/RK93*1

*1: RK93 is not compatible with the F3SP28-3S and F3SP53-4S.

Blank Page

FA-M3

Sequence CPU Instruction Manual - Functions

(for F3SP22-0S, F3SP28-3N/3S, F3SP38-6N/6S, F3SP53-4H/4S, F3SP58-6H/6S, F3SP59-7S)

IM 34M06P13-01E 4th Edition

These appendices provide lists of special devices, as well as formats of documentation which can be used when designing your system. These formatted sheets can be conveniently copied for use as standard forms in your system design.

Four forms are provided, as shown below.

Contents

Blank Page

Appendix 1. Special Relays (M)

Special relays have specific functions, such as indicating the internal state of a sequence CPU module or detecting errors.

In programs, these relays are used mainly for contacts **a** and **b**.

Appendix 1.1 Block Start Status Relays

Block Start Status relays indicate which block is running when selected blocks are being executed.

These relays are numbered in ascending order as M001, M002, …, to correlate with block 1, block 2, ...

Table Appendix 1.1 Block Start Status Relays

Item		Block Start Status Relays		
Sequence CPU Module	**No.**	**Name**	**Function**	**Description**
F3SP05, F3SP08, F3SP21	M0001 to M0032	Block n start status	ON : Run OFF: Stop	Indicates whether block n is executed when the module is configured to execute specified blocks only.
F3SP22, F3SP25, F3SP35	M0001 to M0032			
F3SP28, F3SP38 F3SP53, F3SP58, F3SP59	M2001 to M3024			

Note: The Start Status relays assigned to blocks 1 to 32 are M0001 to M0032 and M2001 to M2032 (M0001 to M0032 have the same values as M2001 to M2032. Similarly, Start Status relays M2033 to M3024 map to blocks 33 to 1024.

CAUTION

Do not write to a special relay, including those not listed in tables in this section (e.g., M067 to M096), unless otherwise stated. Special relays are used by the sequence CPU module. Writing to these relays incorrectly may lead to system shutdown or other failures. Using forced set/reset instruction in debug mode is also prohibited.

CAUTION

Special relays with index modification cannot be specified as destinations for data output and if specified, will result in instruction processing errors during execution.

CAUTION

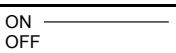
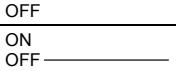
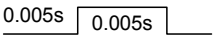
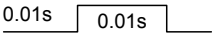
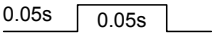
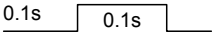
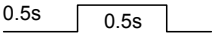
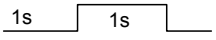
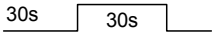
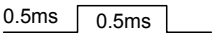
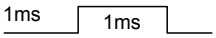
Special relays cannot be specified as output destinations in block transfer and table output ladder instructions, and if specified, will cause instruction processing errors during execution.

- **Block transfer instructions: BMOV, BSET, SMOV, etc.**
- **Table output instructions: ULOGR, FIFWR, etc.**

Appendix 1.2 Utility Relays

Utility relays are used to provide timing in a program or issue instructions to the CPU module.

Table Appendix 1.2 Utility Relays

Item	Utility Relays		
No.	**Name**	**Function**	**Description**
M033	Always ON		Used for initialization or as a dummy contact in a program.
M034	Always OFF		
M035	1 Scan ON at Program Start		Turns on for one scan only after a program starts execution.
M036*	0.01 s Clock		Generates a clock pulse with a 0.01s period.
M037*	0.02 s Clock		Generates a clock pulse with a 0.02 s period.
M038*	0.1 s Clock		Generates a clock pulse with a 0.1 s period.
M039*	0.2 s Clock		Generates a clock pulse with a 0.2 s period.
M040*	1 s Clock		Generates a clock pulse with a 1 s period.
M041*	2 s Clock		Generates a clock pulse with a 2 s period.
M042*	1 min Clock		Generates a clock pulse with a 1 min period.
M047*	1 ms Clock		Generates a clock pulse with a 1 ms period.
M048*	2 ms Clock		Generates a clock pulse with a 2 ms period.
M066	Normal Subunit Transmission Line	ON : Normal transmission line or no fiber-optic FA-bus installed OFF: Unspecified or abnormal transmission line	
M097	ON for One Scan at Sensor CB Start	ON : At startup OFF: In all other cases.	Turns on for one scan when the sensor control block starts (at the first execution of the sensor control block).

*: Blocks M036 to M048 have their rising and falling clock timing synchronized.

SEE ALSO

For details on the M066 Utility relay (Normal Subunit Transmission Line), see "Fiber-optic FA-bus Module and Fiber-optic FA-bus Type 2 Module, FA-bus Type 2 Module" (IM 34M06H45-01E).

Appendix 1.3 Sequence Operation and Mode Status Relays

Sequence operation and mode status relays indicate the status of sequence operation and various modes.

Table Appendix 1.3 Sequence Operation and Mode Status Relays

Item	Sequence Operation and Mode Status Relays		
No.	**Name**	**Function**	**Description**
M129	Run Mode Flag	ON : Run mode OFF: Other modes	Indicates the status of CPU operation.
M130	Debug Mode Flag	ON : Debug mode OFF: Other modes	Indicates the status of CPU operation.
M131	Stop Mode Flag	ON : Stop mode OFF: Other modes	Indicates the status of CPU operation.
M132	Pause Flag	ON : Pause OFF: Run	Indicates the status of program execution during debug mode operation.
M133	Execution Flag	ON : Specified blocks OFF: All blocks	Indicates whether all blocks or specified blocks are executed.
M135	RAM/ROM-based Operation Flag	ON : ROM-based operation OFF: RAM-based operation	Indicates whether operation is based on the ROM or RAM.
M136*	Power-on Operation Flag	ON : Power-on operation OFF: Other modes of operation	Indicates whether operation was initiated by power on or reset
M137*	Sensor CB Execution Status	ON : Run OFF: Stop	Indicates the status of sensor control block execution.
M172 (write-enabled)	Set Clock Time	ON : Time being set OFF:	Requests to set clock data.
M173	Input-offline Flag	ON : Offline OFF: Online	Indicates that input refreshing has stopped.
M174	Output-offline Flag	ON : Offline OFF: Online	Indicates that output refreshing has stopped.
M175	Shared-I/O-offline Flag	ON : Offline OFF: Online	Indicates that shared refreshing has stopped.
M176	Link-I/O-offline Flag	ON : Offline OFF: Online	Indicates that link refreshing has stopped.
M177 to M187	Devices Reserved for Extended Functions		
M188	Carry Flag	ON : Carry enabled OFF: Carry disabled	Carry flag used by shift and rotate operations
M189 to M192	Devices Reserved for Extended Functions		

*: For F3SP21, F3SP22, F3SP25, F3SP28, F3SP35, F3SP38, F3SP53 and F3SP58 only.

SEE ALSO

For details on clock setup, see the specifications of special registers (Z) for clock data.

Appendix 1.4 Self-diagnosis Status Relays

Self-diagnosis status relays indicate the results of self-diagnosis by the sequence CPU.

Table Appendix 1.4 Self-diagnosis Status Relays

Item	Self-diagnosis Status Relays		
No.	**Name**	**Function**	**Description**
M193	Self-diagnosis Error	ON : Error OFF: No error	Result of self diagnosis is stored in special registers Z17 to Z19
M194	Battery Error	ON : Error OFF: Normal	Indicates a failure in backup batteries.
M195	Momentary Power Failure	ON : Momentary power failure OFF: No momentary power failure	Indicates that a momentary power failure has occurred.
M196	Inter-CPU Communication Error	ON : Error OFF: Normal	Indicates that a communication failure has occurred in shared relays (E) or shared registers (R).
M197	Existence of CPU1	ON : Exists. OFF: Does not exist.	Indicates whether or not a CPU exists in slot 1.
M198	Existence of CPU2	ON : Exists. OFF: Does not exist.	Indicates whether or not a CPU exists in slot 2.
M199	Existence of CPU3	ON : Exists. OFF: Does not exist.	Indicates whether or not a CPU exists in slot 3.
M200	Existence of CPU4	ON : Exists. OFF: Does not exist.	Indicates whether or not a CPU exists in slot 4.
M201	Instruction Processing Error	ON : Error OFF: Normal	Information of instruction processing error is stored in special registers Z22 to Z24.
M202	I/O Comparison Error	ON : Error OFF: Normal	Indicates that the state of module installation is not consistent with the program.
M203	I/O Module Error	ON : Error OFF: Normal	Indicates that no access is possible to I/O modules. The slot number of the error module is stored in special registers Z33 to Z40.
M204	Scan Timeout	ON : Error OFF: Normal	Indicates that scan time has exceeded the scan monitoring time.
M210	Subunit Communication Error	ON : Error OFF: Unspecified or normal line	An error has been detected in the fiber-optic FA-bus module. The slot number of the error module is stored in special registers Z89 to Z96.
M211	Subunit Transmitter Switching Has Occurred	ON : Error OFF: Unspecified or normal line	
M212	Sensor CB Scan Timeout	ON : Error OFF: Normal	Indicates that the execution interval of the sensor control block cannot be maintained.
M225*	CPU1 Sequence Program Execution	ON : Executes the program. OFF: Stops the program.	Indicates whether sequence program of CPU in slot 1 is running.
M226*	CPU2 Sequence Program Execution	ON : Executes the program. OFF: Stops the program.	Indicates whether sequence program of CPU in slot 2 is running.
M227*	CPU3 Sequence Program Execution	ON : Executes the program. OFF: Stops the program.	Indicates whether sequence program of CPU in slot 3 is running.
M228*	CPU4 Sequence Program Execution	ON : Executes the program. OFF: Stops the program.	Indicates whether sequence program of CPU in slot 4 is running.

*: For F3SP21, F3SP22, F3SP25, F3SP28, F3SP35, F3SP38, F3SP53 and F3SP58 only.

SEE ALSO

For details on the M210 (Subunit Communication Error) and M211 (Subunit Transmitter Switching Has Occurred) self-diagnosis relays, see “Fiber-optic FA-bus Module and Fiber-optic FA-bus Type 2 Module, FA-bus Type 2 Module” (IM 34M06H45-01E).

Appendix 1.5 FA Link Module Status Relays

FA Link module status relays indicate the status of FA link.

SEE ALSO

For details on FA link module status relays, see the sections on special relays and special registers of “FA Link H Module, Fiber-optic FA Link H Module” (IM 34M06H43-01E).

Table Appendix 1.5 FA Link Module Status Relays

Item	FA Link Module Status Relays		
No.	Name	Function	Description
M257 to M480 M8321 to M8992	FA Link Error	ON : Error OFF: Normal	Indicates the status of FA links.

Appendix 1.6 FL-net Interface Module Status Relays

FL-net interface module status relays indicate the status of FL-net.

Table Appendix 1.6 FL-net Interface Module Status Relays

Item	FL-net Interface Module Status Relay		
No.	Name	Function	Description
M3521 to M3774	Node Participation Status	1: Participating 0: Not participating	FL-net system 1 *1
M3777 to M4030	Upper Layer Operation Signal Error	1: Error 0: Normal	FL-net system 1 *1
M4033 to M4286	Operation Status	1: Run 0: Stop	FL-net system 1 *1
M4289 to M4542	Common Memory Data Valid	1: Valid 0: Invalid	FL-net system 1 *1
M4561 to M4814	Node Participation Status	1: Participating 0: Not participating	FL-net system 2 *2
M4817 to M5070	Upper Layer Operation Signal Error	1: Error 0: Normal	FL-net system 2 *2
M5073 to M5326	Operation Status	1: Run 0: Stop	FL-net system 2 *2
M5329 to M5582	Common Memory Data Valid	1: Valid 0: Invalid	FL-net system 2 *2

*1: If both FL-net and FA link are installed, FL-net are allocated smaller system numbers.
*2: If both FL-net and FA ink are installed, FL-net are allocated larger system numbers.

SEE ALSO

For details, see "FL-net (OPCN-2) Interface Module" (IM 34M06H32-02E).

TIP

A system refers to a group of units connected to one FL-net.

Blank Page

Appendix 2. Special Registers (Z)

Special registers have specific functions, such as indicating the internal state of a programmable controller or detecting errors.

Appendix 2.1 Sequence Operation Status Registers

Sequence operation status registers indicate the status of sequence operation.

Table Appendix 2.1 Sequence Operation Status Registers

Item	Sequence Operation Status Registers		
No.	**Name**	**Function**	**Description**
Z001	Scan Time (Run mode)	Latest scan time	Stores the latest scan time in 100 µs increments.
Z002	Minimum Scan time (Run mode)	Minimum scan time	Allows the latest scan time to be read in 100 µs increments if it is shorter than the minimum scan time.
Z003	Maximum Scan Time (Run mode)	Maximum scan time	Allows the latest scan time to be read in 100 µs increments if it is longer than the maximum scan time.
Z004	Scan Time (Debug mode)	Latest scan time	Stores the latest scan time in 100 µs increments.
Z005	Minimum Scan Time (Debug mode)	Minimum scan time	Allows the latest scan time to be read in 100 µs increments if it is shorter than the minimum scan time.
Z006	Maximum Scan Time (Debug mode)	Maximum scan time	Allows the latest scan time to be read in 100 µs increments if it is longer than the maximum scan time.
Z007	Peripheral-process Scan Time	Latest scan time	Stores the latest scan time in 100 µs increments. (Tolerance: Scan time of one control process)
Z008	Minimum Peripheral-process Scan Time	Minimum scan time	Allows the latest scan time to be read in 100 µs increments if it is shorter than the minimum scan time. (Tolerance: Scan time of one control process)
Z009	Maximum Peripheral-process Scan Time	Maximum scan time	Allows the latest scan time to be read in 100 µs increments if it is longer than the maximum scan time. (Tolerance: Scan time of one control process)

CAUTION

- Do not write to a special register (Z), including those not listed in the table above (e.g., Z010 to Z016), unless otherwise stated. This is because they are used by the sequence CPU module for the system. If you inadvertently write to these registers, a failure, such as a system shutdown, may result.
- You are not allowed to apply index modification to special registers (Z) in an attempt to specify them as the destination of data output. If you do so, an instruction processing error will result.
- In a ladder instruction for continuous data transfer or table-format data output (see examples below), you are not allowed to specify a special register (Z) as the output destination. If you do so, an instruction processing error will result.

  Instructions for continuous data transfer: Block Move instruction (BMOV instruction), Block Set instruction (BSET instruction), String Move instruction (SMOV instruction), etc.

  Instructions for table-format data output: User Log Read instruction (ULOGR instruction), FIFO Write instruction (FIFWR instruction), etc.

Appendix 2.2 Self-diagnosis Status Registers

Self-diagnosis status registers indicate the results of self-diagnostics by the sequence CPU.

Table Appendix 2.2 Self-diagnosis Status Registers

Type	Self-diagnosis Status Registers		
No.	**Name**	**Function**	**Description**
Z017	Self-diagnosis Error	Self-diagnosis error No.	Store the results of self-diagnosis.*
Z018		Self-diagnosis error block No.	
Z019		Self-diagnosis error instruction No.	
Z022	Instruction Processing Error	Instruction processing error No.	Store errors occurring during instruction processing.*
Z023		Instruction processing error block No.	
Z024		Instruction processing error instruction No.	
Z027	I/O Comparison Error	I/O comparison error No.	Store detailed information on I/O comparison errors.*
Z028		I/O comparison error block No.	
Z029		I/O comparison error instruction No.	
Z033 to 040	I/O Error	Slot no. with I/O error 16 ····· 2 1 0 ····· 1 0	Store, as a bit pattern, the slot number for which an I/O error has occurred. Z033: Main unit Z034: Subunit 1 Z035: Subunit 2 Z036: Subunit 3 Z037: Subunit 4 Z038: Subunit 5 Z039: Subunit 6 Z040: Subunit 7
Z041	Module Recognition	Main unit	Slot number 16 ····· 1 0 ····· 1 0 0: No modules are recognized. Unable to read/write. 1: Modules are recognized.
Z042		Subunit 1	
Z043		Subunit 2	
Z044		Subunit 3	
Z045		Subunit 4	
Z046		Subunit 5	
Z047		Subunit 6	
Z048		Subunit 7	
Z089	Subunit Communication Error Slot	Main unit	Slot number 16 ····· 1 0 ····· 1 0 Fiber-optic FA-bus module 0: Normal transmission line; Unspecified transmission line; or Loaded with a wrong module 1: Abnormal transmission line (“Subunit communication error” or “Sub unit transmitter switching has occurred)
Z090		Subunit 1	
Z091		Subunit 2	
Z092		Subunit 3	
Z093		Subunit 4	
Z094		Subunit 5	
Z095		Subunit 6	
Z096		Subunit 7	

*: For information on error numbers (codes) to be saved in these special registers, see Table 8.2, "Details of Self-diagnosis."

SEE ALSO

For more information on the Z089 to Z096 special registers (Subunit Communication Error Slot), see “Fiber-optic FA-bus Module and Fiber-optic FA-bus Type 2 Module, FA-bus Type 2 Module” (IM 34M06H45-01E).

Appendix 2.3 Utility Registers

Table Appendix 2.3 Utility Registers

Type	Utility Registers		
No.	**Name**	**Function**	**Description**
Z049 (write-enabled)	Clock Data	Last two digits of calendar year	Stores "year" as a BCD-coded value. e.g. 1999 as $0099 2000 as $0000
Z050 (write-enabled)		Month	Stores "month" as a BCD-coded value. e.g. January as $0001
Z051 (write-enabled)		Day of month	Stores "day of month" as a BCD-coded value. e.g. 28th as $0028
Z052 (write-enabled)		Hour	Stores "hour" as a BCD-coded value. e.g. 18:00 hours as $0018
Z053 (write-enabled)		Minute	Stores "minute" as a BCD-coded value. e.g. 15 minutes as $0015
Z054 (write-enabled)		Second	Stores "second" as a BCD-coded value. e.g. 30 seconds as $0030
Z055		Day of week ($0000 to $0006)	Stores "day of week" as a BCD-coded value. e.g. Wednesday as $0003
Z056*	Constant Scan Time	Value of constant scan time	0.1 ms increments e.g. 10 ms as 100
Z057*	Constant Scan Time	Value of constant scan time	1 ms increments e.g. 10 ms as 10
Z058	Scan Monitoring Time	Value of scan monitoring time	1 ms increments e.g. 200 ms as 200

*: Available with the F3SP22, F3SP28, F3SP38, F3SP58 and F3SP59 only.

● Setting Clock Data

- For CPU module F3SP□□-□S, use Set Date instruction (DATE), Set Time instruction (TIME), Set Date String instruction (SDATE), and Set Time String instruction (STIME) to set clock data.
- For CPU module F3SP□□-□N/-□H, follow the procedure given below to set clock data.

  (1) Write the clock data to special registers Z049 to Z054 (use a MOV P instruction).

  (2) Set special relay M172 to ON within the same scan as that in step (1) (use a DIFU instruction).

  (3) Set special relay M172 to OFF in the scan subsequent to that in step (2).

  Also stop writing the clock data to special registers Z049 to Z054 in that scan.

- Note that no change is made to the clock data and the data reverts to its original values if the values being set are incorrect.

● Clock Data Accuracy

- The accuracy of clock data is specified as:
- Maximum monthly error = ±8 s (±2 s, when actually measured)

The clock accuracy is reset to the maximum daily error of -1.2 s/+2 s, however, when the power is turned off and on again. In addition, it is possible to input a corrective value from the programming tool. If you input a precise corrective value, the clock data is corrected during the power-off-and-on sequence, thus offsetting the cumulative amount of error.

Appendix 2.4 FA Link Module Status Registers

FA Link module status registers indicate the status of FA link.

SEE ALSO

For details on the FA link module status registers, see Special relays/registers sections in "FA Link H Module Fiber-optic FA Link H Module" (IM 34M06H43-01E).

Table Appendix 2.4 FA Link Module Status Registers

Type	FA Link Module Status Registers		
No.	Name	Function	Description
Z075	Local Station No.		System 1 (FA link)
Z076	Local Station No.		System 2 (FA link)
Z077	Local Station No.		System 3 (FA link)
Z078	Local Station No.		System 4 (FA link)
Z079	Local Station No.		System 5 (FA link)
Z080	Local Station No.		System 6 (FA link)
Z081	Local Station No.		System 7 (FA link)
Z082	Local Station No.		System 8 (FA link)
Z065	Local Station Status	0: Initialization in progress 1: Offline 2: Online	System 1 (FA Link)
Z066	Cyclic Transmission Time		System 1 (FA Link) 1ms increments
Z070	Local Station Status	0: Initialization in progress 1: Offline 2: Online	System 2 (FA Link)
Z071	Cyclic Transmission Time		System 2 (FA Link) 1ms increments
Z257*	Local Station Status	0: Initialization in progress 1: Offline 2: Online	System 3 (FA Link)
Z258*	Cyclic Transmission Time		System 3 (FA Link) 1ms increments
Z262*	Local station status	0: Initialization in progress 1: Offline 2: Online	System 4 (FA Link)
Z263*	Cyclic Transmission Time		System 4 (FA Link) 1ms increments
Z267*	Local Station Status	0: Initialization in progress 1: Offline 2: Online	System 5 (FA Link)
Z268*	Cyclic Transmission Time		System 5 (FA Link) 1ms increments
Z272*	Local Station Status	0: Initialization in progress 1: Offline 2: Online	System 6 (FA Link)
Z273*	Cyclic Transmission Time		System 6 (FA Link) 1ms increments
Z277*	Local Station Status	0: Initialization in progress 1: Offline 2: Online	System 7 (FA Link)
Z278*	Cyclic Transmission Time		System 7 (FA Link) 1ms increments
Z282*	Local Station Status	0: Initialization in progress 1: Offline 2: Online	System 8 (FA Link)
Z283*	Cyclic Transmission Time		System 8 (FA Link) 1ms increments

*: Available with the F3SP22, F3SP25, F3SP28, F3SP35, F3SP38, F3SP53, F3SP58 and F3SP59 only.

TIP

Units that make up a system are known as stations.

Appendix 2.5 Sequence CPU Module Status Registers

CPU module status registers indicate the status of a CPU.

Table Appendix 2.5 Sequence CPU Module Status Registers

Item	CPU Module Status Registers		
No.	Name	Function	Description
Z105	Number of User Log Records		See Section 6.14, "User Log Management Functions" for information on user logs.
Z109*1	Sensor CB Execution Time	Time taken from starting of input refreshing for the sensor control block through program execution to completion of output refreshing. (Unit: 10 µs)	
Z111*1	Maximum Sensor CB Execution Time	The maximum time taken to execute the sensor control block. (Unit: 10 µs)	
Z121 to Z128*2	Model Information	CPU model name and revision number of firmware.	

*1: Only for F3SP22, F3SP28, F3SP38, F3SP53, F3SP58, and F3SP59

*2: F3SP22, F3SP28, F3SP38, F3SP53, F3SP58 and F3SP59

For example, module ”F3SP58-6S” with firmware Rev1,

Z121 ”F3”
Z122 “SP”
Z123 “58”
Z124 “6S”
Z125 “/R”
Z126 “01”
Z127 “/ “
Z128 “ “

F3SP05, F3SP08, F3SP21, F3SP25, F3SP35

For example, module ”F3SP21-0N” with firmware Rev 14

Z121 ”F3”
Z122 “SP”
Z123 “21”
Z124 “-0”
Z125 “*A”
Z126 “14”
Z127 “ “
Z128 “ “

Blank Page

Appendix 3. Forms for System Design

■ Program Coding Sheet

Sheet No.

System Name		Approved by	Checked by	Prepared by
Model	Drawing No.			

Instruction No.	Instruction	Operand					Remarks
0							
1							
2							
3							
4							
5							
6							
7							
8							
9							
0							
1							
2							
3							
4							
5							
6							
7							
8							
9							
0							
1							
2							
3							
4							
5							
6							
7							
8							
9							
0							

■ Relay Devices Assignment Table

Sheet No.

System Name		Approved by	Checked by	Prepared by
Model	Drawing No.			

Device No.	Signal Name	Description	Device No.	Signal Name	Description
1			3		
2			4		
3			5		
4			6		
5			7		
6			8		
7			9		
8			0		
9			1		
0			2		
1			3		
2			4		
3			5		
4			6		
5			7		
6			8		
7			9		
8			0		
9			1		
0			2		
1			3		
2			4		
3			5		
4			6		
5			7		
6			8		
7			9		
8			0		
9			1		
0			2		
1			3		
2			4		

■ Register Devices Assignment Table

Sheet No.

System Name		Approved by	Checked by	Prepared by
Model	Drawing No.			

Device No.	Signal Name	Description	Device No.	Signal Name	Description
1			1		
2			2		
3			3		
4			4		
5			5		
6			6		
7			7		
8			8		
9			9		
0			0		
1			1		
2			2		
3			3		
4			4		
5			5		
6			6		
7			7		
8			8		
9			9		
0			0		

■ Timer/Counter Setpoints Table

Sheet No.

System Name		Approved by	Checked by	Prepared by
Model	Drawing No.			

Device No	Setpoint	Signal Name	Description
1			
2			
3			
4			
5			
6			
7			
8			
9			
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
0			
1			
2			

FA-M3

Sequence CPU Instruction Manual - Functions

(for F3SP22-0S, F3SP28-3N/3S, F3SP38-6N/6S, F3SP53-4H/4S, F3SP58-6H/6S, F3SP59-7S)

IM 34M06P13-01E 4th Edition

Index

Blank Page

Revision Information

Document Name : Sequence CPU Instruction Manual - Functions
(for F3SP22, F3SP28-3N/3S, F3SP38-6N/6S, F3SP53-4H/4S, F3SP58-6H/6S, F3SP59-7S)
Document No. : IM 34M06P13-01E

Edition	Date	Revised Item
1st	Jul. 2000	New publication
2nd	Oct. 2002	Included F3SP59 for -□S Incorporated addendum, errata
3rd	June 2007	Included FL-net Incorporated addendum, errata
4th	Jan. 2012	Include F3SP22-0S Included WideField3 Incorporated errata

Written by PLC Group
International Sales Promotion Dept.
IA Systems Business Headquarters
Yokogawa Electric Corporation
Published by Yokogawa Electric Corporation
2-9-32 Nakacho, Musashino-shi, Tokyo, 180-8750, JAPAN
Printed by Kohoku Publishing & Printing Inc.

Blank Page