MITSUBISHI

MOTION CONTROLLER (SV13/22SFC)

Programming Manual

type A172SHCPUN, A173UHCPU(-S1), A273UHCPU-S3

MITSUBISHI ELECTRIC

INTORODUCTION

Thank you for choosing the motion controller.
Before using the equipment, please read this manual carefully to use it to its optimum.
Please forward this manual to the end user.

Safety Instructions

Do not attempt to install, operate, maintain or inspect this product until you have read through these safety instructions carefully and can use the equipment correctly. Do not use this product until you have a full knowledge of the equipment, safety information and instructions.
In this manual, the safety instruction levels are classified into "WARNING" and "CAUTION".

⚠ WARNING WARNING Denotes that incorrect handling may cause hazardous conditions, resulting in death or severe injury.

⚠ CAUTION CAUTION Denotes that incorrect handling may cause hazardous conditions, resulting in medium or slight injury, or may cause physical damage only.

CAUTION Denotes that incorrect handling may cause hazardous conditions, resulting in medium or slight injury, or may cause physical damage only.

SAFETY PRECAUTIONS

1. For Electric Shock Prevention

⚠ WARNING

- While power is on or the equipment is running, do not open the front casing and terminal cover. Doing so can cause an electric shock.
- Do not run the equipment with the front casing and terminal cover removed. The exposed high-voltage terminals and charging part can cause an electric shock.
- If power is off, do not remove the front casing and terminal cover except for wiring or periodic inspection. The controller and servo amplifier insides are charged and can cause an electric shock.
- Before starting wiring or inspection, switch power off, wait for more than 10 minutes, and check that there are no residual voltages with a tester or the like. Not doing so can cause an electric shock.
- Use the class 3 or higher grounding method to earth the controller, servo amplifiers and servo motors. In addition, do not share grounding with other equipment.
- Any person who is involved in the wiring or inspection of this equipment should be fully competent to do the work.
- Start wiring after installing the controller, servo amplifiers and servo motors. Not doing so can cause an electric shock or injury.
- Operate the switches with dry hands to prevent an electric shock.
- Do not subject the cables to scratches, excessive stress, heavy loads or pinching. Doing so can cause an electric shock.
- While power is on, do not touch the terminal blocks of the controller, servo amplifiers and servo motors. Doing so can cause an electric shock.
- Do not touch the internal power supplies, internal grounds and signal wires of the controller and servo amplifiers. Doing so can cause an electric shock.

2. For fire prevention

⚠ CAUTION

- Mount the controller, servo amplifiers, servo motors and regenerative brake resistors to incombustibles. Mounting them directly or near combustibles can cause a fire.
- If the controller or servo amplifier has failed, switch power off on the power supply side of the servo amplifier. A continuous flow of large current can cause a fire.
- When using a regenerative brake resistor, switch power off with an alarm signal. A regenerative brake transistor failure or the like can overheat the regenerative brake resistor abnormally, causing a fire.
- Provide anti-thermal measures such as flame-retarding treatment for the control box inside surfaces, where the servo amplifiers and regenerative brake resistors are installed, and the wires used.

3. For injury prevention

⚠ CAUTION
⚠ To each terminal, apply only the voltage specified in the A173UHCPU/A172SHCPUN/A171SHCPUN user's manual or the A273UHCPU user's manual and the instruction manuals of the products in use. Not doing so can cause burst, damage, etc.
⚠ Ensure that the cables are connected to the correct terminals. Wrong connection can cause burst, damage, etc.
⚠ Always make sure that polarity is correct. Wrong connection can cause burst, damage, etc.
⚠ While power is on or for some time after power-off, do not touch the servo amplifier radiating fins, regenerative brake resistors, servo motors, etc. as they are hot and you may get burnt.
⚠ Switch power off before touching the servo motor shaft and the machine coupled there. Not doing so can cause injury.
⚠ Stay away from the machine while it is being test-run or taught, for example. Not doing so can cause injury.

4. Additional Instructions

Also note the following points. Incorrect handling can cause a failure, injury, electric shock or the like.

(1) For system construction

⚠ CAUTION
⚠ Install earth leakage breakers for the power supplies of the controller and servo amplifiers.
⚠ When the instruction manuals of the servo amplifiers and like used specify that powering-off magnetic contactors must be installed for error occurrence, install magnetic contactors.
⚠ To ensure an immediate operation stop and power-off, install an external emergency stop circuit.
⚠ When using the controller, servo amplifiers, servo motors and regenerative brake resistors, combine them as specified in the A173UHCPU/A172SHCPUN/A171SHCPUN user's manual or the A273UHCPU user's manual and the instruction manuals of the products in use. Not doing so can cause a fire or failure.
⚠ When the system using the controller, servo amplifiers and servo motors has safety standards (e.g. safety rules for robots), the system must satisfy the safety standards.
⚠ If the abnormal operations of the controller and servo amplifiers differ from the safety-direction operation of the system, configure up remedial circuits outside the controller and servo amplifiers.
⚠ Use dynamic brakes with the servo motors if the coasting of the servo motor can cause a problem at an emergency stop, servo-off or power-off in the system.
⚠ Even if dynamic brakes are used, the coasting distance must be taken into consideration in the system.
⚠ If a vertical shaft drop can cause a problem at an emergency stop, servo-off or power-off in the system, use the dynamic brakes and electromagnetic brakes together.
⚠ Use dynamic brakes for only an error which will occur at an emergency stop or servo-off, and do not use them for normal braking.

⚠ CAUTION

- ⚠ The brakes (electromagnetic brakes) built in the servo motors are designed for holding. Do not use them for normal braking.
- ⚠ Configure up the system to ensure that it has such mechanical allowances that the axes can stop if they pass through stroke end limit switches at maximum speeds.
- ⚠ The wires and cables used should have the wire diameters, heat resistance and flex resistance conforming to the system.
- ⚠ The wires and cables used should have the lengths specified in the A173UHCPU/A172SHCPUN/A171SHCPUN user's manual or the A273UHCPU user's manual and the instruction manuals of the products in use.
- ⚠ The parts (other than the controller, servo amplifiers and servo motors) used with the system should be compatible in ratings and characteristics with the controller, servo amplifiers and servo motors.
- ⚠ To ensure that the rotary parts of the servo motors can never be touched during operation, provide the shafts with covers or the like.
- ⚠ Due to its life or mechanical structure (e.g. when a ballscrew and the servo motor are coupled via a timing belt), the electromagnetic brake may not provide sufficient holding force. Install a stopping device to ensure safety on the machine side.

(2) For parameter setting and programming

⚠ CAUTION

- ⚠ Set parameter values which meet the controller, servo amplifier, servo motor and regenerative brake resistor types and system applications. Wrong setting can disable the protective functions.
- ⚠ Set the regenerative brake resistor type and capacity parameter values which match the operation mode, servo amplifiers and servo power supply module. Wrong setting can disable the protective functions.
- ⚠ Set the mechanical brake output and dynamic brake output used/unused parameter values which meet the system applications. Wrong setting can disable the protective functions.
- ⚠ Set the stroke limit input used/unused parameter values which meet the system applications. Wrong setting can disable the protective functions.
- ⚠ Set the servo motor encoder type (incremental, absolute position type, etc.) parameter values which meet the system applications. Wrong setting can disable the protective functions.
- ⚠ Set the servo motor capacity and type (standard, low inertia, pancake, etc.) parameter values which meet the system applications. Wrong setting can disable the protective functions.
- ⚠ Set the servo amplifier capacity and type parameter values which meet the system applications. Wrong setting can disable the protective functions.
- ⚠ The program instructions used in programs should be used under the conditions specified in this manual.

⚠ CAUTION
⚠ Make the sequence function program capacity, device capacity, latch use range, I/O assignment and error detection-time continued operation enable/disable settings which meet the system applications. Wrong setting can disable the protective functions. ⚠ Some devices used in programs are fixed in applications. Use them under the conditions specified in this manual. ⚠ If communication stops due to a communication error or the like, the input devices and data registers assigned to a link hold the data right before a communication stop. Always use the error remedying interlock programs specified in the instruction manuals of the products in use. ⚠ For programs written for the special function modules, always use the interlock programs specified in the instruction manuals of the special function modules.

(3) For transportation and installation

⚠ CAUTION
⚠ Transport the products in the correct method which meets their weights. ⚠ Use the hanger of the servo motor to only transport the servo motor. Do not use it to transport the servo motor which is being mounted to a machine. ⚠ Do not stack the products over the limit. ⚠ When transporting the controller or servo amplifier, do not hold its wires and cables connected. ⚠ When transporting the servo motor, do not hold its cables, shaft and detector. ⚠ When transporting the controller or servo amplifier, do not hold its front casing. It may drop. ⚠ When transporting, installing or removing the controller or servo amplifier, do not hold its edges. ⚠ When installing the equipment, choose the place which will bear their weights and mount them in accordance with the A173UHCPU/A172SHCPUN/A171SHCPUN user's manual or the A273UHCPU user's manual and the instruction manuals of the products in use. ⚠ Do not stand or rest heavy objects on the product. ⚠ Check that the mounting orientation is correct. ⚠ Leave the specified clearances between the controller or servo amplifier and the control box inside surface, between the controller and the servo amplifier, and between the controller or servo amplifier and the other equipment. ⚠ Do not install or operate the controller, servo amplifiers and servo motors if they are damaged or have parts missing. ⚠ Do not block the suction and exhaust ports of the servo motor provided with a cooling fan. ⚠ Prevent screws, metal fragments or other conductive bodies or oil or other flammable substance from entering the controller, servo amplifiers and servo motors. ⚠ The controller, servo amplifiers and servo motors are precision machines. Do not drop them or give them hard impact.

⚠ CAUTION

⚠ Securely fix the controller and servo amplifiers to the machinery in accordance with the A173UHCPU/A172SHCPUN/A171SHCPUN user's manual or the A273UHCPU user's manual and the instruction manuals of the products in use. Insecure fixing may lead to removal during operation.

⚠ Always install the servo motor provided with reduction gear in the specified direction. Not doing so can cause oil leakage.

⚠ Store and use the equipment under the following environmental conditions.

Environment	Conditions	
	Control unit/servo amplifier	Servomotor
Ambient temperature	0°C to +55°C (Non-freezing)	0°C to +40°C (Non-freezing)
Ambient humidity	As in the instruction manual of the corresponding product	80%RH or less (Non-condensing)
Storage temperature	As in the instruction manual of the corresponding product	–20°C to +65°C
Atmosphere	Indoors (no direct sunlight) Free from corrosive gas, flammable gas, oil mist, dust and dirt	
Altitude	Max. 1000mm above sea level	
Vibration	As in the instruction manual of the corresponding product	

⚠ When coupling, do not give impact to the shaft end of the synchronous encoder or servo motor, e.g. do not hit it with a hammer. Doing so can cause a detector failure.

⚠ Do not give the servo motor shaft with loading of greater than the permissible. Such loading can cause the shaft to be broken.

⚠ When the equipment will not be used for an extended period of time, remove the power supply wires from the controller and servo amplifiers.

⚠ Store the controller and servo amplifiers in antistatic vinyl bags.

⚠ If they have been stored for an extended period of time, consult the service center or service station.

(4) For wiring

⚠ CAUTION

⚠ Wire the equipment correctly and securely. After wiring, recheck for wrong connections, insufficient terminal screw tightening, etc. Improper wiring can cause the servo motors to run away.

⚠ After wiring, reinstall the protective covers such as terminal covers.

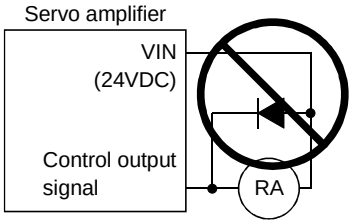
⚠ On the output side of the servo amplifier, do not fit a power capacitor, surge suppressor and radio noise filter (FR-BIF option).

⚠ Make correct connections on the output side (terminals U, V, W). Otherwise, the servo motors will run abnormally.

⚠ Do not connect a commercial power supply to the servo motors directly. Doing so can cause a failure.

⚠ CAUTION

⚠ Mount the surge suppressing diode to the DC relay designed for control output signal, such as a brake signal, in correct orientation. If it is mounted in incorrect orientation, the signal may not be output due to a failure, disabling the protective circuit.



⚠ While power is on, do not connect or disconnect the module-to-module connection cables, encoder cables and PLC extension cables.

⚠ Securely tighten the cable connector fixing screws and fixing mechanisms. Insecure fixing can cause removal during operation.

⚠ Do not bundle the power supply wires and cables.

(5) For test operation and adjustment

⚠ CAUTION

⚠ Before starting operation, confirm and adjust the programs and parameters. A failure to do so may cause some machines to make unexpected motions.

⚠ Never make extreme adjustment changes as they will make operations instable.

⚠ Always zero the axes when using the absolute position system function, after making a new startup, or after changing the controller, absolute value-compatible motor or the like.

(6) For usage

⚠ CAUTION

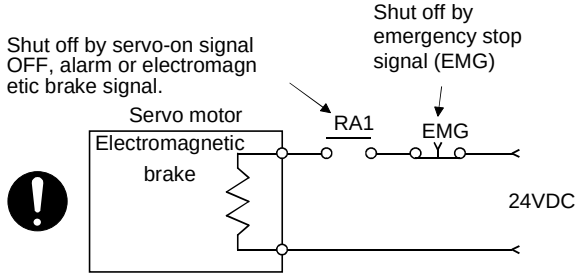
- ⚠ If any of the controller, servo amplifiers and servo motors has emitted smoke, unusual noise, unusual odor or the like, immediately switch power off.
- ⚠ After any program or parameter setting change or maintenance/inspection, always perform test operation before starting actual operation.
- ⚠ Any person who is involved in the disassembly or repair of this equipment should be fully competent to do the work.
- ⚠ Do not modify the equipment.
- ⚠ Install noise filters or shield the wiring, for example, to minimize the influence of electromagnetic interference. Electromagnetic interference may be given to the electronic equipment used near the controller and servo amplifiers.
- ⚠ As for use with CE mark-compatible installations, refer to the "EMC Installation Guidelines" (data number IB(NA)-67320) for motion controllers, and to the corresponding EMC guideline data for other equipment such as servo amplifiers and inverters.
- ⚠ Use the equipment under the following operating conditions.

Item	Conditions
Input power	As in the specifications of the A173UHCPU/A172SHCPUN/A273UHCPU (32-axes feature)
Input frequency	As in the specifications of the A173UHCPU/A172SHCPUN/A273UHCPU (32-axes feature)
Permissible instantaneous power failure time	As in the specifications of the A173UHCPU/A172SHCPUN/A273UHCPU (32-axes feature)

(7) For corrective actions for alarms

⚠ CAUTION

- ⚠ If a self-diagnostic error of the controller or servo amplifier has occurred, confirm the check items and recover in accordance with this manual and the instruction manuals of the products in use, and recover from the error.
- ⚠ If it is assumed that a power failure or product failure may result in a hazardous status, use a servo motor provided with electromagnetic brake or provide an external brake mechanism for holding purpose to prevent such hazard.
- ⚠ The electromagnetic brake operation circuit should have a double circuit structure so that the electromagnetic brake will also be operated by an external emergency stop signal.
- ⚠ Restart operation after removing the cause of alarm occurrence and ensuring safety.
- ⚠ When power is restored after an instantaneous power failure, stay away from the machine as it may restart suddenly. (Design the machine so that personal safety is secured if it restarts.)



(8) For maintenance, inspection and parts replacement

⚠ CAUTION

- ⚠ Perform daily inspection and periodic inspection in accordance with the instruction manuals.
- ⚠ Start maintenance/inspection after backing up the programs and parameters of the controller and servo amplifiers.
- ⚠ When opening or closing the doors and covers, do not put your hands and fingers into their gaps.
- ⚠ Change consumables such as batteries periodically in accordance with the A173UHCPU/A172SHCPUN/A171SHCPUN user's manual or the A273UHCPU user's manual and the instruction manuals of the products in use.
- ⚠ Do not touch the IC leads and contactor contacts.
- ⚠ Do not place the controller and servo amplifiers on metal which may leak electricity or on wood, plastic, vinyl or the like charged with static electricity.
- ⚠ Do not test the equipment with a megger (measure insulation resistance) during inspection.
- ⚠ After changing the controller or servo amplifier, make correct settings of the new unit.
- ⚠ After changing the controller or absolute position-compatible motor, zero the axes in either of the following methods. Not doing so will cause position shifts.
  (1) After writing the servo data to the PLC using the peripheral software, switch power off, then on again and perform zeroing operation.
  (2) Using the backup function of the peripheral software, load the before-replacement backup data.
- ⚠ At the end of maintenance/inspection, check whether the absolute position detecting function detects positions properly.
- ⚠ Do not short, recharge, overheat, burn or disassemble the batteries.
- ⚠ Since the electromagnetic capacitors emit gas if they fail, keep your face away from the controller and servo amplifiers.
- ⚠ The electromagnetic capacitors and fans will deteriorate. Change them periodically to prevent secondary damage. Consult the system service or service station for replacement.

(9) Disposal

⚠ CAUTION

- ⚠ Dispose of this product as general industrial waste.
- ⚠ Do not disassemble the controller, servo amplifier and servo motor parts.
- ⚠ Dispose of the batteries in the method prescribed in the corresponding municipality.

(10) General instruction

All illustrations given in this manual may have been drawn with covers or safety guards removed to provide in-depth description. Before starting operation of the product, always return the covers and guards into original positions as specified and operate the equipment in accordance with this manual.

Revisions

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

Print Date	Manual Number	Revision
Dec., 2000	IB(NA)-0300022-A	First edition

CONTENTS

1. OVERVIEW

This is a programming manual for the motion SFC-compatible CPU operating system software packages "SW3RN-SV13□", "SW3RN-SV22□" designed to run SFC programs on the motion CPU side.
Conventionally, a sequence of machine operations were controlled by the PLC CPU, and motion program start and stop control was exercised by the motion CPU under the start and stop commands of the PLC. Hence, a delay or variation of one PLC scan occurred at the worst between when a command condition enabled until a command was issued, limiting the applications where fast response and short tact time are pursued.
The motion SFC-compatible CPU operating system allows motion side programs to be written in SFC (Sequential Function Chart), which conforms to IEC1131-3, to control a sequence of machine operations. In addition, it also enables event control which runs a program at an interrupt input from an external sensor. Mainly performing the processings irrelevant to sequential control, the PLC controls ladder programs by constant scan execution.

1.1 Features

(1) Since the motion CPU judges whether a transition condition enabled or not to make a start, there are no response delays or variations affected by PLC scan time.

(2) The SFC step processing system (only active steps are executed) ensures rapid processing and fast response.

(3) The motion CPU can perform not only a motion program start but also numerical operations, device SET/RST, etc., making operations via the PLC unnecessary and improving tact time.

(4) The motion-specific transition condition description allows a command to be given to the servo amplifier immediately after a start condition enables.

(5) The motion-specific transition condition description allows a transition to the next step to be made after a start, without waiting for positioning completion.

(6) You can set programs (written in SFC) which run in fast response to external interrupt inputs (NMI).

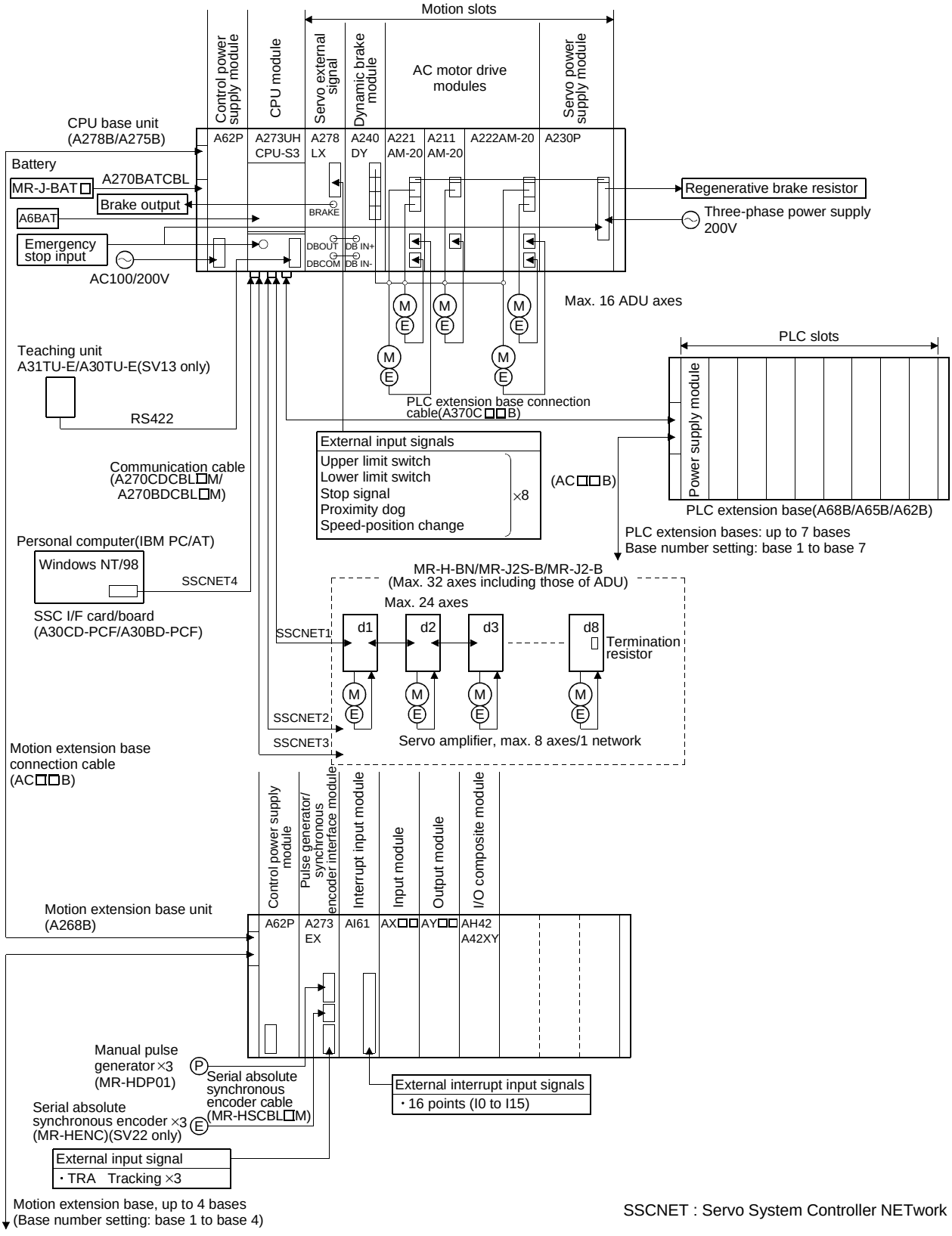
(7) You can set programs (written in SFC) which run in a short cycle (1.777ms, 3.555ms, 7.111ms, 14.222ms).

(8) As a sequence of machine operations can be written in correspondence with operation steps, the resultant program is easy for anyone to understand, improving maintainability.

2. SYSTEM CONFIGURATION

2.1 A273UHCPU-S3 System Overall Configuration

The following system configuration assumes use of the A273UHCPU-S3.



POINT

(1) I/O assignment
- When no I/O assignment is made, the I/O numbers of the PLC extension base 1 start from X/Y80.
- When you want to use the PLC extension base 1 at the I/O numbers of X/Y0 and later, make I/O assignment by setting slots 0 to 7 as "0 free points".

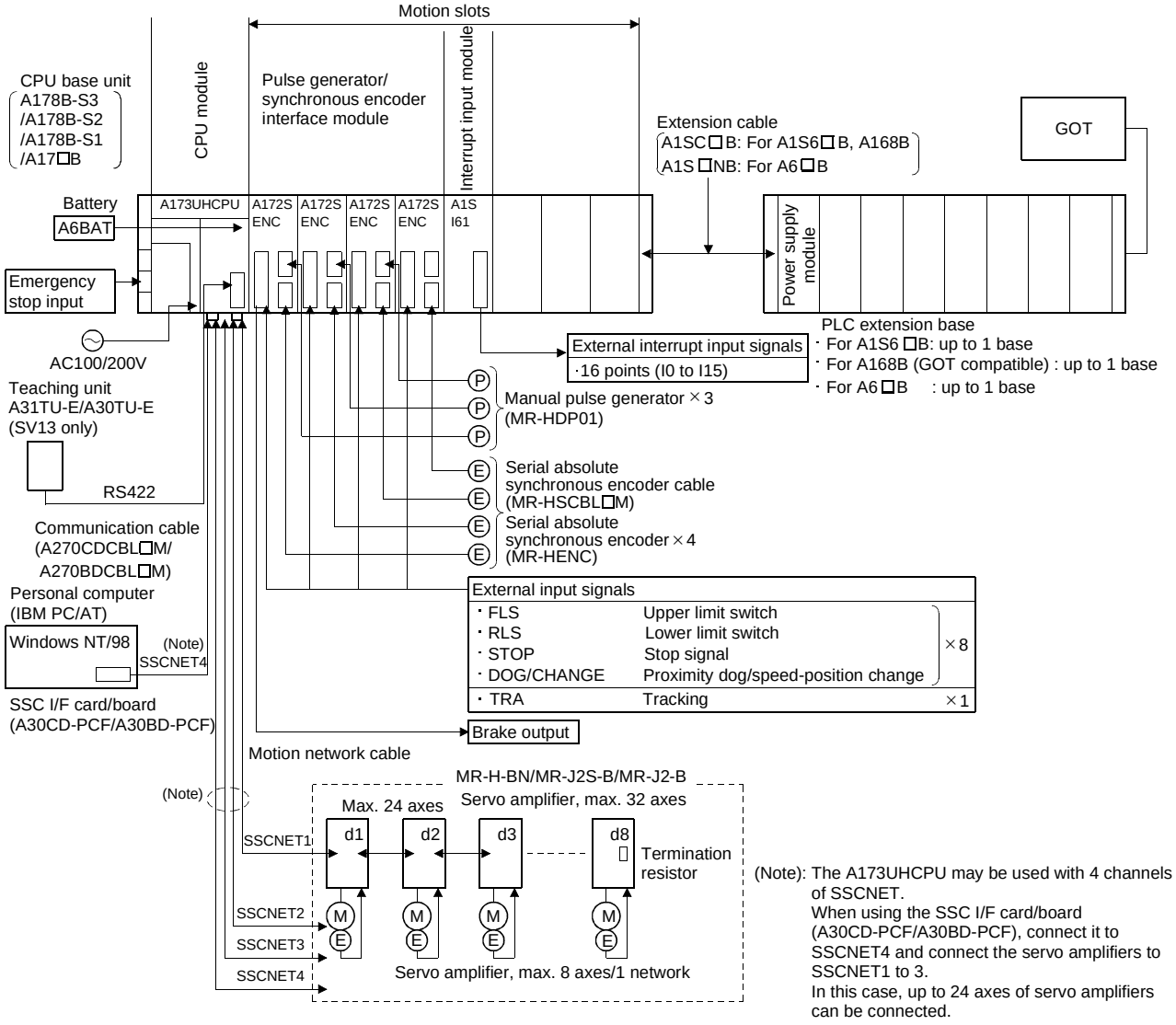
(2) The motion slots accept up to 256 I/O points.

(3) The I/O numbers of the I/O modules loaded in the motion slots should be later than the I/O numbers used with the PLC slots.

(4) The motion slots accept one AI61 interrupt input module.
This module is designed for only event/NMI input to the motion CPU and is irrelevant to PLC interrupt programs.

2.2 A173UHCPU(-S1) System Overall Configuration

The following system configuration assumes use of the A173UHCPU(-S1).



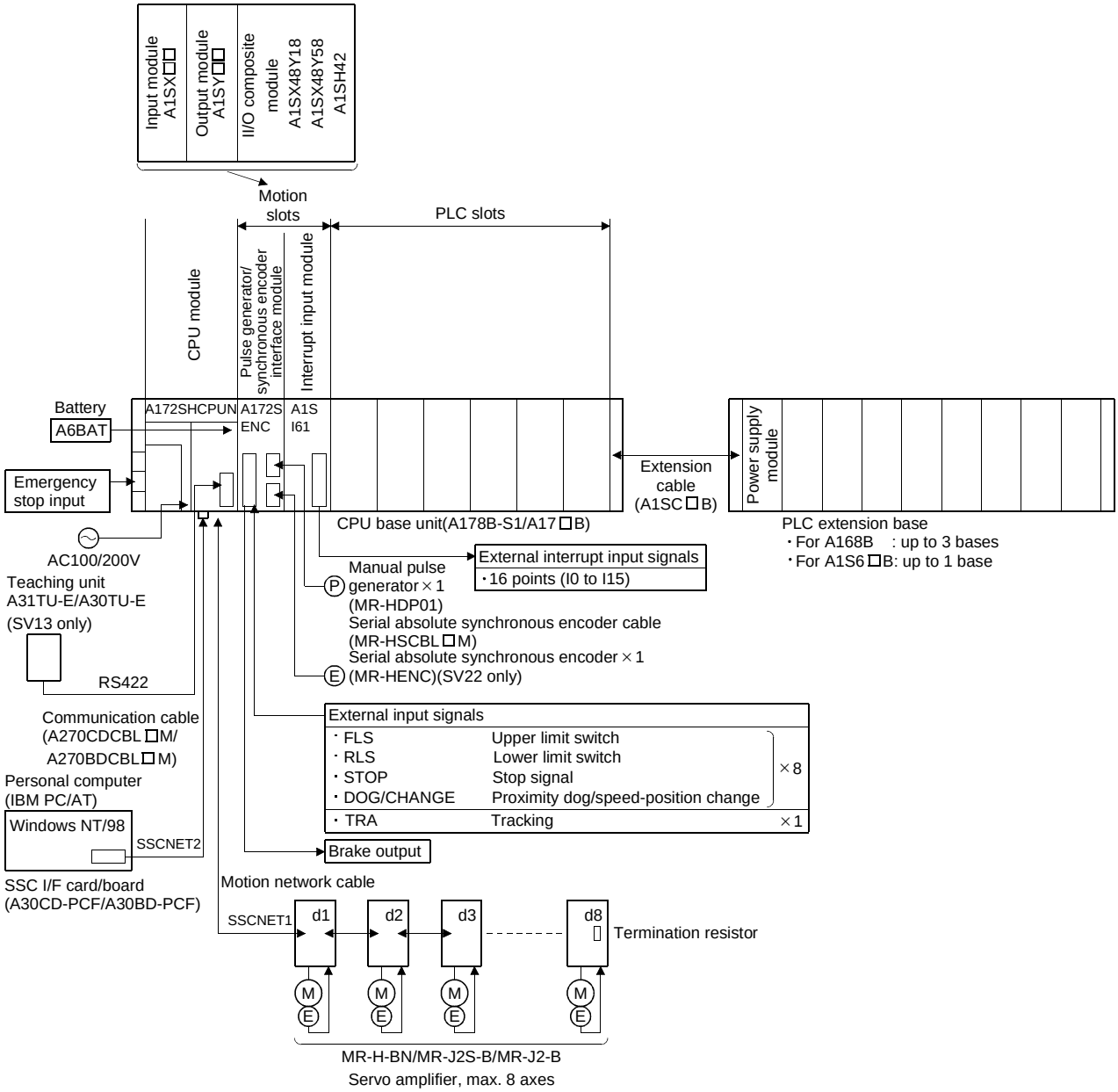
(Note): The A173UHCPU may be used with 4 channels of SSCNET.
When using the SSC I/F card/board (A30CD-PCF/A30BD-PCF), connect it to SSCNET4 and connect the servo amplifiers to SSCNET1 to 3.
In this case, up to 24 axes of servo amplifiers can be connected.

POINT

(1) Use the A168B when using the bus-connection type GOT.
(2) Using the A31TU-E teaching unit provided with deadman switch requires the exclusively used A31TUCBL03M connection cable between the CPU module and A31TU-E connector. The A31TU-E will not operate at all if it is connected directly with the RS422 connector of the CPU, without using the exclusively used cable.
Also, after disconnecting the A31TU-E, fit the A31SHORTCON short-circuit connector designed for A31TUCBL.
(3) The motion slots also accept PLC A1S I/O modules.
(4) The motion slots accept one A1SI61 interrupt input module.
This module is designed for only event/NMI input to the motion CPU and is irrelevant to PLC interrupt programs.
(5) The motion slots accept up to 256 I/O points.
(6) The I/O numbers of the I/O modules loaded in the motion slots should be later than the I/O numbers used with the PLC slots.

2.3 A172SHCPUN System Overall Configuration

The following system configuration assumes use of the A172SHCPUN.



POINT
(1) Use the A168B when setting one or more PLC extension bases. (2) The motion slots also accept PLC A1S I/O modules. The motion slots accept up to 256 I/O points. (Actually, the maximum points is 64-point modules * 2 = 128 points.) (3) The I/O numbers of the I/O modules loaded in the motion slots should be later than the I/O numbers used with the PLC slots. (4) The motion slots accept one A1SI61 interrupt input module. This module is designed for only event/NMI input to the motion CPU and is irrelevant to PLC interrupt programs.

2.4 Software Package List

Application	Peripheral Device	Programming Software Package Type	Operating System Software Package Type			Remarks
			For A172SHCPUN (8-axes feature)	For A173UHCPU(-S1) (32-axes feature)	For A273UHCPU-S3 (32-axes feature)	
For conveyor/ Assembly	IBM PC/AT	SW3RNC-GSVE	SW3RN-SV13D	SW3RN-SV13B	SW3RN-SV13X	With teaching function
For automatic machinery	IBM PC/AT		SW3RN-SV22C	SW3RN-SV22A	SW3RN-SV22W	Without teaching function

(1) Type definition

(Operating System) S W 3 R N - S V 1 3 X

→ Indicates motion SFC compatibility.

(Programming software) S W 3 R N _ - G S V 1 3 P

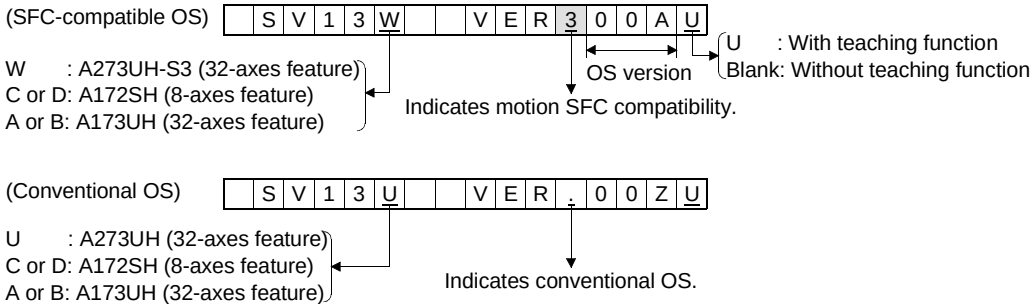
→ OS environment: Windows NT/98

→ Device: IBM PC/AT 100% compatible

→ Indicates conventional OS or motion SFC compatibility.

(2) OS type/version display

On the installation screen of the peripheral, the OS type/version of the connected CPU is displayed as shown below. When the A273UHCPU-S3 is used, this data is also indicated by the CPU front LEDs by performing an indicator reset.



3. PERFORMANCE SPECIFICATIONS

3.1 SFC Performance Specifications

Table 3.1 SFC Performance Specification List

Item			A172SHCPUN	A173UHCPU(-S1)/A273UHCPU-S3 32-axes feature
Program capacity	Code total (SFC chart + operation control + transition)		287k bytes	
	Text total (operation control + transition)		224k bytes	
	Motion control program (servo program)		52k bytes	Approx. 56k bytes
Program storage area	Code • motion control program		PCPU SRAM	
	Text		PCPU SRAM	
Motion SFC program	Number of motion SFC programs		256 (No.0 to 255)	
	Number of motion SFC steps/all programs (1 step+1 transition)		Max. approx. 7.5k steps (varies with the number of operation control program/transition program steps)	
	Motion SFC program name/ program		16 bytes (16 characters) (SFC program name is used as SFC file name)	
	Motion SFC chart size/ program		Max. 64k bytes (Motion SFC chart comments included)	
	Motion SFC steps/ program		Max. 4094 steps	
	Number of selective branches/ branch		255	
	Number of parallel branches/ branch		255	
	Parallel branch nesting		Up to 4 levels	
	Subroutine call nesting		No restrictions	
	Motion SFC chart comments		Max. 80 characters/ symbol	
Operation control program(F/FS) • Transition program (G)	Number of operation control programs	Once execution type	4096(F0 to F4095)	4096 with F and FS combined (F/FS0 to F/FS4095)
		Scan execution type	4096(FS0 to FS4095)	
	Number of transition programs		4096 (G0 to G4095)	
	Code size/ program		Max. approx. 64k bytes (32766 steps)	
	Text size/ program		Max. approx. 64k bytes	
	Number of blocks (lines)/ program		Max. 8192 blocks (in the case of 4 steps (minimum)/ block)	
	Number of characters/ block (line)		Max. 128 characters (comments included)	
	Number of operand/ block		Max. 64 (Operand : constants, word devices and bit devices)	
	( ) nesting/ block		Up to 32 levels	
	Indirect device designation nesting		Up to 2 levels	
	Separation of one block		CR + LF	
	Descriptive expression	Operation control program	Calculation expression / bit conditional expression	
		Transition program	Calculation expression/bit conditional expression/comparison conditional expression	
	Comment statement		Part after '//' is regarded as a comment.	
Motion control program	Number of servo programs		4096 (K0 to K4095)	
	Program steps/all programs		13312	14334
	Program steps/1 program		Max. 13312 steps/ program (for constant-speed control/speed change control)	
	Positioning points		Approx. 800 points/ axis	Approx. 400 points/ axis
Executed specifications	Number of multi executed programs		Max. 256 programs	
	Number of multi active steps		Max. 256 steps/all programs	
	Executed task	Normal task	Executed in motion main cycle	
		Event task	Fixed cycle (1.7ms, 3.5ms, 7.1ms, 14.2ms) 16 external interrupt points (inputs from AI61 interrupt input module installed in motion slot) Executed when 1 interrupt point is provided from PLC. (PLC dedicated instruction ITP is executed) Execution can be masked.	
		NMI task	16 external interrupt points (inputs from AI61 interrupt input module installed in motion slot) 16 points with event tasks and NMI tasks combined (use SFC parameters to set tasks)	

3.2 SCPU Performance Specifications

Table 3.2 SCPU Performance Specification List

Item			A172SHCPUN	A173UHCPU(-S1)		A273UHCPU-S3
Control method			Repeated operation using stored program			
I/O control method			Refresh mode / direct mode selectable	Refresh mode (direct mode can be used partially in accordance with the instruction)		
Programming language			Sequence control-dedicated language (relay symbol language, logic symbolic language, MELSAP II (SFC))			
Number of instructions	Sequence instructions		26	22		
	Basic instructions		131	249		
	Application instructions		102			
	Dedicated instructions		—	204		
	Motion-dedicated instructions		2			
Processing speed (PLC instruction)	Direct mode		0.25 to 1.9μs/step	—		
	Refresh mode		0.25μs/step	0.15μs/step		
Real I/O points			2048 (X/Y0 to X/Y7FF)	8192 (X/Y0 to X/Y1FFF)		
Number of actual I/O points			1024 (X/Y0 to X/Y3FF)	2048 (X/Y0 to X/Y7FF) (within the range of one extension base)		2048 (X/Y0 to X/Y7FF)
Watchdog timer (WDT)			10 to 2000ms	200ms		
Memory capacity (built-in RAM)			192k bytes	Standard	192k bytes (A3NMCA-24 equivalent)	—
				-S1	768k bytes (A3NMCA-96 equivalent)	
Program capacity	Main sequence		Max. 30k steps			
	Sub sequence		—	Max. 30k steps		
	Microcomputer program		Max. 58k steps	—		
Devices	Internal relay (M) points (Note-1)		1000 (M0 to M999)	Total 2048 (set in parameters)	7144 (M0 to M999) (M2048 to M8191)	Total 8192 (set in parameters)
	Latch relay (L) points		1048 (M1000 to M2047)		1048 (M1000 to M2047)	
	Step relay (S) points		0 points (defaults to none)		0 (defaults to none)	
	Link relay (B) points		1024 (B0 to B3FF)	8192 (B0 to B1FFF)		
	Annunciator (F) points		256 (F0 to F255)	2048 (F0 to B2047)		
	Timer (T) (set in parameters)	Number of points	256			2048
		Specifications: 100ms timer	T0 to T199 (setting time: 0.1 to 3276.7s)			
		Specifications: 10ms timer	T200 to T255 (setting time: 0.01 to 327.67s)			
		Specifications: 100ms retentive timer	Defaults to none (setting time: 0.1 to 3276.7s)			
		Specifications: Extended timer	—	T256 to T2047 (setting time: Depends on D, W, R)		
	Counter (C) (set in parameters)	Number of points	256			2048
		Specifications: Normal counter	C0 to C255 (setting range: 1 to 32767)			
		Specifications: Interrupt program counter	C244 to C255 (defaults to none) (setting range: 1 to 32767)			
		Specifications: Extended counter	—	C256 to C1023 (count value setting: Depends on D, W, R)		
	Data register (D) points (Note-1)		1024 (D0 to D1023)	8192 (D0 to D8191)		
	Link register (W) points		1024 (W0 to W3FF)	8192 (W0 to W1FFF)		
	File register (R) points		Max. 8192 (R0 to R8191) (set in parameter)			
	Accumulator (A) points		2 (A0, A1)			
	Index register (V/Z) points		2 (V, Z)	14 (V, V1 to V6, Z, Z1 to Z6)		
	Pointer (P) points		256 (P0 to P255)			
	Interrupt pointer (I) points		32 (I0 to I31)			
	Special relay (M) points		256 (M9000 to M9255)			
	Special register (D) points		256 (D9000 to D9255)			
Number of extended file register blocks			Max. 10 blocks (vary with memory capacity setting)	Standard	Max. 10 blocks(vary with memory capacity setting)	Max. 47 blocks (vary with memory cassette and memory capacity setting)
				-S1	Max. 47 blocks(vary with memory capacity setting)	
Number of comment points			Max. 4032 (64k bytes) 1 point = 16 bytes (set in increments of 64)			
Number of extended comment points (Note-2)			Max. 3968 (63k bytes) 1 point = 16 bytes (set in increments of 64)			
Self-diagnostic function			Detection of watchdog timer, memory, CPU, I/O, battery and other errors			
Error-time operation mode			Selection of stop or continue			
Output mode switching at STOP→RUN			Selection of before-STOP computation status re-output (default) or after-computation output			
Clock function			Year, month, day, hour, minute, second, day of week (automatic leap year judgment)			

(Note-1) : The positioning-dedicated device range varies with the OS.

(Note-2) : Extended comments are not stored into the internal memory of the CPU.

3.3 PCPU Performance Specifications

3.3.1 Motion control specifications

Table 3.3 PCPU Performance Specification List (Motion Control Specifications)

<table>
<tr><td colspan="2" rowspan="2">Item</td><td rowspan="2">A172SHCPUN</td><td rowspan="2">A173UHCPU(-S1)</td><td>A273UHCPU-S3</td></tr>
<tr><td>32-axis feature</td></tr>
<tr><td colspan="2">Number of control axes</td><td>8 axes (2 to 4 multi axes, 8 independent axes)</td><td colspan="2">32 axes (2 to 4 multi axes, 32 independent axes)</td></tr>
<tr><td colspan="2">Interpolation function</td><td colspan="3">Linear interpolation (max. 4 axes), circular interpolation (2 axes)</td></tr>
<tr><td colspan="2">Control method</td><td colspan="3">PTP (Point To Point), speed control, speed-position control, fixed-pitch feed, constant-speed control, position follow-up control, speed change control, high-speed oscillation control, current value change</td></tr>
<tr><td colspan="2">Control unit</td><td colspan="3">mm, inch, degree, PULSE</td></tr>
<tr><td colspan="2">Programming language</td><td colspan="3">Dedicated instructions (servo program)</td></tr>
<tr><td rowspan="3">Positioning</td><td>Method</td><td colspan="3">PTP : Absolute method / incremental method selection
Speed-position control / fixed-pitch feed : Incremental method
Constant-speed control / speed change control : Absolute method / incremental method may be mixed
Position follow-up control / current value change : Absolute method</td></tr>
<tr><td>Position command</td><td colspan="3">Can be selected per axis.
<table>
<tr><th>Control Unit</th><th>Command Unit</th><th>Address Setting Range</th><th>Travel Setting Range</th></tr>
<tr><td>mm</td><td>× 10^{-1} μm</td><td rowspan="2">-2147483648 to 2147483647</td><td rowspan="3">0 to ± 2147483647</td></tr>
<tr><td>inch</td><td>× 10^{-5} inch</td></tr>
<tr><td>degree</td><td>× 10^{-5} degree</td><td>0 to 35999999</td></tr>
<tr><td>PULSE</td><td>× 1 PULSE</td><td>-2147483648 to 2147483647</td><td></td></tr>
</table></td></tr>
<tr><td>Speed command (command unit)</td><td colspan="3">
<table>
<tr><th>Control Unit</th><th colspan="2">Speed Setting Range</th></tr>
<tr><td>mm</td><td>0.01 to 6000000.00</td><td>(mm/min)</td></tr>
<tr><td>inch</td><td>0.001 to 600000.000</td><td>(inch/min)</td></tr>
<tr><td>degree</td><td>0.001 to 2147483.647</td><td>(degree/min)</td></tr>
<tr><td>PULSE</td><td>1 to 10000000</td><td>(PLS/s)</td></tr>
</table></td></tr>
<tr><td rowspan="2">Acceleration/deceleration control</td><td>Automatic trapezoidal acceleration/deceleration</td><td colspan="3">
<table>
<tr><th>Automatic trapezoidal acceleration/deceleration</th><th>Time-Fixed Acceleration/Deceleration</th></tr>
<tr><td>Acceleration time: 1 to 65535ms</td><td rowspan="2">Acceleration/deceleration time : 1 to 5000ms(Enabled for constant-speed control only)</td></tr>
<tr><td>Deceleration time: 1 to 65535ms</td></tr>
</table></td></tr>
<tr><td>S-curve acceleration/deceleration</td><td colspan="3">S-curve ratio : 0 to 100%</td></tr>
<tr><td rowspan="2">Compensation</td><td>Backlash compensation</td><td colspan="3">(0 to 65535) × position command unit (0 to 65535 PULSE with unit converted into PULSE)</td></tr>
<tr><td>Electronic gear</td><td colspan="3">Function to compensate for actual travel error against command value</td></tr>
<tr><td colspan="2">Zeroing function</td><td colspan="3">Not in absolute position system : Proximity dog type or count type can be selected.
In absolute position system : Data setting type, proximity dog type or count type can be selected.</td></tr>
<tr><td colspan="2">JOG operation function</td><td colspan="3">Available</td></tr>
<tr><td colspan="2">Manual pulse generator operation function</td><td>One pulse generator can be connected.
Up to 3-axes can be operated simultaneously.
With smoothing scale factor setting
Input scale factor setting : 1 to 100</td><td>Three pulse generators can be connected.
One A172SENC is required per pulse generator.
Up to 3-axes can be operated simultaneously.
With smoothing scale factor setting
Input scale factor setting : 1 to 100</td><td>Three pulse generators can be connected.
Up to 3-axes can be operated simultaneously.
With smoothing scale factor
Input scale factor setting : 1 to 100</td></tr>
<tr><td colspan="2">M-function</td><td colspan="3">With M-code output function
With M-code completion waiting function</td></tr>
<tr><td colspan="2">Skip function</td><td colspan="3">Available</td></tr>
<tr><td rowspan="2">Limit switch output function</td><td>Number of output points</td><td colspan="3">32 points</td></tr>
<tr><td>Watch data</td><td colspan="3">Motion control data / word device (16-bit integer, 32-bit integer, 64-bit floating-point)</td></tr>
<tr><td rowspan="2">Specified data fast-read function</td><td>Number of read points</td><td colspan="2">Max. 9 points (A172SENC's TRA input (1 point) + one motion slot PLC input module (8 points))</td><td>Max. 11 points(A273EX's TRA input (3 points) + one motion slot PLC input module (8 points))</td></tr>
<tr><td>Data latch timing</td><td colspan="3">Leading edge of TRA input signal
Within 0.8ms from leading edge of input signal to PLC input module</td></tr>
<tr><td colspan="2">Absolute position system</td><td colspan="3">Made compatible by fitting battery to servo amplifier.
(Absolute or incremental system can be specified per axis.)</td></tr>
</table>

3.3.2 Operation control/transition control specifications

Table 3.4 PCPU Performance Specification List (Operation Control/Transition Control Specifications)

<table>
<tr><th>Item</th><th colspan="3">Specifications</th><th>Remarks</th></tr>
<tr><td rowspan="3">Expression</td><td colspan="2">Calculation expression</td><td>Returns a numeric result.
Expressions for calculating indirectly specified data using constants and word devices.</td><td>D100+1, SIN(D100F), etc.</td></tr>
<tr><td rowspan="2">Condi-tional expres-sion</td><td>Bit conditional expression</td><td>Returns a true or false result.
Expression for judging ON or OFF of bit device.</td><td>M0, !M0, M1*!M0,
(M1+M2)*(!M3+M4), etc.</td></tr>
<tr><td>Comparison conditional expression</td><td>Expressions for comparing indirectly specified data and calculation expressions using constants and word devices.</td><td>D100==100,
D10<D102+D10, etc.</td></tr>
<tr><td>Specifiable PLC devices (bit devices)</td><td colspan="3">
<table>
<tr><th colspan="2" rowspan="2">Device</th><th rowspan="2">Symbol</th><th colspan="2">Accessibility</th><th colspan="3">Usable Tasks</th><th rowspan="2">Descripti on Example</th></tr>
<tr><th>Read</th><th>Write</th><th>Normal</th><th>Event</th><th>NMI</th></tr>
<tr><td rowspan="2">Input</td><td>SBUS</td><td>X</td><td>○</td><td>○</td><td rowspan="14">○</td><td rowspan="14">○</td><td rowspan="14">○</td><td>X100</td></tr>
<tr><td>PBUS</td><td>PX</td><td>○</td><td>×</td><td>PX180</td></tr>
<tr><td rowspan="2">Output</td><td>SBUS</td><td>Y</td><td>○</td><td>○</td><td>Y000</td></tr>
<tr><td>PBUS</td><td>PY</td><td>○</td><td>○</td><td>PY1E0</td></tr>
<tr><td colspan="2">Internal relay</td><td>M</td><td>○</td><td>○ (Note-1)</td><td>M20</td></tr>
<tr><td colspan="2">Latch relay</td><td>L</td><td>○</td><td>○</td><td>L1000</td></tr>
<tr><td colspan="2">Link relay</td><td>B</td><td>○</td><td>○</td><td>B3FF</td></tr>
<tr><td colspan="2">Annunciator</td><td>F</td><td>○</td><td>○</td><td>F0</td></tr>
<tr><td colspan="2">Timer contact</td><td>TT</td><td>○</td><td>×</td><td>TT10</td></tr>
<tr><td colspan="2">Timer coil</td><td>TC</td><td>○</td><td>○</td><td>TC10</td></tr>
<tr><td colspan="2">Counter contact</td><td>CT</td><td>○</td><td>×</td><td>CT0</td></tr>
<tr><td colspan="2">Counter coil</td><td>CC</td><td>○</td><td>○</td><td>CC0</td></tr>
<tr><td colspan="2">Special M</td><td>M</td><td>○</td><td>○ (Note-1)</td><td>M9000</td></tr>
</table>
CAUTION
<Restrictions on write-enabled bit devices>
1) Write (e.g. SET, RST, OUT) to the same bit device (16 point increments) cannot be performed from both the sequence ladder and motion SFC programs. (Write operation cannot be guaranteed.) Hence, the side on which write is performed should be managed by the user. Minimum increments are 16 points.
2) Write to device X is allowed only within the input card non-loaded range and to the free numbers outside the link range. When the A172SHCPUN is used, note that the following read response delay will occur from when the SET/RST/DOUT instruction is executed by the motion CPU.
<table>
<tr><th>Read CPU</th><th>Response Delay</th><th>Remarks</th></tr>
<tr><td>Motion CPU</td><td>1 PLC scan</td><td>Do not use for such applications where ON/OFF judgment will be made by the motion CPU right after the SET/RST/ DOUT instruction is executed.</td></tr>
<tr><td>PLC CPU</td><td>None</td><td></td></tr>
</table>
3) When the I/O control method is the "direct system" (A172SHCPUN only), performing write to device Y will not provide output to the output card in the PLC slot. Use the "refresh system" to provide output to the PLC slot.
4) Special M has predetermined applications in the system. Do not perform write to other than the user-set device.
(Note-1) : SET/RST is disabled in the following device ranges.
<table>
<tr><th>CPU</th><th>SET/RST Disable Range</th><th>Remarks</th></tr>
<tr><td>A172SHCPUN</td><td>M2001 to M2008</td><td rowspan="2">Start accepting device</td></tr>
<tr><td>A173UHCPU(-S1) /A273UHCPU-S3</td><td>M2001 to M2032</td></tr>
</table>
• DOUT output is disabled in the following device ranges.
<table>
<tr><th>CPU</th><th>DOUT Output Disable Range</th><th>Remarks</th></tr>
<tr><td rowspan="2">A172SHCPUN</td><td>Designation including M2000 to M2047</td><td>Dedicated device</td></tr>
<tr><td>M9000 to M9255</td><td>Special M</td></tr>
<tr><td rowspan="2">A173UHCPU(-S1) /A273UHCPU-S3</td><td>Designation including M2000 to M2127</td><td>Dedicated device</td></tr>
<tr><td>M9000 to M9255</td><td>Special M</td></tr>
</table>
</td><td>• Each device range (points) varies with the CPU. Refer to "3.2 SCPU Performance Specifications".

• Inputs X/outputs Y on the PLC slot side are represented as SBUS and those on the motion slot side as PBUS. (In the operation control program/transition program, they are automatically represented as PX/PY according to the system setting information.)

*The input response of device X and the output response of Y differ between the SBUS (PLC slot) and PBUS (motion slot). For details, refer to the next page.</td></tr>
</table>

Table 3.4 PCPU Performance Specification List (Operation Control/Transition Control Specifications) (Continued)

<table>
<tr><th>Item</th><th>Specifications</th><th>Remarks</th></tr>
<tr><td>Specifiable PLC device (word device)</td><td>
<table>
<tr><th rowspan="2">Device</th><th rowspan="2">Symbol</th><th colspan="2">Accessibility</th><th colspan="3">Usable Tasks</th><th rowspan="2">Descrip-tion Example</th></tr>
<tr><th>Read</th><th>Write</th><th>Normal</th><th>Event</th><th>NMI</th></tr>
<tr><td>Data register</td><td>D</td><td>○</td><td>○</td><td rowspan="5">○</td><td rowspan="5">○</td><td rowspan="5">○</td><td>D0L</td></tr>
<tr><td>Link register</td><td>W</td><td>○</td><td>○</td><td>W1F:F</td></tr>
<tr><td>Timer current value</td><td>T</td><td>○</td><td>×</td><td>T0</td></tr>
<tr><td>Counter current value</td><td>C</td><td>○</td><td>×</td><td>C10</td></tr>
<tr><td>Special D</td><td>D</td><td>○</td><td>○</td><td>D9000</td></tr>
</table>
CAUTION
<Restrictions on write-enabled word devices>
1) Special D has predetermined applications in the system.
Do not perform write to other than the user-set device.
</td><td></td></tr>
<tr><td>Motion device (word device)</td><td>
<table>
<tr><td rowspan="2">Motion register (#)</td><td>Number of points</td><td>8192 points (#0 to #8191)
(#8000 to #8191 are SFC-dedicated devices)</td></tr>
<tr><td>Data</td><td>16 bits/point</td></tr>
<tr><td rowspan="2">Coasting timer (FT)</td><td>Number of points</td><td>1 point (FT)</td></tr>
<tr><td>Data</td><td>32 bits/point (-2147483648 to 2147483647)</td></tr>
</table>
<table>
<tr><th rowspan="2">Device</th><th rowspan="2">Symbol</th><th colspan="2">Accessibility</th><th colspan="3">Usable Tasks</th><th rowspan="2">Description Example</th></tr>
<tr><th>Read</th><th>Write</th><th>Normal</th><th>Event</th><th>NMI</th></tr>
<tr><td>Motion register</td><td>#</td><td>○</td><td>○</td><td rowspan="2">○</td><td rowspan="2">○</td><td rowspan="2">○</td><td>#OF</td></tr>
<tr><td>Coasting timer</td><td>FT</td><td>○</td><td>×</td><td>FT</td></tr>
</table>
CAUTION
1) The motion device cannot be accessed directly from the sequence ladder.
When you want to read, perform read via the PLC device (after assignment).
</td><td>A273UH-S3, A173UH(-S1) and A172SH have the same specifications.</td></tr>
<tr><td rowspan="5">Data type</td><td rowspan="2">(None)</td><td>16-bit integer type (signed)</td><td>-32768 to 32767</td><td rowspan="2">K10, D100, etc.</td></tr>
<tr><td>16-bit integer type (unsigned)</td><td>0 to 65535</td></tr>
<tr><td rowspan="2">L</td><td>32-bit integer type (signed)</td><td>-2147483648 to 2147483647</td><td rowspan="2">2147283647, W100L, etc.</td></tr>
<tr><td>32-bit integer type (unsigned)</td><td>0 to 4294967295</td></tr>
<tr><td>F</td><td>64-bit floating-point type
(double precision real number type)</td><td>IEEE format</td><td>1.23, #10F, etc.</td></tr>
<tr><td rowspan="3">Constant</td><td>K</td><td>Decimal constant</td><td rowspan="3">The above data type symbol 'L' or '. (decimal point)' provided at the end indicates the data type. The constant without the data type is regarded as the applicable minimum type.</td><td rowspan="3">K-100, H0FFL, etc.
'K' may be omitted.</td></tr>
<tr><td>H</td><td>Hexadecimal constant</td></tr>
<tr><td></td><td></td></tr>
<tr><td>Number of instructions</td><td>
<table>
<tr><td>Binary operation</td><td>6</td></tr>
<tr><td>Bit operation</td><td>6</td></tr>
<tr><td>Sign</td><td>1</td></tr>
<tr><td>Standard function</td><td>15</td></tr>
<tr><td>Type conversion</td><td>6</td></tr>
<tr><td>Bit device status</td><td>2</td></tr>
<tr><td>Bit device control</td><td>4</td></tr>
<tr><td>Logical operation</td><td>4</td></tr>
<tr><td>Comparison operation</td><td>6</td></tr>
<tr><td>Motion dedicated function</td><td>2</td></tr>
<tr><td>Others</td><td>5</td></tr>
</table>
</td><td>57 in total</td><td></td></tr>
<tr><td rowspan="5">Read response of input X on motion CPU</td><td rowspan="2">PBUS
(Input module in motion slot)</td><td colspan="2">Input response</td><td>Direct read control at instruction execution</td><td rowspan="5">• Input X and output Y of the PBUS are always under direct control, independently of the PLC's I/O control method.

CAUTION
• Output Y of the PBUS is write-enabled from only the motion CPU.
Do not perform write from the PLC CPU (output is not provided to real output).</td></tr>
<tr><td colspan="2">Response to PLC device</td><td>Refresh is made at instruction execution and in normal task cycle.</td></tr>
<tr><td rowspan="2">SBUS
(Input module in PLC slot)</td><td>Refresh mode</td><td>Input response</td><td>1 sequence scan</td></tr>
<tr><td>Direct mode
(A172SH only)</td><td>Input response</td><td>1 sequence scan</td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td rowspan="5">Write response of output Y on motion CPU</td><td rowspan="2">PBUS</td><td colspan="2">Response to PLC device</td><td>At instruction execution</td></tr>
<tr><td colspan="2">Actual output response</td><td>Direct write control at instruction execution</td></tr>
<tr><td rowspan="3">SBUS</td><td rowspan="2">Refresh mode</td><td>Response to PLC device</td><td>At instruction execution</td></tr>
<tr><td>Actual output response</td><td>Refreshed at PLC's END</td></tr>
<tr><td>Direct mode
(A172SH only)</td><td>Response to PLC device</td><td>At instruction execution</td></tr>
<tr><td></td><td></td><td>Actual output response</td><td>Not output</td></tr>
</table>

(1) Operation control/transition instruction list

Classification	Symbol	Function	Format	Number of Basic Steps	Usable Programs		Usable Expressions			
					F/FS	G	Calculation expression	Bit conditional expression	Comparison conditional expression	Y/N transition's conditional expression
Binary operation	=	Substitution	(D)=(S)	4	○	○	○	—	—	—
	+	Addition	(S1)+(S2)	4	○	○	○	—	—	—
	-	Subtraction	(S1)-(S2)	4	○	○	○	—	—	—
	*	Multiplication	(S1)*(S2)	4	○	○	○	—	—	—
	/	Division	(S1)/(S2)	4	○	○	○	—	—	—
	%	Remainder	(S1)%(S2)	4	○	○	○	—	—	—
Bit operation	~	Bit inversion (complement)	~(S)	2	○	○	○	—	—	—
	&	Bit logical AND	(S1)&(S2)	4	○	○	○	—	—	—
	|	Bit logical OR	(S1)|(S2)	4	○	○	○	—	—	—
	^	Bit exclusive OR	(S1)^(S2)	4	○	○	○	—	—	—
	>>	Bit right shift	(S1)>>(S2)	4	○	○	○	—	—	—
	<<	Bit left shift	(S1)<<(S2)	4	○	○	○	—	—	—
Sign	-	Sign inversion (complement of 2)	-(S)	4	○	○	○	—	—	—
Standard function	SIN	Sine	SIN(S)	2	○	○	○	—	—	—
	COS	Cosine	COS(S)	2	○	○	○	—	—	—
	TAN	Tangent	TAN(S)	2	○	○	○	—	—	—
	ASIN	Arcsine	ASIN(S)	2	○	○	○	—	—	—
	ACOS	Arccosine	ACOS(S)	2	○	○	○	—	—	—
	ATAN	Arctangent	ATAN(S)	2	○	○	○	—	—	—
	SQRT	Square root	SQRT(S)	2	○	○	○	—	—	—
	LN	Natural logarithm	LN(S)	2	○	○	○	—	—	—
	EXP	Exponential operation	EXP(S)	2	○	○	○	—	—	—
	ABS	Absolute value	ABS(S)	2	○	○	○	—	—	—
	RND	Round-off	RND(S)	2	○	○	○	—	—	—
	FIX	Round-down	FIX(S)	2	○	○	○	—	—	—
	FUP	Round-up	FUP(S)	2	○	○	○	—	—	—
	BIN	BCD→BIN conversion	BIN(S)	2	○	○	○	—	—	—
	BCD	BIN→BCD conversion	BCD(S)	2	○	○	○	—	—	—
Type conversion	SHORT	Converted into 16-bit integer type (signed)	SHORT(S)	2	○	○	○	—	—	—
	USHORT	Converted into 16-bit integer type (unsigned)	USHORT(S)	2	○	○	○	—	—	—
	LONG	Converted into 32-bit integer type (signed)	LONG(S)	2	○	○	○	—	—	—
	ULONG	Converted into 32-bit integer type (unsigned)	ULONG(S)	2	○	○	○	—	—	—
	FLOAT	Regarded as signed data and converted into 64-bit floating-point type	FLOAT(S)	2	○	○	○	—	—	—
	UFLOAT	Regarded as unsigned data and converted into 64-bit floating-point type	UFLOAT(S)	2	○	○	○	—	—	—
Bit device status	(None)	ON (normally open contact)	(Bit conditional expression)	2	○	○	—	○	—	○
	!	OFF (normally closed contact)	!(Bit conditional expression)	2	○	○	—	○	—	○

Classification	Symbol	Function	Format	Number of Basic Steps	Usable Programs		Usable Expressions			
					F/FS	G	Calculation expression	Bit conditional expression	Comparison conditional expression	Y/N transition's conditional expression
Bit device control	SET	Device set	SET(D)	3	○	○	—	○	—	—
			SET(D)= (conditional expression)	4	○	○	—	○	○	—
	RST	Device reset	RST(D)	3	○	○	—	○	—	—
			RST(D)= (conditional expression)	4	○	○	—	○	○	—
	DOUT	Device output	DOUT(D),(S)	4	○	○	—	○	—	—
	DIN	Device input	DIN(D),(S)	4	○	○	—	○	—	—
Logical operation	(None)	Logical acknowledgement	(Conditional expression)	0	○	○	—	○	○	○
	!	Logical negation	!(conditional expression)	2	○	○	—	○	○	○
	*	Logical AND	(Conditional expression) * (conditional expression)	4	○	○	—	○	○	○
	+	Logical OR	(Conditional expression) + (conditional expression)	4	○	○	—	○	○	○
Comparison operation	==	Equal to	(Calculation expression) == (calculation expression)	4	○	○	—	—	○	○
	!=	Not equal to	(Calculation expression) != (calculation expression)	4	○	○	—	—	○	○
	<	Less than	(Calculation expression) < (calculation expression)	4	○	○	—	—	○	○
	<=	Less than or equal to	(Calculation expression) <= (calculation expression)	4	○	○	—	—	○	○
	>	More than	(Calculation expression) > (calculation expression)	4	○	○	—	—	○	○
	>=	More than or equal to	(Calculation expression) >= (calculation expression)	4	○	○	—	—	○	○
Motion dedicated function	CHGV	Speed change request	CHGV((S1),(S2))	4	○	○	—	—	—	—
	CHGT	Torque limit value change request	CHGT((S1),(S2))	4	○	○	—	—	—	—
Others	EI	Event task enable	EI	1	○	○	—	—	—	—
	DI	Event task disable	DI	1	○	○	—	—	—	—
	NOP	No operation	NOP	1	○	○	—	—	—	—
	BMOV	Block move	BMOV(D),(S),(n)	7	○	○	—	—	—	—
	TIME	Time to wait	TIME(S)	7	—	○	—	—	—	—

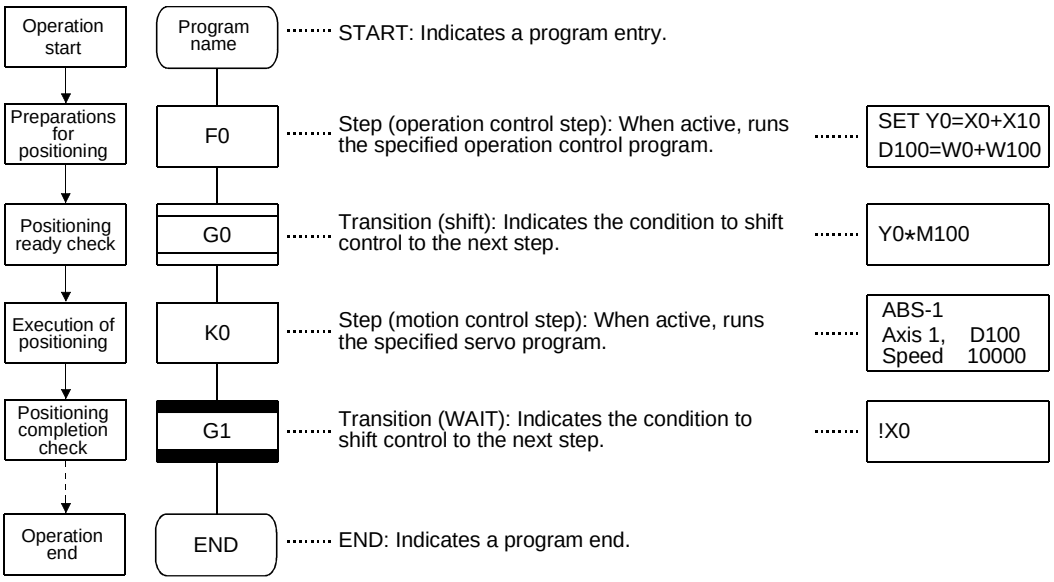
(2) Rough calculation expression for operation control/transition program's single-program code size

2 + (1 + total number of basic steps in 1 block
+ number of 32-bit constants/1 block × 1
+ number of 64-bit constants/1 block × 3) × number of blocks (steps)
(1 step = 2 bytes)

4. SFC PROGRAMS

4.1 SFC Program Structure

As shown below, an SFC program consists of START, steps, transitions, END and others.



When started, the above SFC program performs the following operations.

(1) The step (F0) is activated and the operation specified at the step (F0) is performed (preparations for positioning). A step in such an active state is called an active step.

(2) Whether the condition specified at the transition (G0) has enabled or not (whether the positioning program can be started or not) is checked. When the condition enables, the active step (F0) is deactivated and the next step (K0) is activated (servo program K0 is started).

(3) At the transition (G1), whether the step (K0) has completed its operation (servo program K0 has completed positioning) is checked. When the operation is completed (condition enables), control transits to the next step.

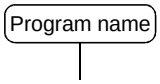
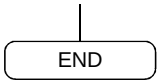
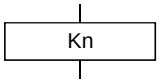
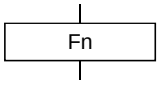
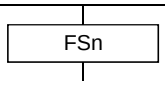
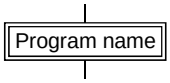
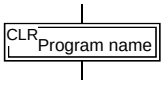
(4) With the transition of an active step as described in above (1) to (3), control is exercised and ends at END.

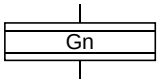
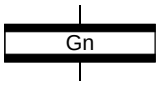
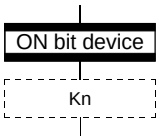
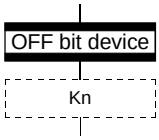
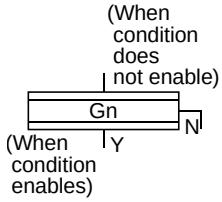
Refer to Chapter 9 Task Operations for details of the run timing of the SFC program such as above.

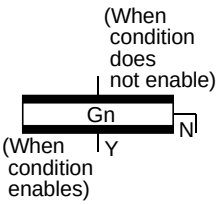
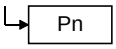
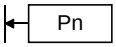
POINT
The number of steps which can be active steps simultaneously is up to 256, with those of all SFC programs combined. Excess of 256 will result in an SFC program error 16120.

4.2 SFC Chart Symbol List

Parts acting as SFC program components are shown below. In an SFC program, these parts are connected by directed lines to represent an operation sequence and transition control.

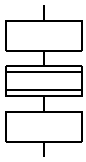
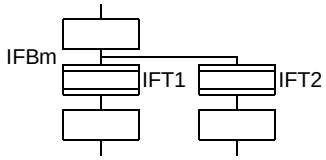
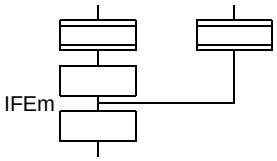
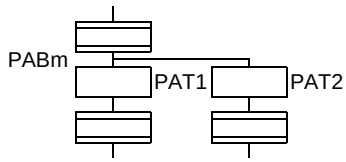
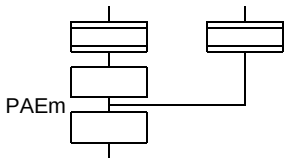
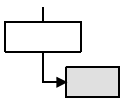
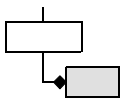
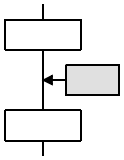
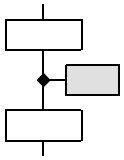
Classification	Name	Symbol (Code size (byte))	List Representation	Function
Program start/end	START	 (0)	Program name	• Indicates a program entry with a program name. • Specify this program name for a subroutine call. • Only one program name may be used with one program.
	END	 (8)	END	• Indicates a program end (exit). • When a subroutine called is made, execution returns to the call source program. • Multiple or no symbols may be set within one program.
Step	Motion control step	 (8)	CALL Kn	• Starts a servo program Kn (K0 to K4095).
	Once execution operation control step	 (8)	CALL Fn	• Runs an operation control program Fn (F0 to F4095) once.
	Scan execution type operation control step	 (8)	CALL FSn	• Repeats an operation control program FSn (FS0 to FS4095) until the next transition condition enables.
	Subroutine call/start step	 (8)	GSUB program name	• GSUB followed by WAIT performs a "subroutine call" and shifts control to the specified program. When END is executed, control returns to the call source program. • GSUB followed by other than WAIT performs a "subroutine start", starts the specified program, and shifts execution to the next (lower part). The start source and destination programs are run at the same time, and when END is executed, the call destination program ends.
	Clear step	 (8)	CLR program name	• Stops and ends the specified program being run. After an end, restarting the program starts it from the initial (start step). • When the specified program is being "subroutine called", the subroutine program being run is also stopped. • After the specified program has been "subroutine started", the subroutine program being run is not stopped. • When clear is performed on the "subroutine called" subroutine, the specified subroutine being run is stopped, and execution returns to the call source program and shifts to the next.

Classification	Name	Symbol (Code size (byte))	List Representation	Function
Transition	Shift (Pre-read transition)	 (8)	SFT Gn	• When this transition is preceded by a motion control step, execution does not wait for completion of the motion operation, and shifts to the next step when the transition condition Gn (G0 to G4095) enables. • When this transition is preceded by an operation control step, execution shifts to the next step when the transition condition enables after operation has been performed. • When this transition is preceded by a subroutine call/start step, execution does not wait for completion of the subroutine operation, and shifts to the next step when the transition condition enables.
	WAIT	 (8)	WAIT Gn	• When this transition is preceded by a motion control step, execution waits for completion of the motion operation and shifts to the next step when the transition condition Gn (G0 to G4095) enables. • When this transition is preceded by an operation control step, execution shifts to the next step when the transition condition enables after operation has been performed (same operation is performed as in Shift). • When this transition is preceded by a subroutine call/start step, execution waits for completion of the subroutine operation and shifts to the next step when the transition condition enables.
	WAITON	 (14)	WAITON bit device	• Prepares for starting the next motion control step, and when the specified bit device turns ON, issues a command immediately. • Always pair this transition with a motion control step one-for-one.
	WAITOFF	 (14)	WAITOFF bit device	• Prepares for starting the next motion control step, and when the specified bit device turns OFF, issues a command immediately. • Always pair this transition with a motion control step one-for-one.
	Shift Y/N		IFBm IFT1 SFT Gn : JMP IFEm IFT2 SFT Gn+? : JMP IFEm IFEm	• When this transition is preceded by a motion control step, execution does not wait for completion of the motion operation, and shifts to the lower step when the transition condition Gn (G0 to G4095) enables, or shifts to the right-connected step when the condition does not enable. • When this transition is preceded by an operation control step, execution shifts to the low step when the transition condition enables after operation has been performed, or shifts to the right-connected step when the condition does not enable. • When this transition is preceded by a subroutine call/start step, execution does not wait for completion of the subroutine operation, and shifts to the lower step when the transition condition enables, or shifts to the right-connected step when the condition does not enable.

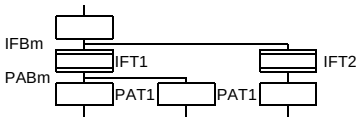
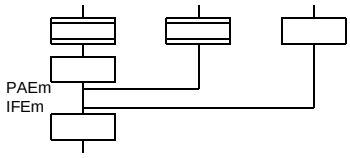
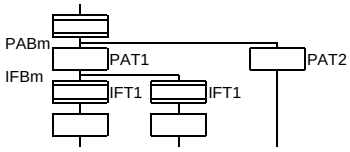
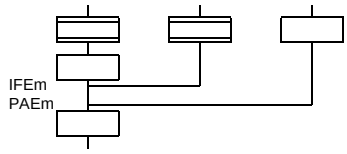
Classification	Name	Symbol (Code size (byte))	List Representation	Function
Transition	WAIT Y/N		IFBm IFT1 WAIT Gn : JMP IFEm IFT2 SFT Gn+? : JMP IFEm IFEm	• When this transition is preceded by a motion control step, execution waits for completion of the motion operation, and shifts to the lower step when the transition condition Gn (G0 to G4095) enables, or shifts to the right-connected step when the condition does not enable. • When this transition is preceded by an operation control step, execution shifts to the low step when the transition condition enables after operation has been performed, or shifts to the right-connected step when the condition does not enable (same operation as in Shift). • When this transition is preceded by a subroutine call/start step, execution waits for completion of the subroutine operation, and shifts to the lower step when the transition condition enables, or shifts to the right-connected step when the condition does not enable.
Jump	Jump	 (14)	JMP Pn	• Jumps to the specified pointer Pn (P0 to P16383) within its own program.
Pointer	Pointer	 (8)		• Indicates a jump destination pointer (label). • This pointer can be set at a step, transition, branch point or coupling point. • P0 to P16383 can be set in a single program. The same numbers may also be used in other programs.

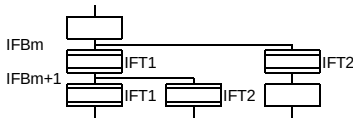
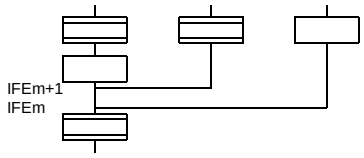
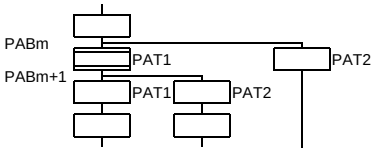
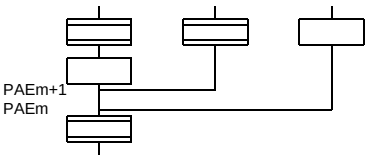
4.3 Branch and Coupling Chart List

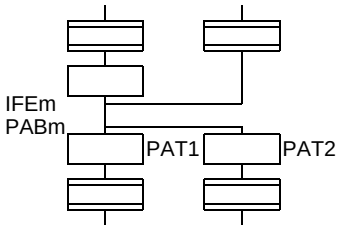
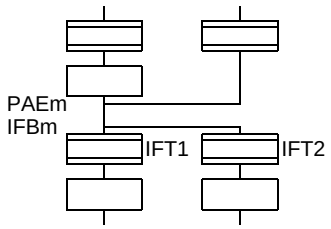
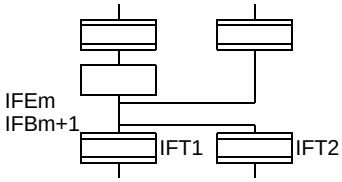
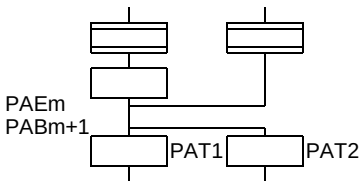
The following are branch and coupling patterns which specify step and transition sequences in SFC charts.

	Name (Code size (byte))	SFC Symbol	List Representation	Function
Basic type	Series transition (Corresponding symbol size)		List representation corresponding to SFC chart symbols shown in 4.2.	• Steps and transitions connected in series are processed in order from top to bottom. • Steps and transitions need not be lined up alternately. • When a transition is omitted, unconditional shift transition processing is performed.
	Selective branch ((Number of branches + 2) × 10)		CALL Kn IFBm IFT1 SFT Gn CALL Fn : JMP IFEm IFT2 SFT Gn' CALL Fn' : (JMP IFEm) IFEm CALL Fn''	• After the step or transition preceding a branch is executed, the route whose transition condition enables first is executed. • Selective branch destinations should always be started by transitions, all of which must be Shift or WAIT. (Using Shift and WAIT together will cause a parallel branch.)
	Selective coupling (8)			• After the route branched by a selective branch has been processed, execution shifts to a coupling point. • A coupling may be preceded and followed by either a step or a transition.
	Parallel branch (Number of branches × 22 + number of coupling points × 2 + 12)		SFT Gn PABm PAT1 CALL Fn SET Gn' : JMP PAEm PAT2 CALL Fn' SET Gn'' : (JMP PAEm) PAEm CALL Fn'' :	• Multiple routes (steps) connected in parallel are executed simultaneously. • Each parallel branch destination may be started by either a step or transition.
	Parallel coupling (8)			• Execution waits at the coupling point for executions of the routes branched by a parallel branch to be completed, and shifts to the next when executions of all routes are completed. A coupling may be preceded and followed by either a step or a transition. • When this coupling is preceded by an FS step, scans are executed during waiting. After waiting is complete, scans are not executed.
	Jump transition (Corresponding symbol size)	<Normal jump>  <Coupling jump> 	CALL Fn JMP Pn	1) Normal jump • After the step or transition preceding this jump transition is executed, execution shifts to the pointer Pn specified within its own program. • The jump destination may either be a step or transition. • When a jump takes place from an FS step to a transition, scans are executed during waiting for the transition condition of the jump destination to enable. 2) Coupling jump • When a jump to the other route within a parallel branch takes place after the parallel branch, a "coupling jump" takes place and execution waits at the jump destination.
		 	CALL Fn' Pn JMP Pn	

Combining the basic type branches/couplings provides the following application types, which are defined as in the basic types.

	Name	SFC Symbol	List Representation	Function
Application type	Selective branch | Parallel branch		CALL Kn IFBm IFT1 SFT Gn PABm PAT1 CALL Fn : JMP PAEm PAT2 CALL Fn' : (JMP IFEm) PAEm JMP IFEm IFT2 SET Gn' CALL Fn'' : (JMP IFEm) IFEm SET Gn''	• After a selective branch, a parallel branch can be performed.
	Parallel coupling | Selective coupling			• The selective coupling point can be the same as the coupling point of a parallel coupling for selective branch→parallel branch. Note that in an SFC chart, this type is displayed in order of a parallel coupling → a selective coupling, as shown on the left. • In this case, you cannot set a pointer (Pn) between the parallel coupling point (PAEm) and the selective coupling point (IFEm).
	Parallel branch | Selective branch		SFT Gn PABm PAT1 CALL Fn IFBm IFT1 SET Gn' CALL Fn' : JMP IFEm IFT2 SFT Gn'' CALL Fn'' : (JMP IFEm) IFEm JMP PAEm PAT2 CALL Fn''' : CALL Kn (JMP PAEm) PAEm SET Gn'''	• After a parallel branch, a selective branch can be performed.
	Selective coupling | Parallel coupling			• The parallel coupling point can be the same as the coupling point of a selective coupling for parallel branch→selective branch. Note that in an SFC chart, this type is displayed in order of a selective coupling → a parallel coupling, as shown on the left. • In this case, you cannot set a pointer (Pn) between the selective coupling point (IFEm) and the parallel coupling point (PAEm).

	Name	SFC Symbol	List Representation	Function
Application type	Selective branch | Selective branch		CALL Kn IFBm IFT1 SFT Gn IFBm+1 IFT1 SET Gn' : JMP IFEm+1 IFT2 SFT Gn'' : (JMP IFEm+1) IFEm+1 JMP IFEm IFT2 SFT Gn''' CALL Fn' : (JMP IFEm) IFEm SET Gn'''' :	• After a selective branch, a selective branch can be performed.
	Selective coupling | Selective coupling			• The two selective coupling points for selective branch→selective branch can be the same. Note that in an SFC chart, this type is displayed in order of a selective coupling → a selective coupling, as shown on the left. • In this case, you cannot set a pointer (Pn) between the selective coupling point (IFEm+1) and the selective coupling point (IFEm).
	Parallel branch | Parallel branch		SFT Gn PABm PAT1 CALL Fn PABm+1 PAT1 CALL Fn' : JMP PAEm+1 PAT2 CALL Fn'' : (JMP PAEm+1) PAEm+1 JMP PAEm PAT2 CALL Fn''' : CALL Kn JMP PAEm PAEm SET Gn''' :	• After a parallel branch, a parallel branch can be performed. • A parallel branch can be nested up to four levels.
	Parallel coupling | Parallel coupling			• The two parallel coupling points for parallel branch→parallel branch can be the same. Note that in an SFC chart, this type is displayed in order of a parallel coupling → a parallel coupling, as shown on the left. • In this case, you cannot set a pointer (Pn) between the parallel coupling point (PAEm+1) and the parallel coupling point (PAEm).

	Name	SFC Symbol	List Representation	Function
Appli-cation type	Selective coupling | Parallel branch		: (JMP IFEm) IFEm PABm PAT1 CALL Fn : JMP PAEm PAT2 CALL Fn' : (JMP PAEm) PAEm :	• The selective coupling point and parallel branch point can be the same. Note that in an SFC chart, this type is displayed in order of a selective coupling → a parallel branch, as shown on the left. • In this case, you cannot set a pointer (Pn) between the selective coupling point (IFEm) and the parallel branch point (PABm).
	Parallel coupling | Selective branch		: JMP PAEm PAEm IFBm IFT1 SET Gn : JMP IFEm IFT2 SET Gn' : (JMP IFEm) IFEm :	• The parallel coupling point and selective branch point can be the same. Note that in an SFC chart, this type is displayed in order of a parallel coupling → a selective branch, as shown on the left. • Execution waits at the parallel coupling point and shifts to the selective branch. • In this case, you cannot set a pointer (Pn) between the parallel coupling point (PAEm) and the selective branch point (IFBm).
	Selective coupling | Selective branch		: (JMP IFEm) IFEm IFBm+1 IFT1 SET Gn : JMP IFEm+1 IFT2 SET Gn' : (JMP IFEm+1) IFEm+1	• The selective coupling point and selective branch point can be the same. Note that in an SFC chart, this type is displayed in order of a selective coupling → a selective branch, as shown on the left. • In this case, you cannot set a pointer (Pn) between the selective coupling point (IFEm) and the selective branch point (IFBm+1).
	Parallel coupling | Parallel branch		: (JMP PAEm) PAEm PABm+1 PAT1 CALL Fn : JMP PAEm+1 PAT2 CALL Fn' : (JMP PAEm+1) PAEm+1 :	• The parallel coupling point and parallel branch point can be the same. Note that in an SFC chart, this type is displayed in order of a parallel coupling → a parallel branch, as shown on the left. • Execution waits at the parallel coupling point and shifts to the parallel branch. • In this case, you cannot set a pointer (Pn) between the parallel coupling point (PAEm) and the parallel branch point (PABm+1).

4.4 SFC Program Names

Set "SFC program names" to SFC program No. 0 to No. 255 individually. (Make this setting in the "SFC program management window" on the SFC program edit screen.)
Set an SFC program name within 16 characters. Specify this SFC program name for a "subroutine call/start step (GSUB)" and "clear step (CLR)".
SFC programs correspond to No. 0 to No. 255 and saved in a one program-for-one file format. The preset "SFC program name" is used as the file name of the SFC program file for user file management. (Refer to Chapter 14 for full information.)

POINT
(1) You can set an SFC program to any of No. 0 to No. 255. There are no specific programs which have special roles. (2) You cannot use "$" in the first character of an SFC program name. (3) You cannot use " ¥ : ; , . * ? “ < > |" in SFC program names.

4.5 Steps

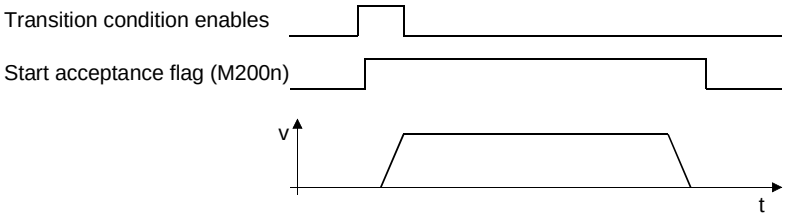
4.5.1 Motion control step

Name	Symbol	Function
Motion control step	Kn	Starts a servo program Kn. Specifying range: K0 to K4095

[Operations]

(1) The start acceptance flag of the axis specified in the specified servo program Kn (n = 0 to 4095) turns ON.
(2) The specified servo program Kn (n = 0 to 4095) starts.

Execution timing



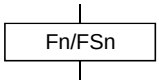
[Errors]

(1) The absence of the specified servo program Kn will result in an SFC program error 16200 and stop the SFC program running at the point of error detection.

[Instructions]

(1) To make a current value change in the SFC program, specify the CHGA instruction in the servo program and call it at the motion control step.
(2) If the servo program has stopped due to a major/minor error which occurred at or during a start of the servo program specified at the motion control step, the SFC program continues running. To stop the SFC program at error detection, provide an error detection condition at the transition (transition condition).

4.5.2 Operation control step

Name	Symbol	Function
Operation control step		Runs an operation control program Fn/FSn. Specifying range: F0 to F4095/FS0 to FS4095

[Operations]

(1) Once execution type operation control step Fn
Fn runs the specified operation control program Fn (n = 0 to 4095) once.
(2) Scan execution type operation control step FSn
FSn repeats the specified operation control program FSn (n =0 to 4095) until the next transition condition enables.

[Errors]

(1) The absence of the specified operation control program Fn/FSn will result in an SFC program error 16201 and stop the SFC program running at the point of error detection.

[Instructions]

(1) For operation expressions that may be described in operation control programs, refer to Chapter 5 Operation Control Programs.
(2) The SFC program continues running if an operation or similar error occurs during execution of the operation control program.

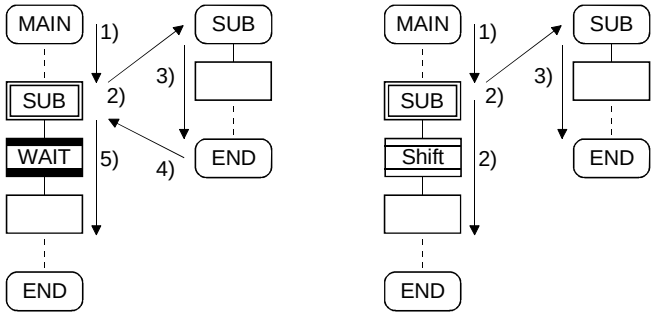
4.5.3 Subroutine call/start step

Name	Symbol	Function
Subroutine call/start step	Program name	Calls/starts the SFC program of the specified program name.

(1) Calls/starts the SFC program of the specified program name.

(2) Control varies with the type of the transition coupled next to the subroutine call/start step.
   (a) For WAIT
   A subroutine call is performed. When the subroutine call step is executed, control shifts to the specified program as shown below, and when END of the called program is executed, control returns to the call source program.

   (b) For other than WAIT
   A subroutine start is performed. When the subroutine start step is executed, control starts the specified program and then shifts to the next as shown below. Hence, the start source and destination SFC programs are run in parallel. The started program ends when END is executed.



For WAIT　　For other than WAIT

[Errors]

(1) The absence of the specified SFC program at a subroutine call/start will result in an SFC program error 16005 and stop the call/start source SFC program running at the point of error detection.

(2) If the called/started SFC program is already starting at a subroutine call/start, an SFC program error 16006 will occur and the call/start source SFC program running is stopped at the point of error detection.

(3) Calling/starting its own program at a subroutine call/start will result in an SFC program error 16110 and stop the call/start source SFC program running at the point of error detection.

(4) When the subroutine to be called/started at a subroutine call/start in the SFC program 2 which was called/started from the SFC program 1 is the SFC program 1 (main program), an SFC program error 16111 will occur and the call/start source SFC program 2 running is stopped at the point of error detection.

[Instructions]

(1) There are no restrictions on the depth of subroutine call/start nesting.

(2) For a subroutine start, the start source SFC program continues processing if the start destination SFC program stops due to an error.

(3) For a subroutine call, the call source SFC program stops running as soon as the call destination SFC program stops due to an error.

4.5.4 Clear step

Name	Symbol	Function
Clear step	CLR Program name	Stops the running SFC program of the specified program name.

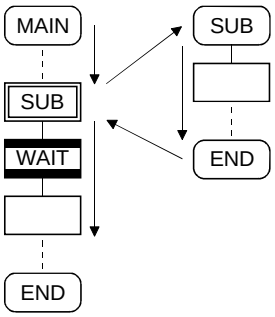
[Operations]

(1) Stops the specified SFC program running.

(2) After stopped, the clear-specified SFC program will not start automatically if it has been set to start automatically.

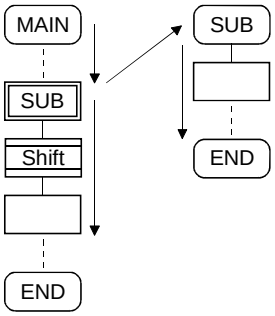
(3) The specified program may be its own program.

(4) If the specified program is being subroutine called, the subroutine program called is also stopped. (See below)



If the program has been "subroutine called" as shown on the left
- When the main program (MAIN) is cleared
  Even if the subroutine (SUB) is running, both the main program(MAIN) and subroutine (SUB) stop running.
- When the subroutine (SUB) is cleared
  If the subroutine (SUB) is running, the subroutine (SUB) stops running and execution returns to the main program (MAIN).

(5) When the specified program has been subroutine started, the subroutine program started continues processing. (See below)



If the program has been "subroutine started" as shown on the left
- When the main program (MAIN) is cleared
  Even if the subroutine (SUB) is running, the main program (MAIN) stops running but the started subroutine (SUB) continues processing.
- When the subroutine (SUB) is cleared
  If the subroutine (SUB) is running, only the subroutine (SUB) stops running.

(6) When the servo program started from the specified program is starting, the servo program continues processing.

[Errors]

(1) The absence of the SFC program specified at the clear step will result in an SFC program error 16203.

[Instructions]

(1) When the SFC program specified at the clear step is not starting, an error does not occur specifically and this step is ignored.

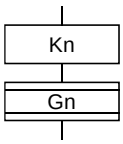
(2) If the SFC program running is stopped by the clear step, the output is held.

4.6 Transitions

You can describe conditional and operation expressions at transitions. The operation expression described here is repeated until the transition condition enables, as at the scan execution type operation step.
For the conditional/operation expressions that can be described in transition conditions, refer to Chapter 6 Transition Programs.

(1) Combinations with motion control steps

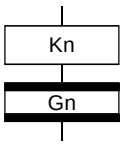
(a) Motion control step + shift



[Operations]

- Does not wait for the servo program Kn started at the motion control step to complete its operation, and shifts to the next step when the transition condition Gn enables.

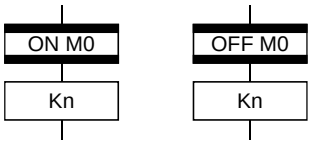
(b) Motion control step + WAIT



[Operations]

- Waits for the servo program Kn started at the motion control step to complete its operation, and shifts to the next step when the transition condition Gn enables.
- The operation completion condition of the servo program Kn is not needed in the transition condition Gn.
- An error stop of the started servo program Kn at/during a start is also regarded as an operation completion.

(c) WAITON/WAITOFF + motion control step



[Operations]

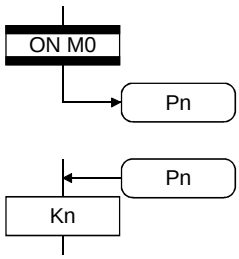
- Prepares for the start of the motion control step next to WAITON/WAITOFF, and makes a start immediately when the specified bit turns ON/OFF. When the motion control step is executed without being used with WAITON/WAITOFF, preparations for a start are made after the transition condition preceding the motion control step enables. This will cause a variation of delay/starting time between when the transition condition enables and when a start is made, but a combination with WAITON/WAITOFF can eliminate the variation of the above delay/starting time.

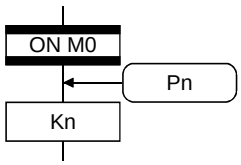
- Specifiable bit devices

	A172SHCPUN	A173UHCPU(-S1)/ A273UHCPU-S3
X	X0 to X7FF	X0 to X1FFF
Y	Y0 to Y7FF	Y0 to Y1FFF
M	M0 to M2047	M0 to M8191
Special M	M9000 to M9255	M9000 to M9255
L	L0 to L2047	L0 to L8191
B	B0 to B3FF	B0 to B1FFF
F	F0 to F255	F0 to F2047
TC (timer coil)	TC0 to TC255	TC0 to TC2047
TT (timer contact)	TT0 to TT255	TT0 to TT2047
CC (counter coil)	CC0 to CC255	CC0 to CC1023
CT (counter contact)	CT0 to CT255	CT0 to CT1023

[Instructions]

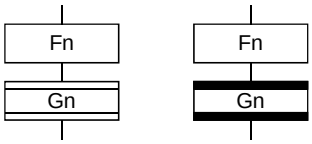
- Always pair a transition with a motion control step one-for-one. If the step following WAITON/WAITOFF is not a motion control step, an SFC program error 16102 will occur and the SFC program running will stop at the point of error detection.
- An error will not occur if the jump destination immediately after WAITON/WAITOFF is a motion control step. (Left below)
- A pointer may exist immediately after WAITON/WAITOFF. (Right below)





- If a servo program specified at a motion control step could not be started due to a major/minor error, an SFC program continues running and execution shifts to the next, independently of the WAITON/WAITOFF bit device status. To stop the SFC program at error detection, provide an error detection condition at the next transition (transition condition).

(2) Combination with operation control step



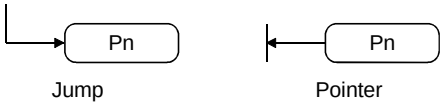
[Operations]

- At an operation control step, both Shift and WAIT perform the same operation, and after an operation control program Fn is run, execution shifts to the next step when the transition condition Gn enables.

(3) Combination with subroutine call/start step

Refer to the section of 4.5 (3) Subroutine call/start step.

4.7 Jump, Pointer

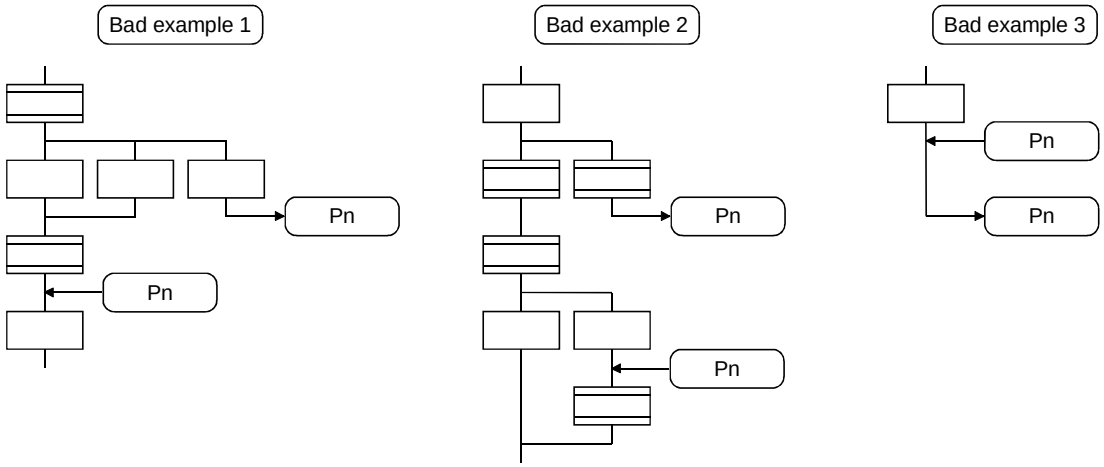


[Operations]

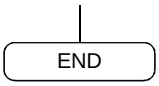
- Setting a jump will cause a jump to the pointer Pn specified in its own program.
- You can set pointers at steps, transitions, branch points and coupling points.
- You can set pointers Pn, P0 to P16383, in a single program.

[Instructions]

- You cannot make a jump setting which will exit from within parallel branch-parallel coupling. (Bad example 1 given below)
- You cannot make a jump setting from outside parallel branch-parallel coupling to within parallel branch-parallel coupling. (Bad example 2 given below)
- You cannot make a setting where a label and a jump will continue. (Bad example 3 given below)



4.8 END



[Operations]

- Ends a program.
- Making a subroutine call will return to the call source SFC program.

[Instructions]

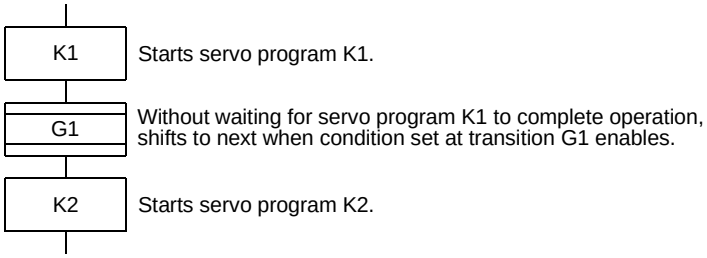
- END may be set a multiple number of times within a single program.
- END cannot be specified between a parallel branch and a parallel coupling.
- The output is held after the SFC program is ended by END.

4.9 Branches, Couplings

4.9.1 Series transition

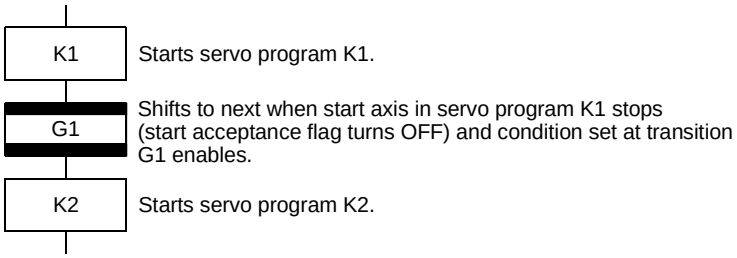
Shifts execution to the subsequent step or transition connected in series.

(1) To start a servo program or subroutine and shift execution to the next without waiting for operation completion
Set Shift at a transition.
In this case, the transition (shift) may be omitted.
When you omitted the transition, an unconditional shift transition is performed.



POINT
When a subroutine start is made, it own program and a subroutine program are processed in parallel.

(2) To start a servo program or subroutine and proceed to the next step on operation completion
Set WAIT at a transition.

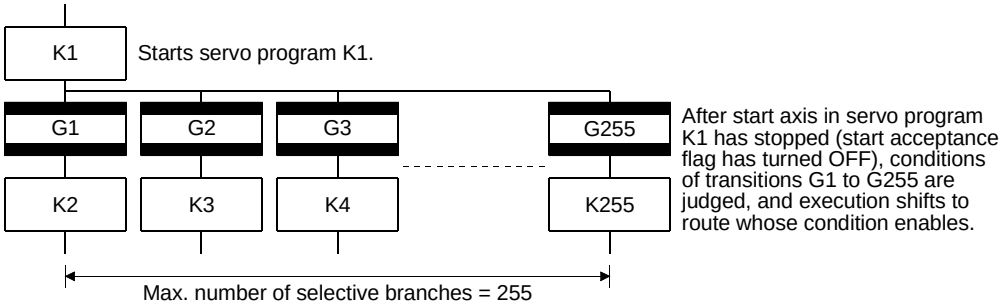


POINT
(1) The above start acceptance flag of the axis started in the next servo program K2 is not included in interlocks. To use it as an interlock, the user should set it in the transition condition G1. (2) WAIT must be set to proceed to the next step on operation completion. However, when there are specifically no conditions to be set as interlocks, set "NOP (No Operation)" in the transition program (Gn).

4.9.2 Selective branch, selective coupling

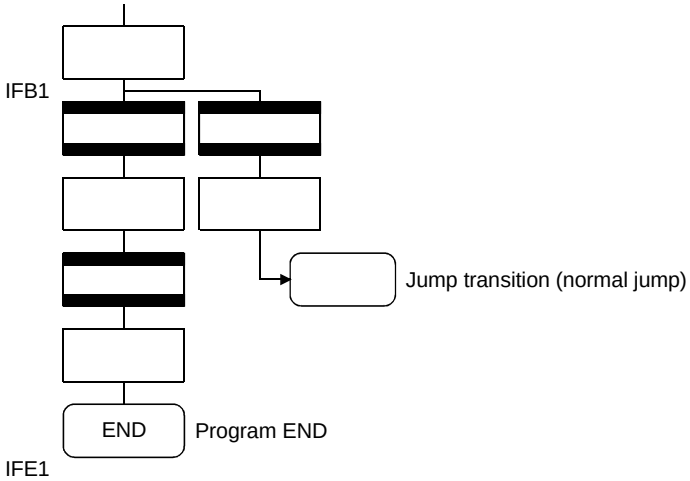
(1) Selective branch
Executes only the route whose condition was judged to have enabled first among the conditions of multiple transitions connected in parallel.
Transitions must be all Shifts or WAITs.

(Example) For WAIT



POINT
(1) Transition condition judgment is not always executed from left to right.
(2) Using Shift and WAIT together will cause a parallel branch.

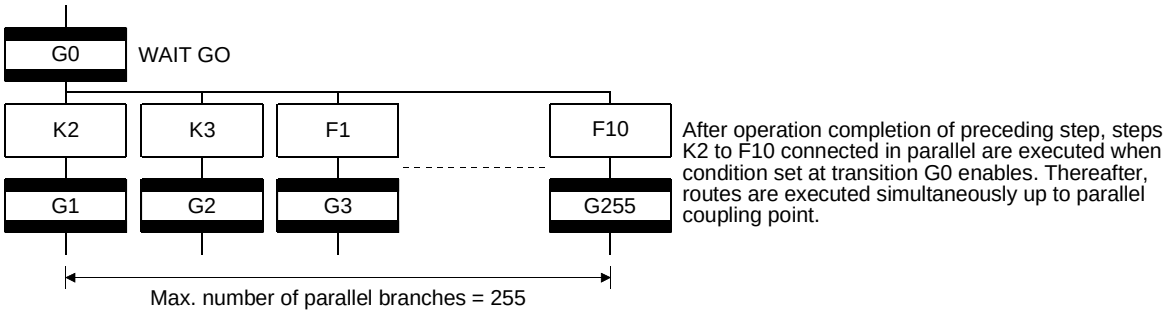
(2) Selective coupling
Recoupling of routes into a single route after their processing completions following a selective branch will be a selective coupling. However, you can also make a setting where no coupling will be made as shown below.



4.9.3 Parallel branch, parallel coupling

(1) Parallel branch

Simultaneously executes multiple steps connected in parallel. A parallel branch destination may be started by either a step or a transition.



POINT
"Shift" or "WAIT" can be set to a transition preceding a parallel branch. "WAITON" and "WAITOFF" cannot be set.

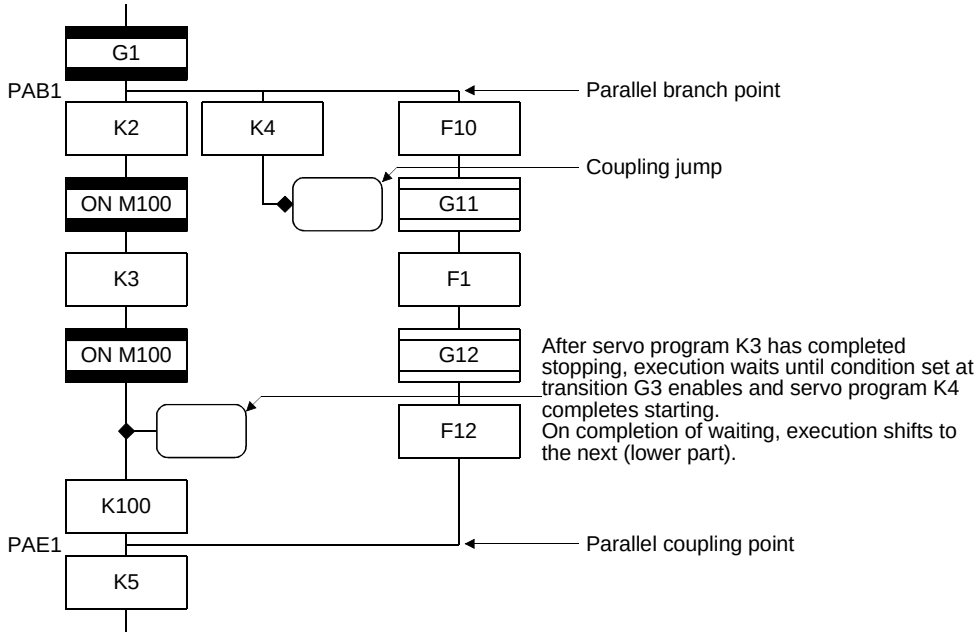
(2) Parallel coupling

A parallel branch must be coupled by a parallel coupling.

A jump setting to another branch route can be made within parallel branch-parallel coupling.

In this case, a jump destination is a midway parallel coupling point (coupling jump).

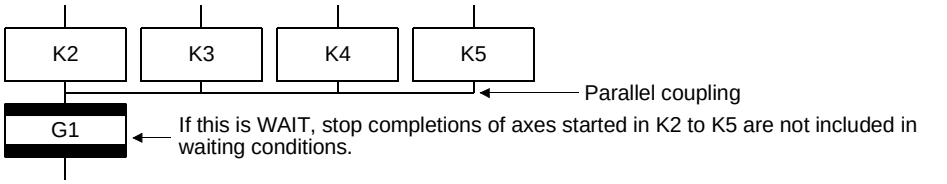
You cannot set a jump to exit from within parallel branch-parallel coupling.



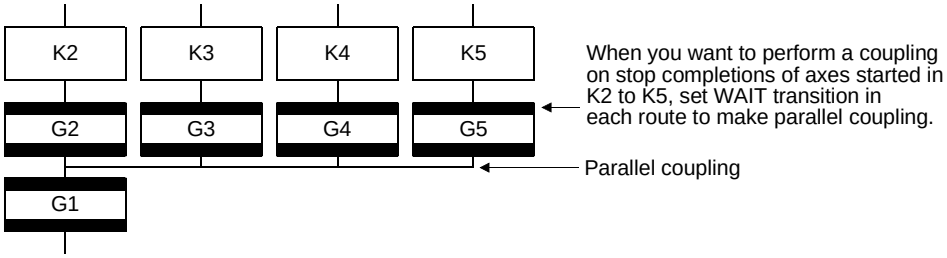
POINT
The number of parallel branches need not match that of couplings at a parallel coupling point. (In the example of the diagram in Section 4.9.3 (2), the number of parallel branches is 3 and that of couplings is 2.)

When a WAIT transition is set right after a parallel coupling, the stop completions of the axes are not included in the waiting conditions if the parallel coupling is preceded by motion control steps. To perform a parallel coupling on stop completions, set WAIT transitions before a parallel coupling.

(×)



(○)



4.10 Y/N Transitions

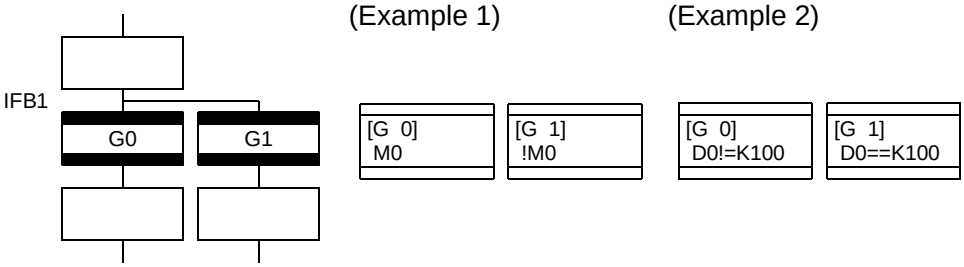
When you want to branch to routes when a transition condition enables and does not enable, "Shift Y/N transition" or "WAIT Y/N transition" will be useful.

Name	Symbol	Function
Shift Y/N transition	(When condition does not enable) Gn N (When condition enables) Y	• When a transition condition set at Gn enables, execution shifts to the lower step. When that condition does not enable, execution shifts to the right-connected step. • Differences between "Shift Y/N" and "WAIT Y/N" are the same as those between "Shift" and "WAIT".
WAIT Y/N transition	(When condition does not enable) Gn N (When condition enables) Y	

A Y/N transition is designed to describe the following two-route selective branch program easily.

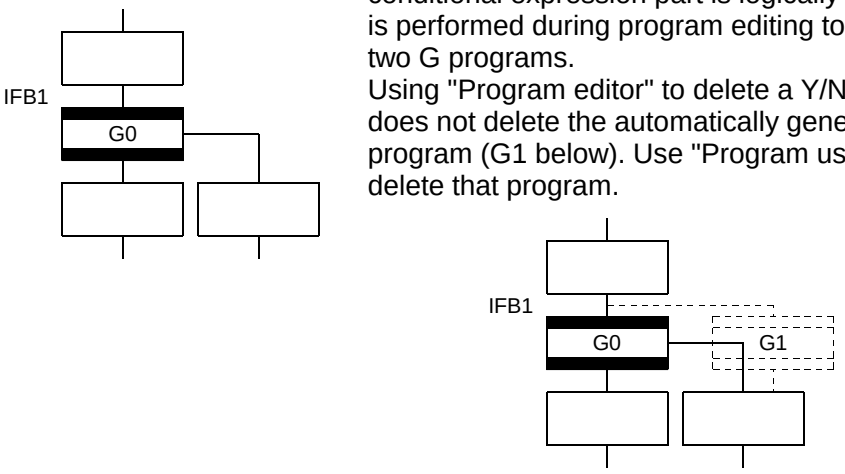
<Conventional description>

• G0 and G1 programs should be different only in acknowledgement/negation of the conditional expressions.



<New description>

• As a G0 program, set the G0 program shown in above (Example 1) or (Example 2).
• The SFC program list codes after conversion are the same as in the conventional description (different only in SFC chart representation). Therefore, "automatic search for free G number → automatic generation of program whose conditional expression part is logically negated" is performed during program editing to occupy two G programs.
Using "Program editor" to delete a Y/N transition does not delete the automatically generated G program (G1 below). Use "Program use list" to delete that program.



(1) Automatic free G number search feature

(a) When not set to automatic numbering

Searches for a free number forward, starting with the "set G number + 1" at the "Shift Y/N" or "WAIT Y/N" symbol.

When no free numbers are found after a search up to 4095, a search is made from 0 to the "set G number - 1".

(b) When set to automatic numbering

Searches for a free number forward (or backward) in the automatic numbering range, starting with the "automatically numbered G number + 1 (or -1)" at the "Shift Y/N" or "WAIT Y/N" symbol. (The searching method is as in the automatic numbering setting.)

(2) Automatic logical NOT program generation feature

Automatically generates a program which logically negates the conditional expression block (last block) of the transition program set at "Shift Y/N" or "WAIT Y/N".

The basic is as described below.

<Set program (conditional expression block)>

Conditional expression//(bit conditional expression or comparison conditional expression)

<Logically negated, automatically generated program (conditional expression block)>

!Conditional expression//(bit conditional expression or comparison conditional expression)

Examples are given below.

<Set program (conditional expression block)>

(Example 1)

M0	//Bit device ON

(Example 2)

D0!=K100	//Data register D0 is not K100

<Logically negated, automatically generated program (conditional expression block)>

(Example 1)

!(M0)	//Bit device OFF

(Example 2)

!(D0!=K100)	//Data register D0 is K100

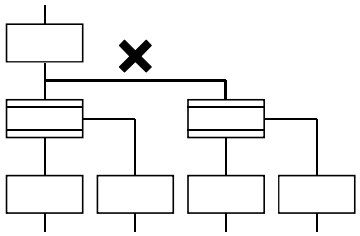
POINT
• For the instructions usable in the conditional expressions of "Shift Y/N" or "WAIT Y/N" transition programs, refer to "Section 3.3.2 (1) Operation control/transition instruction list".

(3) Instructions for SFC charts

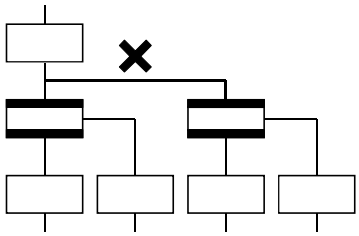
Any SFC chart that will be meaningless to or conflict with the definition of Y/N transitions will result in an error at the time of editing (or SFC chart conversion). Their patterns and instructions will be given below.

(a) When "Shift Y/N" or "WAIT Y/N" is connected as a selective branch or parallel branch: Error

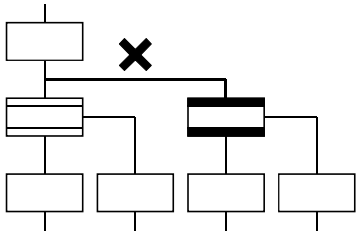
- "Shift Y/N" used as selective branch



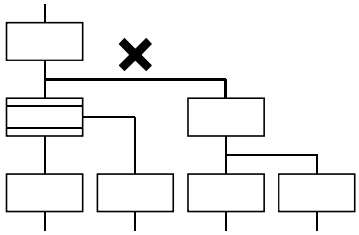
- "WAIT Y/N" used as selective branch



- "Shift Y/N" and "WAIT Y/N" used as parallel branch

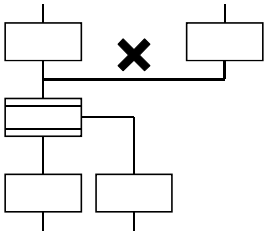


- "Shift (or WAIT) Y/N" used with other step/transition as parallel branch or selective branch

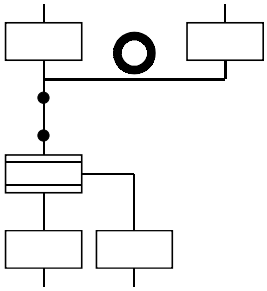


(b) When a coupling precedes "Shift Y/N" or "WAIT Y/N: Provide "coupling-branch continuation" in between.

- Direct coupling with "Shift Y/N" or "WAIT Y/N" is not allowed.

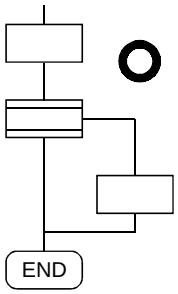


- "Provide "coupling-branch continuation" in between.

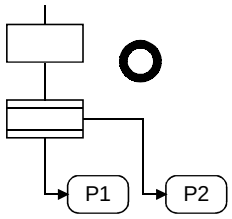


(c) The following patterns may be set.

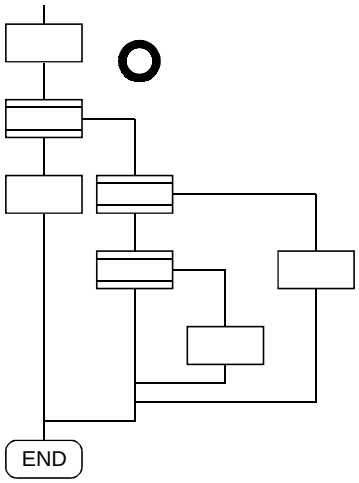
- End (END) from "Shift Y/N" or "WAIT Y/N"



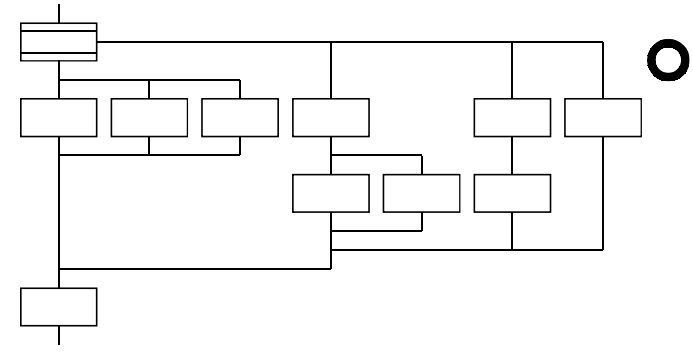
- Jump from "Shift Y/N" or "WAIT Y/N"

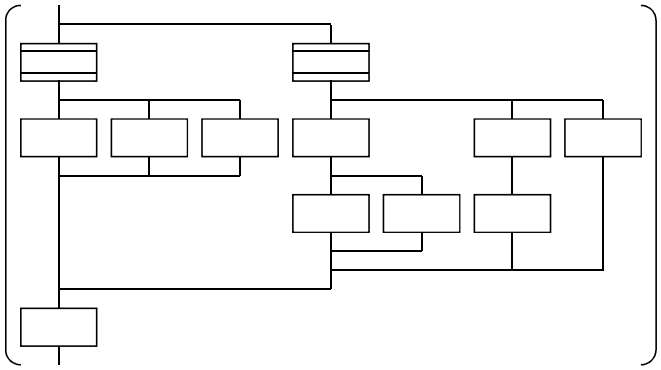


- Continuation from "Shift Y/N" or "WAIT Y/N" to "Shift Y/N" or "WAIT Y/N" (selective branch-selective branch)



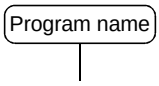
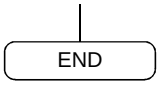
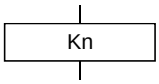
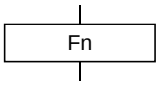
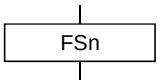
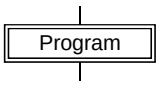
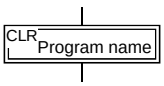
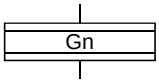
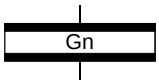
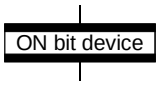
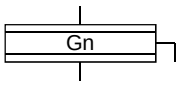
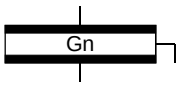
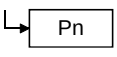
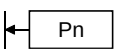
- When there are two or more connection lines from Y/N side of "Shift Y/N" or "WAIT Y/N", selective branch→selective branch or parallel branch continues.





4.11 SFC Comments

A comment can be set to each symbol of the step/transition in an SFC chart.
Comments are shown in an SFC chart by changing the display mode to "Comment display" on the SFC program edit screen.
Since SFC comments are stored into the CPU code area, performing read from PLC displays the SFC chart with comments.

Classification	Name	Symbol	Comment Setting
Program start/end	START		Comment setting cannot be made.
	END		
Step	Motion control step		Max. 80 characters Displayed in 20 characters × 4 lines
	Once execution type operation control step		
	Scan execution type operation control step		
	Subroutine call/start step		
	Clear step		
Transition	Shift (preread transition)		
	WAIT		
	WAITON		
	WAITOFF		
	Shift Y/N		
	WAIT Y/N		
Jump	Jump		Max. 64 characters Displayed in 16 characters × 4 lines
Pointer	Pointer		

POINT

(1) SFC comments are stored into the CPU code area.
The CPU code area stores the SFC chart codes, operation control (F/FS) program codes, transition (G) program codes and SFC comments.
Be careful not to set too many comments to avoid code area overflow.
(Refer to "3.1 SFC Performance Specifications" for the code area sizes.)

(2) You cannot use "," in comment statements.

5. OPERATION CONTROL PROGRAMS

5.1 Operation Control Programs

(1) Operation control programs

(a) In operation control programs, you can set assignment operation expressions, motion-dedicated functions and bit device control commands.

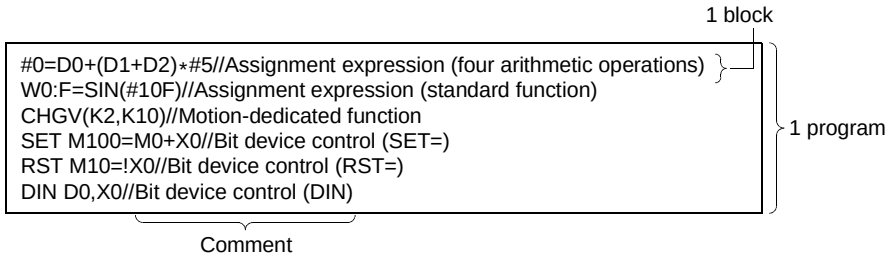
(b) You can set multiple blocks in a single operation control program.

(c) There are no restrictions on the number of blocks that may be set in a single operation control program. However, one program is within 64k bytes.

(d) The maximum number of characters in one block is 128.

(e) You cannot set transition conditions.
Transition conditions may be set only in transition programs.

An operation control program example is given below.



(2) Priorities of operators and functions

Operators and functions have the following priorities.
Using parentheses allows an operation sequence to be specified freely.

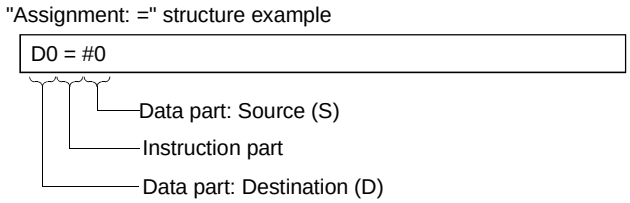
Priority	Item (Operator, Function)
High ↕ Low	Calculation within parentheses ((...))
	Standard function (SIN, COS, etc.), Type conversion (USHORT, LONG, etc.)
	Bit inversion (~), logical negation (!), sign inversion (-)
	Multiplication (*), division (/), remainder (%)
	Addition (+), subtraction (-)
	Bit left shift (<<), bit right shift (>>)
	Comparison operators: Less than (<), less than or equal to (<=), more than (>), more than or equal to (>=)
	Comparison operators: Equal to (==), not equal to (!=)
	Bit logical AND (&)
	Bit exclusive OR (^)
	Bit logical OR (|)
	Logical AND (*)
	Logical OR (+)
	Assignment (=)

(3) Instruction structure

Many of the instructions usable in operation control programs can be divided into instruction and data parts.

The instruction and data parts are used for the following purposes.

- Instruction part..... Indicates the function of that instruction.
- Data part..... Indicates the data used in the instruction.



(a) Source (S)

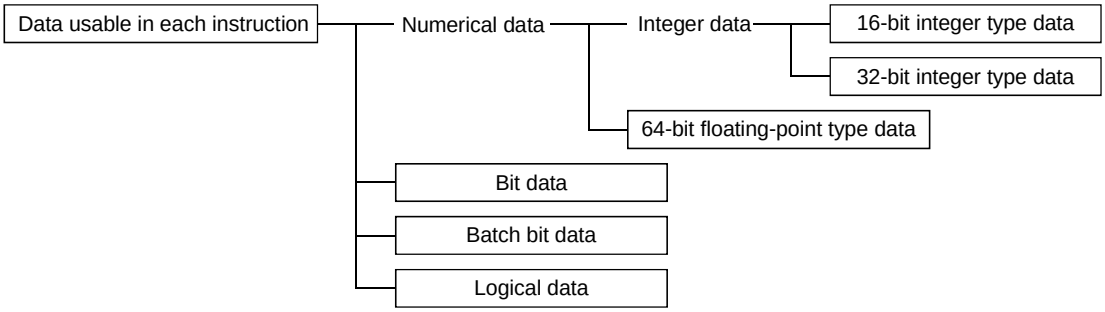
1) The source is the data used in an operation.
2) It varies with the device specified in each instruction, as described below.
   - Bit or word device
     Specify the device which stores the data used in operation.
     The data must have been stored in the specified device until the operation is executed.
     Changing the data stored in the specified device during program run allows changing the data used in that instruction.
   - Constant
     Specify the numerical value used in an operation.
     As the constant is set during program creation, it cannot be changed during program run.

(b) Destination (D)

1) As the destination data, after-operation data is stored.
2) To the destination data, always set the device for storing the data.

(4) How to specify data

There are the following six different data usable in each instruction.



(a) 16-bit integer type data

The 16-bit integer type data is 16-bit integral value data.
Word devices are used in increments of 1 point.
Data ranges are as indicted below.

	Decimal Representation	Hexadecimal Representation
Data range	K-32768 to K32767	H0000 to HFFFF

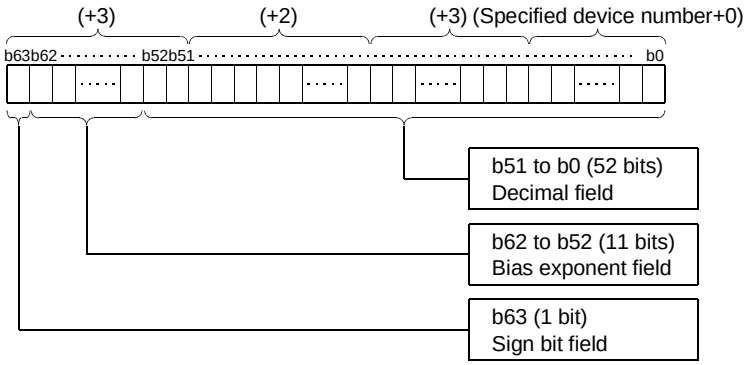
(b) 32-bit integer type data

The 32-bit integer type data is 32-bit integral value data.
Word devices are used in increments of 2 points: (specified device number), (specified device number+1). Data ranges are as indicted below.

	Decimal Representation	Hexadecimal Representation
Data range	K-2147483648L to K2147483647L	H00000000L to HFFFFFFFFL

(c) 64-bit floating-point type data

The 64-bit floating-point type data is IEEE-formatted, 64-bit floating-point value data.
Word devices are used in increments of 4 points: (specified device number), (specified device number+1), (specified device number+2), (specified device number+3).
1) The internal bit locations are as shown below.



2) The represented value is as follows. (The bias value is H3FF.)
$(-1)^{[\text{Sign bit field}]} * (1.0+[\text{decimal field}]) * 2^{([\text{Bias exponent field}]-[\text{bias value}])}$

3) Data ranges are as indicted below.

	Decimal Representation	Hexadecimal Representation
Data range	K-1.79E+308 to K-2.23E-308, K0.0, K2.23E-308 to K1.79E+308	H0000000000000000, H0010000000000000 to H7FE1CCF385EBC89F, H8000000000000000, H8010000000000000 to HFFE1CCF385EBC89F

4) A round-off error may be produced in a 64-bit floating-point type data operation. Especially when using 64-bit floating-point type data in a comparison operation, note that a round-off error may cause an intended operation.

Example) In the following transition program, the result of the comparison operation may not become true depending on the value of #200F due to a round-off error.

```
#100F=SQRT(#200F)
#300F=#100F*#100F
#200F==#300F
```

(d) Bit data

The bit data is the data where a contact/coil or similar device is handled in increments of 1 bit. It is used in device set (SET=) and device reset (RST=).

Example

```
SET M0
```

Bit data

(e) Batch bit data

The batch bit data is the data where bit data is handled in increments of 16/32 points. It is used in device input (DIN) and device output (DOUT). As indicated below, whether the bit data is handled in increments of 16 or 32 points is governed by the data type of the word device used as an input destination/output source.

	Increments of 16 Points	Increments of 32 Points
Program example	DIN #0, M0 DOUT M0, D0	DIN #0L, M0 DOUT M0, DOL
Used devices	(Specified device number) to (specified device number+15) M0 to M15 in the above program example	(Specified device number) to (specified device number+31) M0 to M31 in the above program example

(f) Logical data

The logical data is a value returned by a bit or comparison conditional expression and indicates whether the result is true or false.
Normally, it is used in the conditional expression of a transition program. In an operation control program, the logical data is used in a bit conditional expression set to device set (SET=) or device reset (RST=).

Example 1

```
SET M0 = X10
```

Logical data

Bit data

Example 2

```
RST M5 = !X10*M100
```

Logical data

Bit data

Example 3 (transition program)

```
D0 == K100
```

Logical data

5.2 Device Descriptions

Word and bit device descriptions are indicated below.

(1) Word device descriptions

	Device Descriptions			Device Number (n) Specifying Ranges	
	16-bit integer type	32-bit integer type (n is even number)	64-bit floating-point type (n is even number)	A172SHCPUN	A173UHCPU(-S1) /A273UHCPU-S3
Data register	Dn	DnL	DnF	0 to 1023	0 to 8191
Link register	Wn	WnL	Wn:F	0 to 3FF	0 to 1FFF
Timer current value	Tn	—	—	0 to 255	0 to 2047
Counter current value	Cn	—	—	0 to 255	0 to1023
Special register	Dn	DnL	DnF	9000 to 9255	
Motion device	#n	#nL	#nF	0 to 8191	
Coasting timer	—	FT	—	—	

(a) For differentiation, the 32-bit floating-point type is ended by L and the 64-bit floating-point type by F (:F for the link register).

(b) The timer current value T and counter current value C may be used only as a 16-bit integer type.

(c) For the 32-bit integer type and 64-bit floating-point type, specify the device number with an even number. (You cannot use an odd number for setting).

(d) The coasting timer FT is incremented per 888μs. (The coasting timer is a 32-bit integer type.)

(2) Bit device descriptions

	Device Description	Device Number (n) Specifying Ranges	
		A172SHCPUN	A173UHCPU(-S1) /A273UHCPU-S3
Input relay	Xn/PXn	0 to 7FF	0 to 1FFF
Output relay	Yn/PYn	0 to 7FF	0 to 1FFF
Internal relay	Mn	0 to 2047	0 to 8191
Latch relay	Ln	0 to 2047	0 to 8191
Link relay	Bn	0 to 3FF	0 to 1FFF
Annunciator	Fn	0 to 255	0 to 2047
Timer contact	TTn	0 to 255	0 to 2047
Timer coil	TCn	0 to 255	0 to 2047
Counter contact	CTn	0 to 255	0 to 1023
Counter coil	CCn	0 to 255	0 to 1023
Special relay	Mn	9000 to 9255	

(a) When using the device in DIN or DOUT as batch bit data, specify n as a multiple of 16.

(3) Indirect designation of device numbers

In the above word/bit device descriptions, device numbers (n) can be specified indirectly.

(a) Using word device to specify device number (n) indirectly

1) You cannot use the word device with which the device number was specified indirectly.

2) You can use the 16- and 32-bit integer type word devices for indirect designation.
You cannot use the 64-bit floating-point type.
(Description examples)

Good Example	Bad Example
#(D10)	#(D(D5))
D(#10L)F	D(#4F)

(b) Using operation expression to specify device number indirectly

1) Device numbers can be specified indirectly by calculation expressions which use the following data and operators.

Usable data	16-bit integer type word device
	32-bit integer type word device
	16-bit integer type constant
	32-bit integer type constant
Usable operators	Addition: +
	Subtraction: -
	Multiplication: *
	Division: /
	Remainder: %
	Sign inversion: -

2) You cannot use the word device with which the device number was specified indirectly.

3) Only one operator may be used.
(Description examples)

Good Example	Bad Example
#(D10-K5)	#(D(D5)F+K20)
D(#10L%H6L)F	D(#4L<<K2)

(Note) : When you want to use the result of calculation other than the above to specify the device number indirectly, describe it in two blocks as shown below.

```
D0=SHORT(ASIN(#0F))
W0=#(D0)
```

5.3 Constant Descriptions

The constant descriptions of the 16-bit integer type, 32-bit integer type and 64-bit floating-point type are indicated below.

	16-Bit Integer Type	32-Bit Integer Type	64-Bit Floating-Point Type
Decimal representation	K-32768 to K32767	K-2147483648L to K2147483647L	K-1.79E+308 to K-2.23E-308, K0.0, K2.23E-308 to K1.79E+308
Hexadecimal representation	H0000 to HFFFF	H00000000L to HFFFFFFFFL	—

(1) The 32-bit integer type is ended by L and the 64-bit floating-point type is provided with a decimal point and exponent part (E) to denote their data types explicitly.

(2) The constant without the data type is regarded as the applicable minimum type.

(3) The constant in decimal representation is headed by K and the one in hexadecimal representation by H.
K can be omitted.

(4) The 64-bit floating-point type cannot be represented in hexadecimal.

5.4 Binary Operations

5.4.1 Substitution : =

F/FS	G
○	○

Format	Number of Basic Steps
(D)=(S)	4

(1) Usable data

Data to be Set	Usable Data										
		Word device				Constant					
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(S)	—	○	○	○	○	○	○	○	○	—	—
(D)	—	○	○	○	—	—	—	—	—	—	—

(Note): T and C are write-disabled and cannot be used at (D).

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Word device/constant/calculation expression to be assigned	Data type of (D)
(D)	Word device which will store the operation result	

(3) Functions

(a) The data value specified at (S) is assigned to the specified word device.

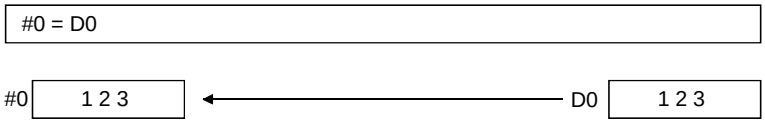
(b) When (S) and (D) differ in data type, the data at (S) is converted into the data type of (D) and the resultant data is assigned.
(When (D) is a 16- or 32-bit integer type and (S) is a 64-bit floating-point type, the fraction part of (S) is discarded.)

(4) Errors

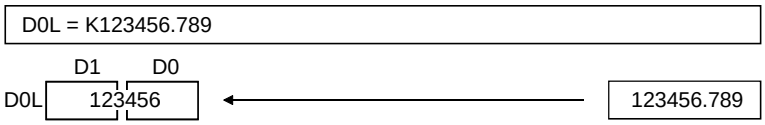
(a) An operation error will occur if:
1) The data at (S) is outside the data type range of (D); or
2) (D) or (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which assigns the D0 value to #0

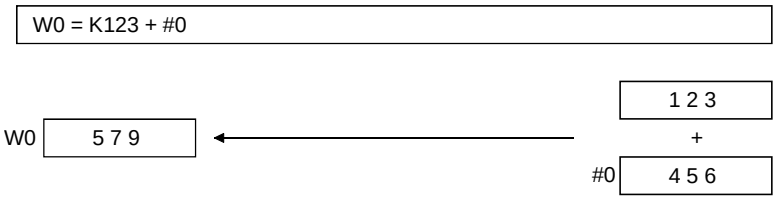


(b) Program which assigns K123456.789 to D0L



The 64-bit floating-point type is converted into the 32-bit integer type and the result is assigned.

(c) Program which assigns the result of adding K123 and #0 to W0



5.4.2 Addition : +

F/FS	G
○	○

Format	Number of Basic Steps
(S1)+(S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	○	○	○	○	○	○	—	—
(S2)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Augend data	Data type of (S1) or (S2) which is greater
(S2)	Addend data	

(3) Functions

(a) The data specified at (S2) is added to the data specified at (S1).

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before operation is performed.

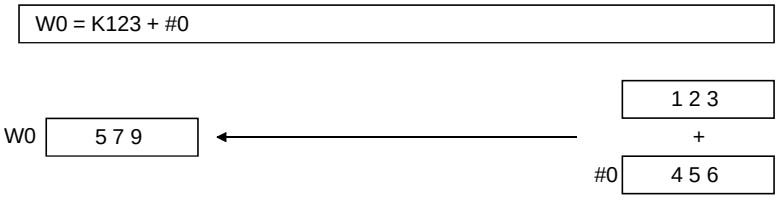
(4) Errors

(a) An operation error will occur if:

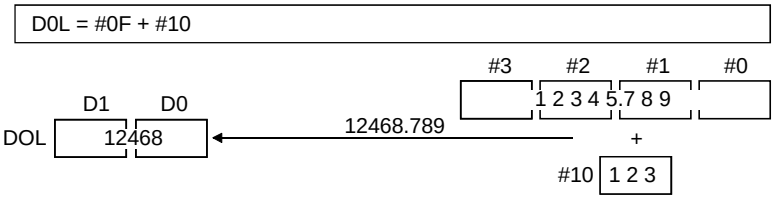
1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which assigns the result of adding K123 and #0 to W0



(b) Program which assigns the result of adding #0F and #10 to D0L



The 64-bit floating-point type data are used for addition, and the result is converted into the 32-bit integer type and then assigned.

5.4.3 Subtraction : -

F/FS	G
○	○

Format	Number of Basic Steps
(S1)-(S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	○	○	○	○	○	○	—	—
(S2)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Minuend data	Data type of (S1) or (S2) which is greater
(S2)	Subtracted data	

(3) Functions

(a) The data specified at (S2) is subtracted from the data specified at (S1).

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before operation is performed.

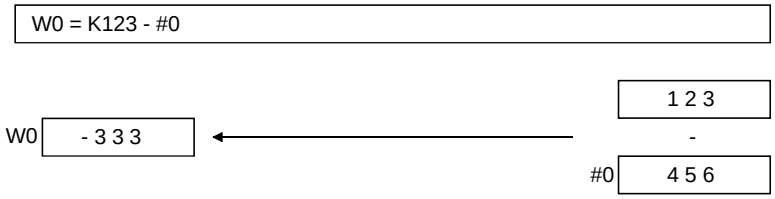
(4) Errors

(a) An operation error will occur if:

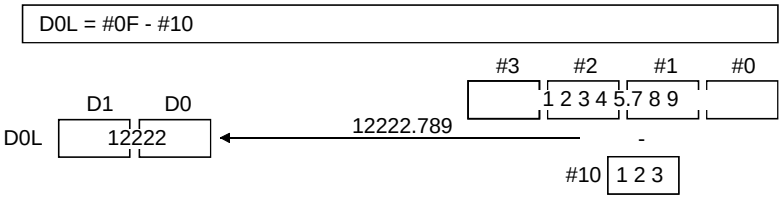
1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which assigns the result of subtracting #0 from K123 to W0



(b) Program which assigns the result of subtracting #10 from #0F to D0L



The 64-bit floating-point type data are used for subtraction, and the result is converted into the 32-bit integer type and then assigned.

5.4.4 Multiplication : *

F/FS	G
○	○

Format	Number of Basic Steps
(S1)*(S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	○	○	○	○	○	○	—	—
(S2)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Multiplicand data	Data type of (S1) or (S2)
(S2)	Multiplier data	which is greater

(3) Functions

(a) The data specified at (S1) is multiplied by the data specified at (S2).

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before operation is performed.

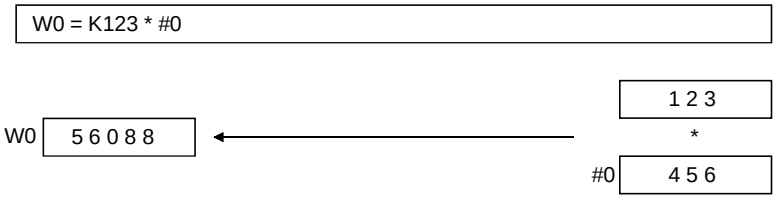
(4) Errors

(a) An operation error will occur if:

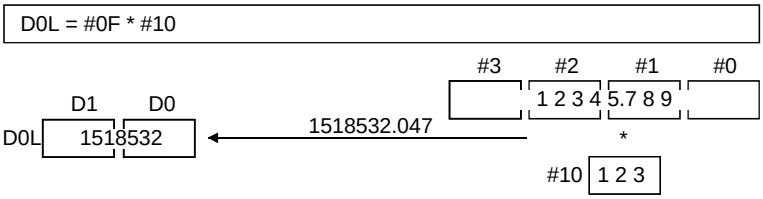
1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which assigns the result of multiplying K123 by #0 to W0



(b) Program which assigns the result of multiplying #0F by #10 to D0L



The 64-bit floating-point type data are used for multiplication, and the result is converted into the 32-bit integer type and then assigned.

5.4.5 Division : /

F/FS	G
○	○

Format	Number of Basic Steps
(S1)/(S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	○	○	○	○	○	○	—	—
(S2)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Dividend data	Data type of (S1) or (S2)
(S2)	Divisor data	which is greater

(3) Functions

(a) The data specified at (S1) is divided by the data specified at (S2) to find a quotient.

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before operation is performed.

(4) Errors

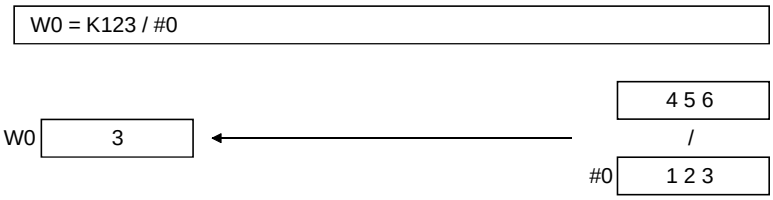
(a) An operation error will occur if:

1) (S2) is 0; or

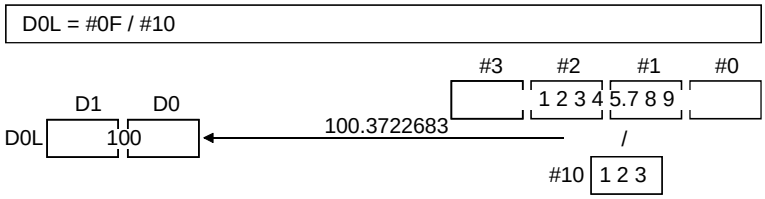
2) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which divides K123 by #0 and assigns a quotient to W0



(b) Program which divides #0F by #10 and assigns a quotient to D0L



The 64-bit floating-point type data are used for division, and the quotient is converted into the 32-bit integer type and then assigned.

5.4.6 Remainder : %

F/FS	G
○	○

Format	Number of Basic Steps
(S1)%(S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	—	○	○	○	—	○	—	—
(S2)	—	○	○	—	○	○	○	—	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Dividend data	Data type (integer type) of (S1) or (S2) which is greater (Integer type)
(S2)	Divisor data	

(3) Functions

(a) The data specified at (S1) is divided by the data specified at (S2) to find a remainder.

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before operation is performed.

(4) Errors

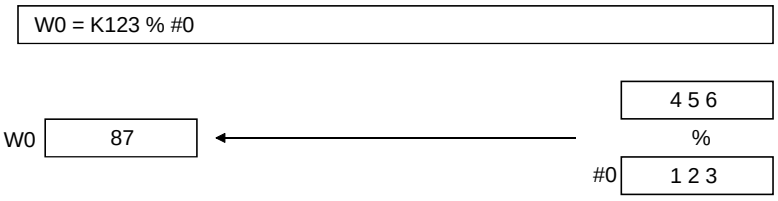
(a) An operation error will occur if:

1) (S2) is 0; or

2) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which divides K123 by #0 and assigns a remainder to W0



5.5 Bit Operations

5.5.1 Bit inversion (complement) : ~

F/FS	G
○	○

Format	Number of Basic Steps
~ (S)	2

(1) Usable data

Data to be Set	Usable Data										
		Word device				Constant					
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(S)	—	○	○	—	○	○	○	—	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data whose bits will be inverted	Data type of (S) (Integer type)

(3) Functions

(a) The bit inverted value of the data specified at (S) is found.

(4) Errors

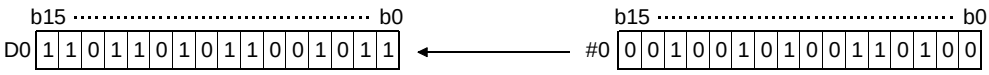
(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which finds the bit inverted value of #0 and assigns the value to D0

```
D0 = ~ #0
```



5.5.2 Bit logical AND : &

F/FS	G
○	○

Format	Number of Basic Steps
(S1)&(S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	—	○	○	○	—	○	—	—
(S2)	—	○	○	—	○	○	○	—	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Data which will be ANDed bit-by-bit	Data type of (S1) or (S2) which is greater (Integer type)
(S2)		

(3) Functions

(a) The bit-by-bit logical product of the data specified at (S1) and the data specified at (S2) is found.

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before operation is performed. At this time, note that signed data is converted.

(4) Errors

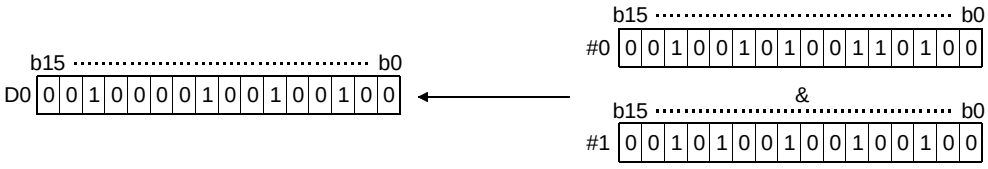
(a) An operation error will occur if:

1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which ANDs #0 and #1 and assigns the result to D0

```
D0 = #0 & #1
```



5.5.3 Bit logical OR : |

F/FS	G
○	○

Format	Number of Basic Steps
(S1) | (S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	—	○	○	○	—	○	—	—
(S2)	—	○	○	—	○	○	○	—	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Data which will be ORed bit-by-bit	Data type of (S1) or (S2) which is greater (Integer type)
(S2)		

(3) Functions

(a) The bit-by-bit logical add of the data specified at (S1) and the data specified at (S2) is found.

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before operation is performed. At this time, note that signed data is converted.

(4) Errors

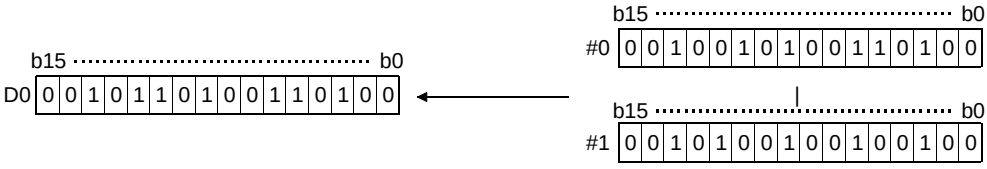
(a) An operation error will occur if:

1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which ORs #0 and #1 and assigns the result to D0

```
D0 = #0 | #1
```



5.5.4 Bit exclusive logical OR : ^

F/FS	G
○	○

Format	Number of Basic Steps
(S1)^(S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	—	○	○	○	—	○	—	—
(S2)	—	○	○	—	○	○	○	—	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Data which will be EXCLUSIVE ORed bit-by-bit	Data type of (S1) or (S2) which is greater (Integer type)
(S2)		

(3) Functions

(a) The bit-by-bit exclusive logical add of the data specified at (S1) and the data specified at (S2) is found.

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before operation is performed. At this time, note that signed data is converted.

(4) Errors

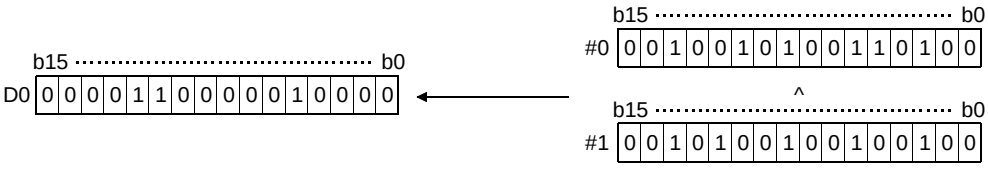
(a) An operation error will occur if:

1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which EXCLUSIVE ORs #0 and #1 and assigns the result to D0

```
D0 = #0 ^ #1
```



5.5.5 Bit right shift : >>

F/FS	G
○	○

Format	Number of Basic Steps
(S1) >> (S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	—	○	○	○	—	○	—	—
(S2)	—	○	○	—	○	○	○	—	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Data to be right-shifted	Data type of (S1)
(S2)	Number of right shifts	(Integer type)

(3) Functions

(a) The data specified at (S1) is shifted to the right by the number of times specified at (S2).

(b) If the most significant bit of (S1) is 1, 1 enters the most significant bit of the right shift result.
If the most significant bit of (S1) is 0, 0 enters the most significant bit of the right shift result.

(c) When (S1) is a 16-bit integer type and (S2) is a negative number or not less than 16, the result is 0.

(d) When (S1) is a 32-bit integer type and (S2) is a negative number or not less than 32, the result is 0.

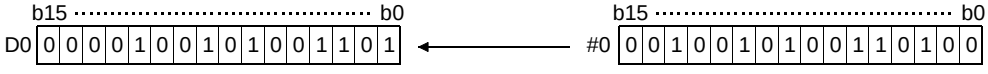
(4) Errors

(a) An operation error will occur if:
1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which shifts #0 two bit positions to the right and assigns the result to D0

```
D0 = #0 >> K2
```



5.5.6 Bit left shift : <<

F/FS	G
○	○

Format	Number of Basic Steps
(S1) << (S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	—	○	○	○	—	○	—	—
(S2)	—	○	○	—	○	○	○	—	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Data to be left-shifted	Data type of (S1)
(S2)	Number of left shifts	(Integer type)

(3) Functions

(a) The data specified at (S1) is shifted to the left by the number of times specified at (S2).

(b) 0 enters the least significant bit of the left shift result.

(c) When (S1) is a 16-bit integer type and (S2) is a negative number or not less than 16, the result is 0.

(d) When (S1) is a 32-bit integer type and (S2) is a negative number or not less than 32, the result is 0.

(4) Errors

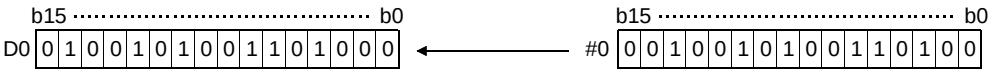
(a) An operation error will occur if:

1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which shifts #0 one bit position to the left and assigns the result to D0

```
D0 = #0 << K1
```



5.5.7 Sign inversion (complement of 2) : -

F/FS	G
○	○

Format	Number of Basic Steps
- (S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data whose sign will be inverted	Data type of (S)

(3) Functions

(a) The sign-inverted value of the data specified at (S) is found.

(4) Errors

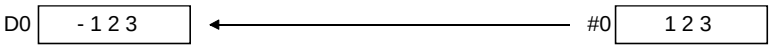
(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which assigns the sign-inverted value of #0 to D0

```
D0 = - #0
```



5.6 Standard Functions

5.6.1 Sine : SIN

F/FS	G
○	○

Format	Number of Basic Steps
SIN(S)	2

(1) Usable data

Data to be Set	Usable Data										
		Word device				Constant					
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Angle data on which SIN (sine) operation will be performed	Floating-point type

(3) Functions

(a) SIN (sine) operation is performed on the data specified at (S).

(b) The data specified at (S) is in an angle (degree) unit.

(c) If (S) is an integer type, it is converted into a floating-point type before operation is performed.

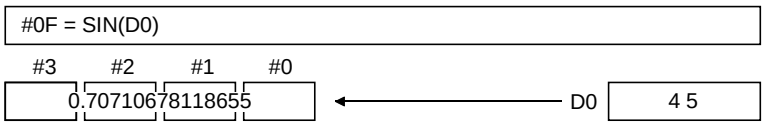
(4) Errors

(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which performs the SIN operation of D0 and assigns the result to #0F



5.6.2 Cosine : COS

F/FS	G
○	○

Format	Number of Basic Steps
COS(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Angle data on which COS (cosine) operation will be performed	Floating-point type

(3) Functions

(a) COS (cosine) operation is performed on the data specified at (S).

(b) The data specified at (S) is in an angle (degree) unit.

(c) If (S) is an integer type, it is converted into a floating-point type before operation is performed.

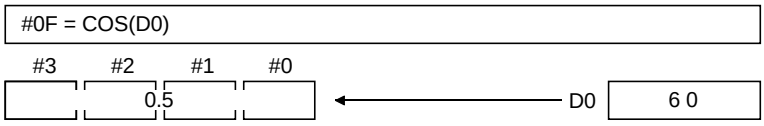
(4) Errors

(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which performs the COS operation of D0 and assigns the result to #0F



5.6.3 Tangent : TAN

F/FS	G
○	○

Format	Number of Basic Steps
TAN(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Angle data on which TAN (tangent) operation will be performed	Floating-point type

(3) Functions

(a) TAN (tangent) operation is performed on the data specified at (S).

(b) The data specified at (S) is in an angle (degree) unit.

(c) If (S) is an integer type, it is converted into a floating-point type before operation is performed.

(4) Errors

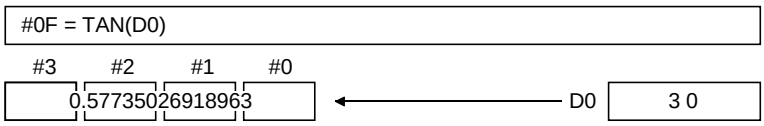
(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range; or

2) (S) is 90+(180∗n). (n is an integer)

(5) Program examples

(a) Program which performs the TAN operation of D0 and assigns the result to #0F



5.6.4 Arcsine : ASIN

F/FS	G
○	○

Format	Number of Basic Steps
ASIN(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	SIN value data on which SIN^{-1} (arcsine) operation will be performed	Floating-point type

(3) Functions

(a) SIN^{-1} (arcsine) operation is performed on the SIN value data specified at (S) to find an angle.

(b) The SIN value specified at (S) must be within the range -1.0 to 1.0.

(c) The operation result is in an angle (degree) unit.

(d) If (S) is an integer type, it is converted into a floating-point type before operation is performed.

(4) Errors

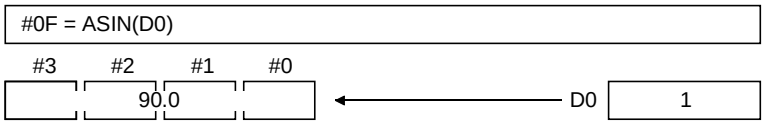
(a) An operation error will occur if:

1) (S) is outside the range -1.0 to 1.0; or

2) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which performs the SIN^{-1} (arcsine) operation of D0 and assigns the result to #0F



5.6.5 Arccosine : ACOS

F/FS	G
○	○

Format	Number of Basic Steps
ACOS(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	COS value data on which COS^{-1} (arccosine) operation will be performed	Floating-point type

(3) Functions

(a) COS^{-1} (arccosine) operation is performed on the COS value data specified at (S) to find an angle.

(b) The COS value specified at (S) must be within the range -1.0 to 1.0.

(c) The operation result is in an angle (degree) unit.

(d) If (S) is an integer type, it is converted into a floating-point type before operation is performed.

(4) Errors

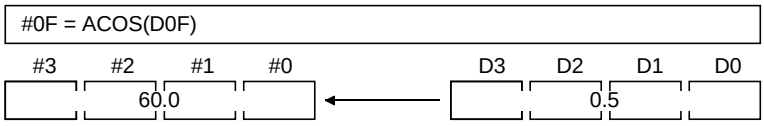
(a) An operation error will occur if:

1) (S) is outside the range -1.0 to 1.0; or

2) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which performs the COS^{-1} (arccosine) operation of D0F and assigns the result to #0F



5.6.6 Arctangent : ATAN

F/FS	G
○	○

Format	Number of Basic Steps
ATAN(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	TAN value data on which TAN^{-1} (arctangent) operation will be performed	Floating-point type

(3) Functions

(a) TAN^{-1} (arccosine) operation is performed on the TAN value data specified at (S) to find an angle.

(b) The operation result is in an angle (degree) unit.

(c) If (S) is an integer type, it is converted into a floating-point type before operation is performed.

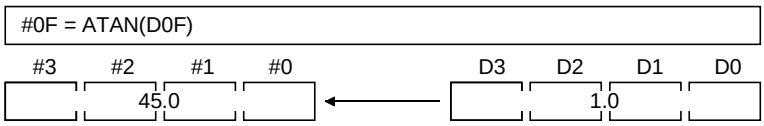
(4) Errors

(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which performs the TAN^{-1} (arctangent) operation of D0F and assigns the result to #0F



5.6.7 Square root : SQRT

F/FS	G
○	○

Format	Number of Basic Steps
SQRT(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data on which square root operation will be performed	Floating-point type

(3) Functions

(a) The square root of the data specified at (S) is found.

(b) Only a positive number may be specified at (S). (Operation cannot be performed with a negative number.)

(c) If (S) is an integer type, it is converted into a floating-point type before operation is performed.

(4) Errors

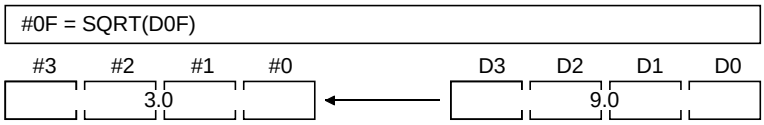
(a) An operation error will occur if:

1) (S) is a negative number; or

2) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which finds the square root of D0F and assigns the result to #0F



5.6.8 Natural logarithm : LN

F/FS	G
○	○

Format	Number of Basic Steps
LN(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data on which natural logarithm operation will be performed	Floating-point type

(3) Functions

(a) The base e natural logarithm of the data specified at (S) is found.

(b) Only a positive number may be specified at (S). (Operation cannot be performed with a negative number.)

(c) If (S) is an integer type, it is converted into a floating-point type before operation is performed.

(4) Errors

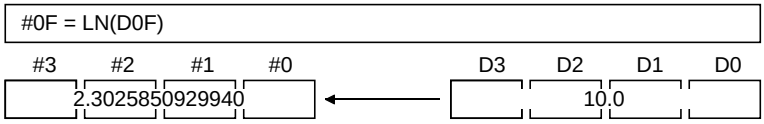
(a) An operation error will occur if:

1) (S) is 0 or a negative number; or

2) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which finds the natural logarithm of D0F and assigns the result to #0F



5.6.9 Exponential operation : EXP

F/FS	G
○	○

Format	Number of Basic Steps
EXP(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data on which exponential operation will be performed	Floating-point type

(3) Functions

(a) Exponential operation is performed on the base e data specified at (S).

(b) If (S) is an integer type, it is converted into a floating-point type before operation is performed.

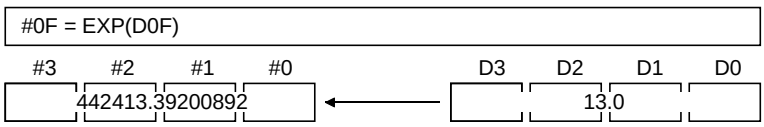
(4) Errors

(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which performs exponential operation of D0F and assigns the result to #0F



5.6.10 Absolute value : ABS

F/FS	G
○	○

Format	Number of Basic Steps
ABS(S)	2

(1) Usable data

Data to be Set	Usable Data											
		Word device				Constant						
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data on which absolute value conversion will be performed	Data type of (S)

(3) Functions

(a) The absolute value of the data specified at (S) is found.

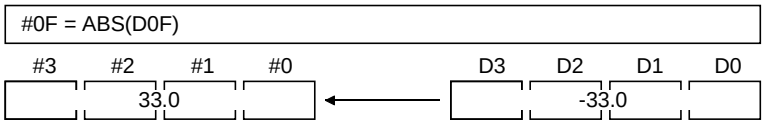
(4) Errors

(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which finds the absolute value of D0F and assigns the result to #0F



5.6.11 Round-off : RND

F/FS	G
○	○

Format	Number of Basic Steps
RND(S)	2

(1) Usable data

Data to be Set	Usable Data											
		Word device				Constant						
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data whose fractional portion will be rounded off	Data type of (S)

(3) Functions

(a) The rounded-off fractional portion value of the data specified at (S) is found.

(b) If (S) is a negative number, the absolute value of (S) is found and its fractional portion is rounded off and signed.

(c) If (S) is an integer type, its value is returned unchanged, with no conversion processing performed.

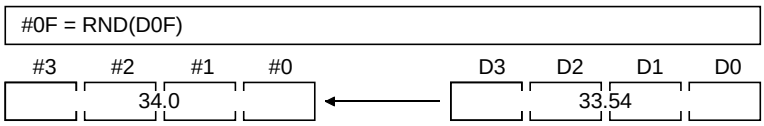
(4) Errors

(a) An operation error will occur if:

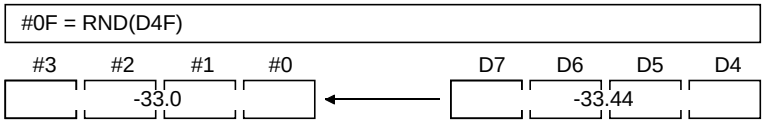
1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which finds the rounded-off fractional portion value of D0F and assigns the result to #0F



(b) Program which finds the rounded-off fractional portion value of D4F and assigns the result to #0F (when D4F is a negative number)



5.6.12 Round-down : FIX

F/FS	G
○	○

Format	Number of Basic Steps
FIX(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data whose fractional portion will be rounded down	Data type of (S)

(3) Functions

(a) The largest integer not greater than the data specified at (S) is found.

(b) If the (S) value is positive, the absolute value will be smaller, and if it is negative, the absolute value will be greater.

(c) If (S) is an integer type, its value is returned unchanged, with no conversion processing performed.

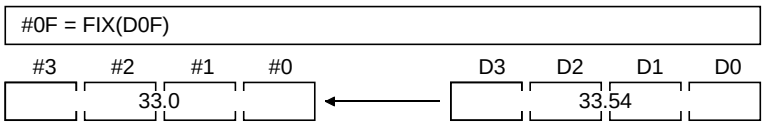
(4) Errors

(a) An operation error will occur if:

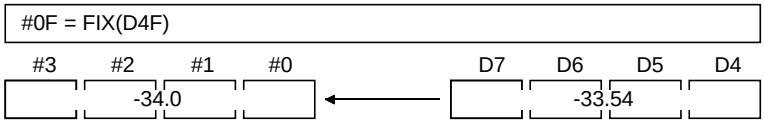
1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which finds the rounded-down fractional portion value of D0F and assigns the result to #0F



(b) Program which finds the rounded-down fractional portion value of D4F and assigns the result to #0F (when D4F is a negative number)



5.6.13 Round-up : FUP

F/FS	G
○	○

Format	Number of Basic Steps
FUP(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data whose fractional portion will be rounded up	Data type of (S)

(3) Functions

(a) The smallest integer not less than the data specified at (S) is found.

(b) If the (S) value is positive, the absolute value will be greater, and if it is negative, the absolute value will be smaller.

(c) If (S) is an integer type, its value is returned unchanged, with no conversion processing performed.

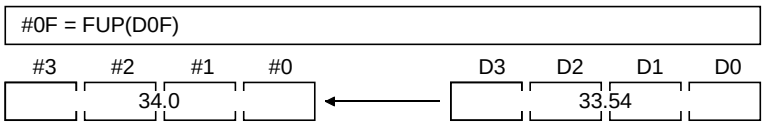
(4) Errors

(a) An operation error will occur if:

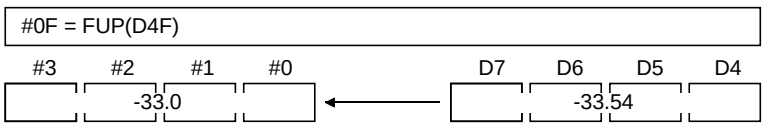
1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which finds the rounded-up fractional portion value of D0F and assigns the result to #0F



(b) Program which finds the rounded-up fractional portion value of D4F and assigns the result to #0F (when D4F is a negative number)



5.6.14 BCD→BIN conversion : BIN

F/FS	G
○	○

Format	Number of Basic Steps
BIN(S)	2

(1) Usable data

Data to be Set	Usable Data										
		Word device				Constant					
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(S)	—	○	○	—	○	○	○	—	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	BCD data which will be converted into BIN data	Data type of (S) (Integer type)

(3) Functions

(a) The BCD data specified at (S) is converted into BIN data.

(b) If (S) is a 16-bit integer type, the data range is 0 to 9999.

(c) If (S) is a 32-bit integer type, the data range is 0 to 99999999.

(4) Errors

(a) An operation error will occur if:

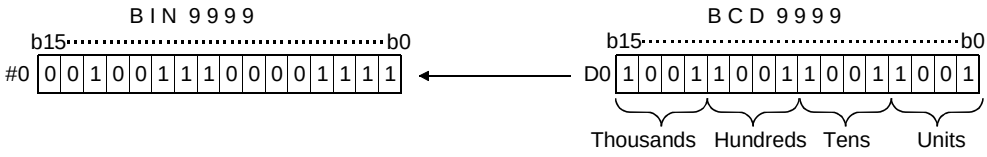
1) A value other than 0 to 9 is in any digit of (S); or

2) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which converts the BCD data of D0 into BIN data and assigns the result to #0

```
#0 = BIN(D0)
```



5.6.15 BIN→BCD conversion : BCD

F/FS	G
○	○

Format	Number of Basic Steps
BCD(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	—	○	○	○	—	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	BIN data which will be converted into BCD data	Data type of (S) (Integer type)

(3) Functions

(a) The BIN data specified at (S) is converted into BCD data.

(b) If (S) is a 16-bit integer type, the data range is 0 to 9999.

(c) If (S) is a 32-bit integer type, the data range is 0 to 99999999.

(4) Errors

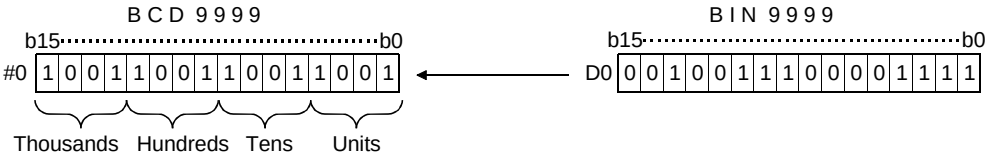
(a) An operation error will occur if:

1) The data is other than 0 to 9999 when (S) is a 16-bit integer type;
2) The data is other than 0 to 99999999 when (S) is a 32-bit integer type; or
3) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which converts the BIN data of D0 into BCD data and assigns the result to #0

```
#0 = BCD(D0)
```



5.7 Type Conversions

5.7.1 Signed 16-bit integral value conversion : SHORT

F/FS	G
○	○

Format	Number of Basic Steps
SHORT(S)	2

(1) Usable data

Data to be Set	Usable Data										
		Word device				Constant					
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data which will be converted into signed 16-bit integral value	16-bit integer type

(3) Functions

(a) The data specified at (S) is converted into a signed 16-bit integral value.

(b) The data range of (S) is -32768 to 32767.

(c) When (S) is a 64-bit floating-point type, its fractional portion is rounded down before conversion is made.

(d) If (S) is a 16-bit integer type, its value is returned unchanged, with no conversion processing performed.

(4) Errors

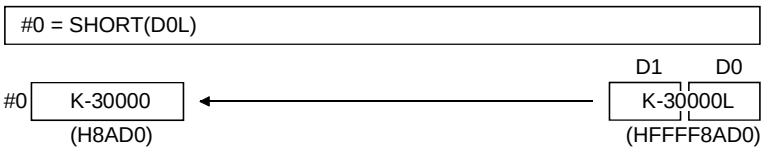
(a) An operation error will occur if:

1) The (S) data is outside the range -32768 to 32767; or

2) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which converts the data of D0L into a signed 16-bit integral value and assigns the result to #0



5.7.2 Unsigned 16-bit integral value conversion : USHORT

F/FS	G
○	○

Format	Number of Basic Steps
USHORT(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data which will be converted into unsigned 16-bit integral value	16-bit integer type

(3) Functions

(a) The data specified at (S) is converted into an unsigned 16-bit integral value.

(b) The data range of (S) is 0 to 65535.

(c) When (S) is a 64-bit floating-point type, its fractional portion is rounded down before conversion is made.

(d) If (S) is a 16-bit integer type, its value is returned unchanged, with no conversion processing performed.

(4) Errors

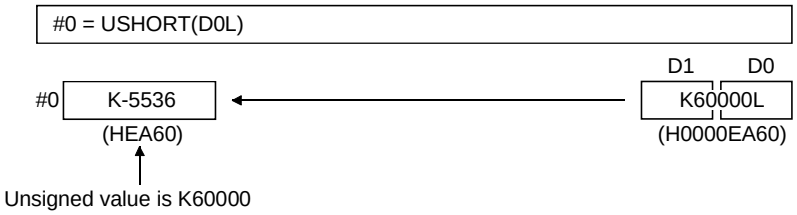
(a) An operation error will occur if:

1) The (S) data is outside the range 0 to 65535; or

2) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which converts the data of D0L into an unsigned 16-bit integral value and assigns the result to #0



5.7.3 Signed 32-bit integral value conversion : LONG

F/FS	G
○	○

Format	Number of Basic Steps
LONG(S)	2

(1) Usable data

Data to be Set	Usable Data											
		Word device				Constant						
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data which will be converted into signed 32-bit integral value	32-bit integer type

(3) Functions

(a) The data specified at (S) is converted into a signed 32-bit integral value.

(b) The data range of (S) is -2147483648 to 2147483647.

(c) When (S) is a 64-bit floating-point type, its fractional portion is rounded down before conversion is made.

(d) If (S) is a 32-bit integer type, its value is returned unchanged, with no conversion processing performed.

(4) Errors

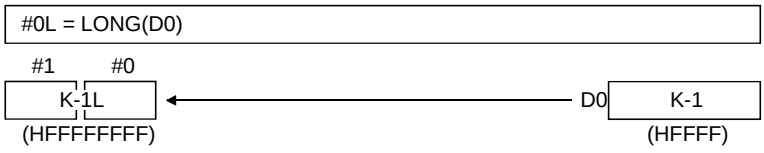
(a) An operation error will occur if:

1) The (S) data is outside the range -2147483648 to 2147483647; or

2) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which converts the data of D0 into a signed 32-bit integral value and assigns the result to #0L



5.7.4 Unsigned 32-bit integral value conversion : ULONG

F/FS	G
○	○

Format	Number of Basic Steps
ULONG(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data which will be converted into unsigned 32-bit integral value	32-bit integer type

(3) Functions

(a) The data specified at (S) is converted into an unsigned 32-bit integral value.

(b) The data range of (S) is 0 to 4294967295.

(c) When (S) is a 64-bit floating-point type, its fractional portion is rounded down before conversion is made.

(d) If (S) is a 32-bit integer type, its value is returned unchanged, with no conversion processing performed.

(4) Errors

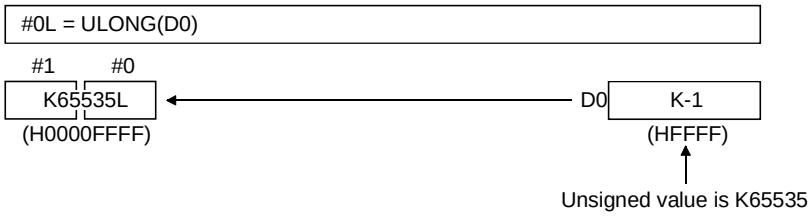
(a) An operation error will occur if:

1) The (S) data is outside the range 0 to 4294967295; or

2) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which converts the data of D0 into an unsigned 32-bit integral value and assigns the result to #0L



5.7.5 Signed 64-bit floating-point value conversion : FLOAT

F/FS	G
○	○

Format	Number of Basic Steps
FLOAT(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data which will be converted into signed 64-bit floating-point value	64-bit floating-point type

(3) Functions

(a) The data specified at (S) is converted into a signed 64-bit floating-point value.

(b) If (S) is a 64-bit floating-point type, its value is returned unchanged, with no conversion processing performed.

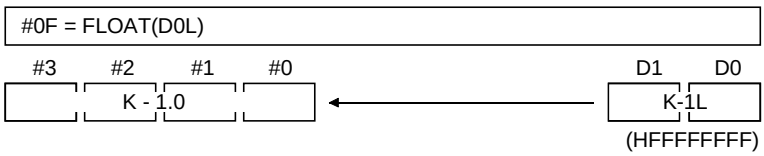
(4) Errors

(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which converts the data of D0L into a signed 64-bit floating-point value and assigns the result to #0F



5.7.6 Unsigned 64-bit floating-point value conversion : UFLOAT

F/FS	G
○	○

Format	Number of Basic Steps
UFLOAT(S)	2

(1) Usable data

Data to be Set	Usable Data										
		Word device				Constant					
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(S)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data which will be converted into unsigned 64-bit floating-point value	64-bit floating-point type

(3) Functions

(a) The data specified at (S) is converted into an unsigned 64-bit floating-point value.

(b) If (S) is a 64-bit floating-point type, its value is returned unchanged, with no conversion processing performed.

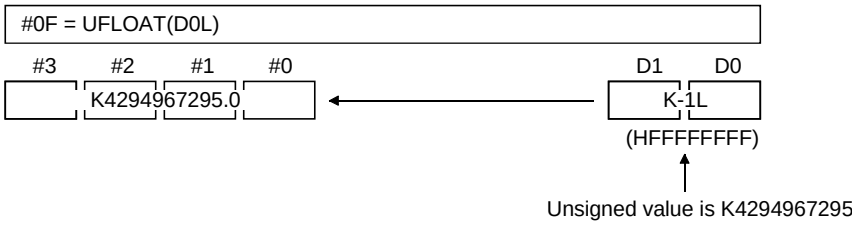
(4) Errors

(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which converts the data of D0L into an unsigned 64-bit floating-point value and assigns the result to #0F



5.8 Bit Device Statuses

5.8.1 ON (normally open contact) : (None)

F/FS	G
○	○

Format	Number of Basic Steps
(S)	2

(1) Usable data

Data to be Set	Usable Data											
		Word device				Constant						
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(S)	○	—	—	—	—	—	—	—	—	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Bit device used in bit conditional expression	Logical type (true/false)

(3) Functions

(a) True is returned when the bit device specified at (S) in a bit conditional expression is ON (1), or false is returned when that bit device is OFF (0).

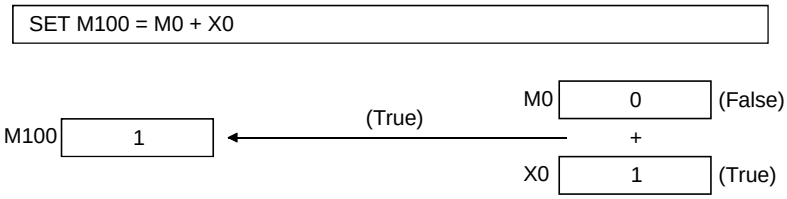
(4) Errors

(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which sets M100 when either of M0 and X0 is ON (1)



5.8.2 OFF (normally closed contact) : !

F/FS	G
○	○

Format	Number of Basic Steps
!(S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	○	—	—	—	—	—	—	—	—	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Bit device used in bit conditional expression	Logical type (true/false)

(3) Functions

(a) True is returned when the bit device specified at (S) in a bit conditional expression is OFF (0), or false is returned when that bit device is ON (1).

(4) Errors

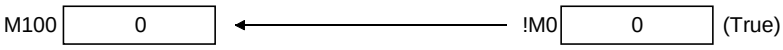
(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which resets M100 when M0 is OFF (0)

```
RST M100 = !M0
```



5.9 Bit Device Controls

5.9.1 Device set : SET=

F/FS	G
○	○

Format	Number of Basic Steps
SET(D)=(S)	4

(1) Usable data

Data to be Set	Usable Data										
		Word device				Constant					
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(D)	○	—	—	—	—	—	—	—	—	—	—
(S)	○	—	—	—	—	—	—	—	—	○	○

(Note) • PX, TT and CT are write-disabled and cannot be used at (D).
• At (D), you cannot use M2001 to M2032 with the A273UH-S3 or M2001 to M2008 with the A172SH.

(2) Data to be set

Data to be Set	Description	Data Type of Result
(D)	Bit data for device set	Bit logical type (true/false)
(S)	Condition data which determines whether device set will be performed or not	

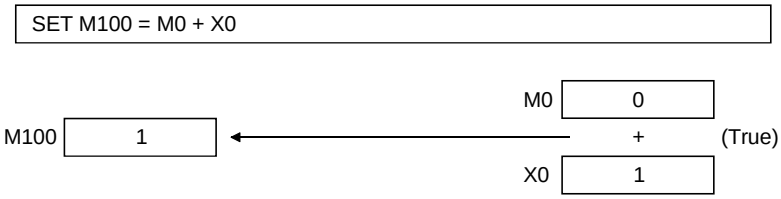
(3) Functions
(a) If the data specified at (S) is true, the bit data specified at (D) is set.

(b) (S) can be omitted.
At this time, the format is "SET(D)" and device set is made unconditionally.

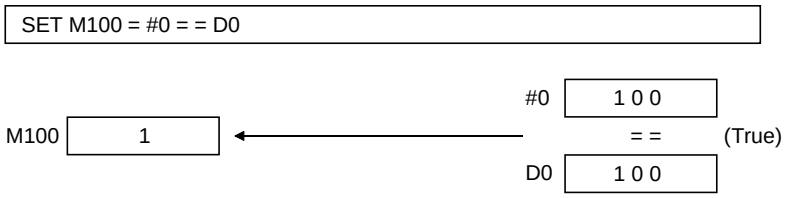
(c) When this instruction is set as a transition condition in the last block of a transient program, whether the data specified at (S) is true or false is returned as logical type data.
In this case, (S) cannot be omitted.

(4) Errors
(a) An operation error will occur if:
1) (D) or (S) is an indirectly specified device and its device number is outside the range.

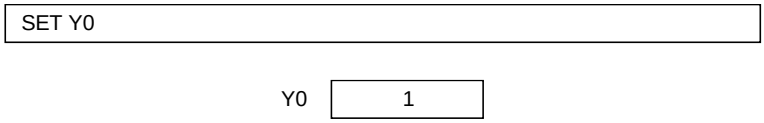
(5) Program examples
(a) Program which sets M100 when either of M0 and X0 is 1



(b) Program which sets M100 when #0 is equal to D0



(c) Program which sets Y0 unconditionally



5.9.2 Device reset : RST=

F/FS	G
○	○

Format	Number of Basic Steps
RST(D)=(S)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(D)	○	—	—	—	—	—	—	—	—	—	—
(S)	○	—	—	—	—	—	—	—	—	○	○

(Note) • PX, TT and CT are write-disabled and cannot be used at (D).
• At (D), you cannot use M2001 to M2032 with the A273UH-S3 or M2001 to M2008 with the A172SH.

(2) Data to be set

Data to be Set	Description	Data Type of Result
(D)	Bit data for device reset	Bit logical type (true/false)
(S)	Condition data which determines whether device reset will be performed or not	

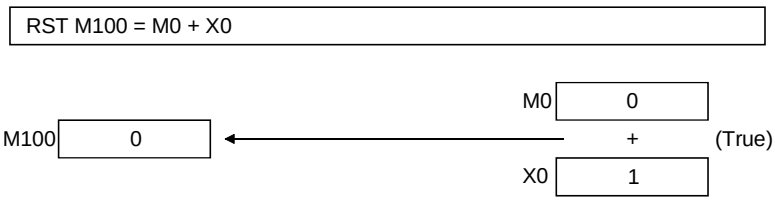
(3) Functions
(a) If the data specified at (S) is true, the bit data specified at (D) is reset.

(b) (S) can be omitted.
At this time, the format is "RST(D)" and device reset is made unconditionally.

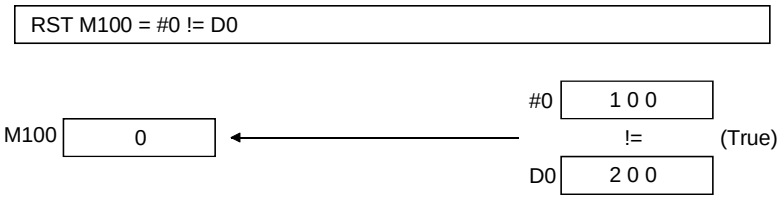
(c) When this instruction is set as a transition condition in the last block of a transient program, whether the data specified at (S) is true or false is returned as logical type data.
In this case, (S) cannot be omitted.

(4) Errors
(a) An operation error will occur if:
1) (D) or (S) is an indirectly specified device and its device number is outside the range.

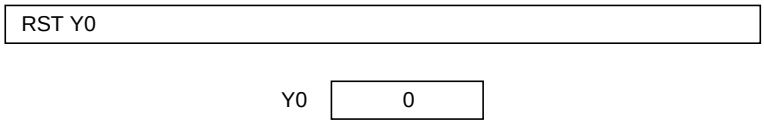
(5) Program examples
(a) Program which resets M100 when either of M0 and X0 is 1



(b) Program which resets M100 when #0 is equal to D0



(c) Program which resets Y0 unconditionally



5.9.3 Device output : DOUT

F/FS	G
○	○

Format	Number of Basic Steps
DOUT(D), (S)	4

(1) Usable data

Data to be Set	Usable Data											
		Word device				Constant						
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(D)	○	—	—	—	—	—	—	—	—	—	—
(S)	—	○	○	—	○	○	○	—	○	—	—

(Note) • PX, TT, CT and special M cannot be used at (D).
• At (D), you cannot use the range including M2000 to M2127 with the A273UH-S3 or the range including M2000 to M2047 with the A172SH.

(2) Data to be set

Data to be Set	Description	Data Type of Result
(D)	Output destination bit data	Batch bit
(S)	Output source data	

(3) Functions

(a) The data specified at (S) is output to the bit data specified at (D).

(b) Specify a multiple of 16 as the device number of the bit data specified at (D).

(c) If the type of (S) is a 16-bit integer type, 16 points of the (S) data, starting at the least significant bit, are output in order to the bit devices headed by the one specified at (D).

(d) If the type of (S) is a 32-bit integer type, 32 points of the (S) data, starting at the least significant bit, are output in order to the bit devices headed by the one specified at (D).

(4) Errors

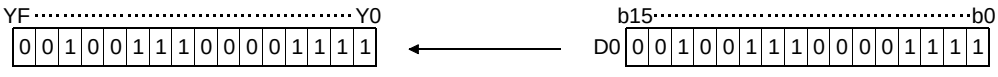
(a) An operation error will occur if:

1) (D) or (S) is an indirectly specified device and its device number is outside the range.
2) (D) is an indirectly specified device and its device number is not a multiple of 16.

(5) Program examples

(a) Program which outputs the data of D0 to Y0-YF

```
DOUT Y0, D0
```



5.9.4 Device input : DIN

F/FS	G
○	○

Format	Number of Basic Steps
DIN(D), (S)	4

(1) Usable data

Data to be Set	Usable Data										
		Word device				Constant					
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(D)	—	○	○	—	—	—	—	—	—	—	—
(S)	○	—	—	—	—	—	—	—	—	—	—

(Note) : T and C are write-disabled and cannot be used at (D).

(2) Data to be set

Data to be Set	Description	Data Type of Result
(D)	Input destination data	Data type of (D) (Integer type)
(S)	Input source bit data	

(3) Functions

(a) The bit data specified at (S) is input to the data specified at (D).

(b) Specify a multiple of 16 as the device number of the bit data specified at (S).

(c) If the type of (D) is a 16-bit integer type, 16 points of the (D) data, starting at the least significant bit, are input in order to the bit devices headed by the one specified at (S).

(d) If the type of (D) is a 32-bit integer type, 32 points of the (D) data, starting at the least significant bit, are input in order to the bit devices headed by the one specified at (S).

(4) Errors

(a) An operation error will occur if:

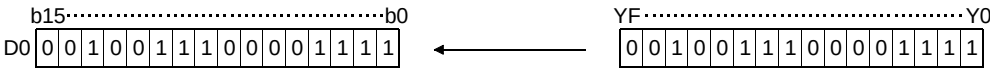
1) (D) or (S) is an indirectly specified device and its device number is outside the range.

2) (S) is an indirectly specified device and its device number is not a multiple of 16.

(5) Program examples

(a) Program which inputs the data of X0-XF to D0

```
DIN D0, X0
```



5.10 Logical Operations

5.10.1 Logical acknowledgement : (None)

F/FS	G
○	○

Format	Number of Basic Steps
(S)	—

(1) Usable data

Data to be Set	Usable Data										
		Word device				Constant					
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(S)	○	—	—	—	—	—	—	—	—	○	○

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data which will be logically acknowledged	Logical type (true/false)

(3) Functions

(a) Whether the logical type data specified at (S) is true or false is returned unchanged. (Logical acknowledgement)

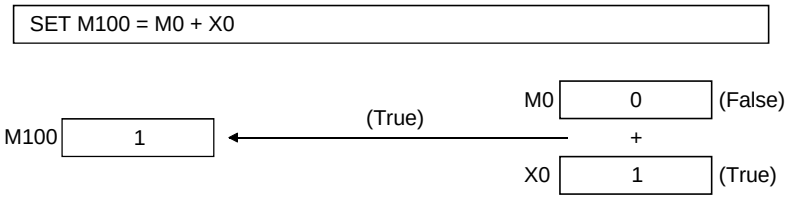
(4) Errors

(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which sets M100 when either of M0 and X0 is ON (1)



5.10.2 Logical negation : !

F/FS	G
○	○

Format	Number of Basic Steps
! (S)	2

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	○	—	—	—	—	—	—	—	—	○	○

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Data which will be logically negated	Logical type (true/false)

(3) Functions

(a) The data specified at (S) is logically negated.

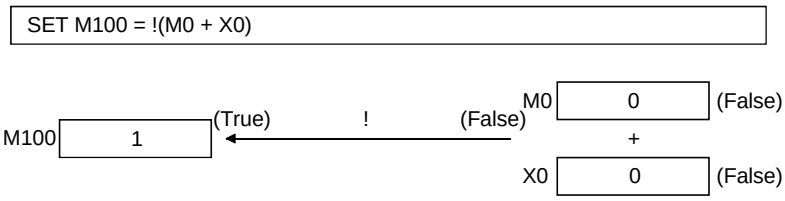
(4) Errors

(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which sets M100 when "either of M0 and X0 is not ON (1)" (i.e. when M0 and X0 are both OFF (0))



5.10.3 Logical AND : *

F/FS	G
○	○

Format	Number of Basic Steps
(S1)*(S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	○	—	—	—	—	—	—	—	—	○	○
(S2)	○	—	—	—	—	—	—	—	—	○	○

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Data which will be ANDed	Logical type (true/false)
(S2)		

(3) Functions

(a) The data specified at (S1) and the data specified at (S2) are ANDed.

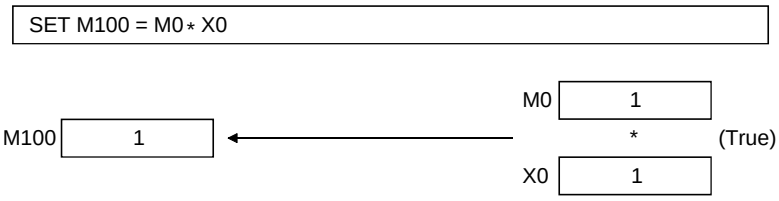
(4) Errors

(a) An operation error will occur if:

1) (S) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which sets M100 when M0 and X0 are both 1



5.10.4 Logical OR : +

F/FS	G
○	○

Format	Number of Basic Steps
(S1)+(S2)	4

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	○	—	—	—	—	—	—	—	—	○	○
(S2)	○	—	—	—	—	—	—	—	—	○	○

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Data which will be ORed	Logical type (true/false)
(S2)		

(3) Functions

(a) The data specified at (S1) and the data specified at (S2) are ORed.

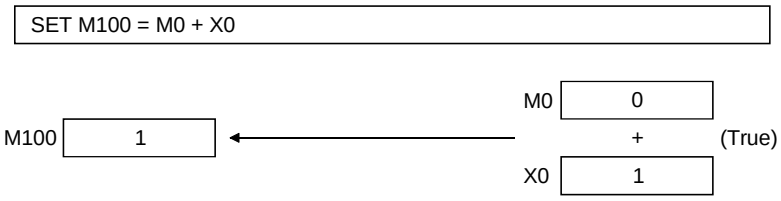
(4) Errors

(a) An operation error will occur if:

1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which sets M100 when either of M0 and X0 is 1



5.11 Comparison Operations

5.11.1 Equal to : ==

F/FS	G
○	○

Format	Number of Basic Steps
(S1)==(S2)	4

(1) Usable data

Data to be Set	Usable Data										
		Word device				Constant					
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(S1)	—	○	○	○	○	○	○	○	○	—	—
(S2)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Data which will be compared	Logical type (true/false)
(S2)		

(3) Functions

(a) The data specified at (S1) and the data specified at (S2) are compared, and the result is true if they are equal.

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before comparison is performed.

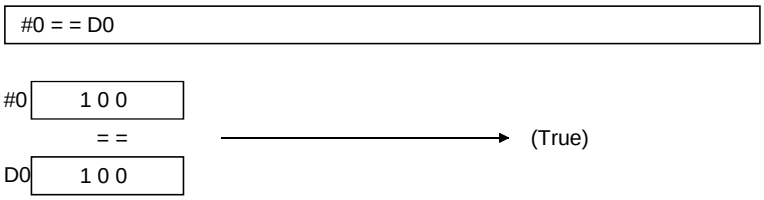
(4) Errors

(a) An operation error will occur if:

1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which compares whether #0 and D0 are equal or not



5.11.2 Not equal to : !=

F/FS	G
○	○

Format	Number of Basic Steps
(S1)!=(S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	○	○	○	○	○	○	—	—
(S2)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Data which will be compared	Logical type (true/false)
(S2)		

(3) Functions

(a) The data specified at (S1) and the data specified at (S2) are compared, and the result is true if they are not equal.

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before comparison is performed.

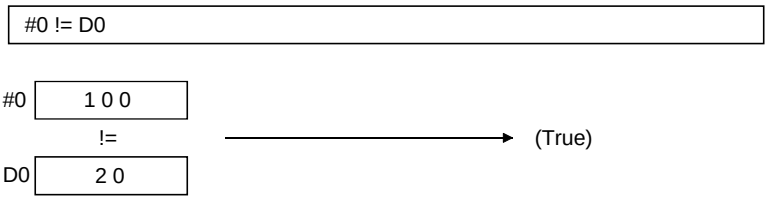
(4) Errors

(a) An operation error will occur if:

1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which compares whether #0 and D0 are unequal or not



5.11.3 Less than : <

F/FS	G
○	○

Format	Number of Basic Steps
(S1)<(S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	○	○	○	○	○	○	—	—
(S2)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Data which will be compared	Logical type (true/false)
(S2)		

(3) Functions

(a) The result is true if the data specified at (S1) is less than the data specified at (S2).

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before comparison is performed.

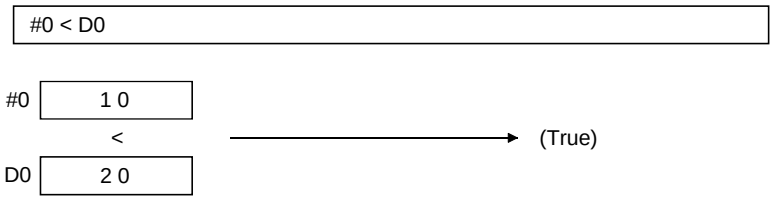
(4) Errors

(a) An operation error will occur if:

1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which compares whether #0 is less than D0 or not



5.11.4 Less than or equal to: <=

F/FS	G
○	○

Format	Number of Basic Steps
(S1)<=(S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	○	○	○	○	○	○	—	—
(S2)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Data which will be compared	Logical type (true/false)
(S2)		

(3) Functions

(a) The result is true if the data specified at (S1) is less than or equal to the data specified at (S2).

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before comparison is performed.

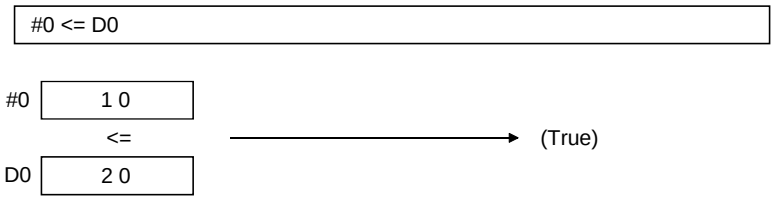
(4) Errors

(a) An operation error will occur if:

1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which compares whether #0 is less than or equal to D0 or not



5.11.5 More than : >

F/FS	G
○	○

Format	Number of Basic Steps
(S1)>(S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	○	○	○	○	○	○	—	—
(S2)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Data which will be compared	Logical type (true/false)
(S2)		

(3) Functions

(a) The result is true if the data specified at (S1) is greater than the data specified at (S2).

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before comparison is performed.

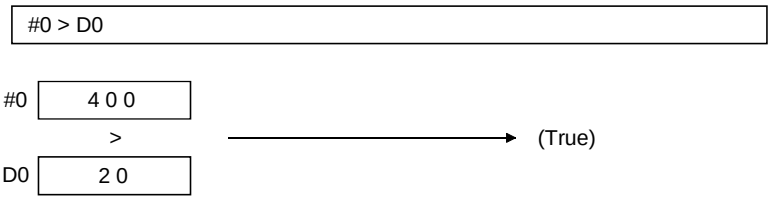
(4) Errors

(a) An operation error will occur if:

1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which compares whether #0 is greater than D0 or not



5.11.6 More than or equal to: >=

F/FS	G
○	○

Format	Number of Basic Steps
(S1)>=(S2)	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	○	○	○	○	○	○	○	○	—	—
(S2)	—	○	○	○	○	○	○	○	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Data which will be compared	Logical type (true/false)
(S2)		

(3) Functions

(a) The result is true if the data specified at (S1) is greater than or equal to the data specified at (S2).

(b) When (S1) and (S2) differ in data type, the data of the smaller data type is converted into that of the greater type before comparison is performed.

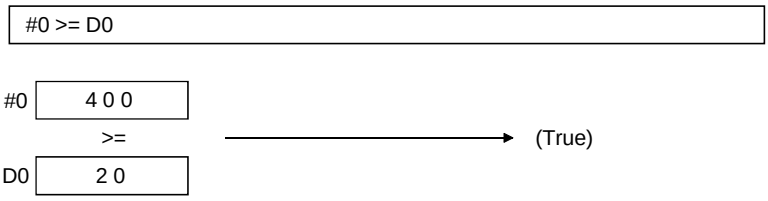
(4) Errors

(a) An operation error will occur if:

1) (S1) or (S2) is an indirectly specified device and its device number is outside the range.

(5) Program examples

(a) Program which compares whether #0 is greater than or equal to D0 or not



5.12 Motion-Dedicated Functions (CHGV, CHGT)

5.12.1 Speed change request : CHGV

F/FS	G
○	○

Format	Number of Basic Steps
CHGV((S1), (S2))	4

(1) Usable data

Data to be Set	Usable Data										
		Word device				Constant					
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
(S1)	—	—	—	—	—	○	—	—	—	—	—
(S2)	—	○	○	—	—	○	○	—	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Axis number to which speed change request will be given	—
(S2)	Specified speed	

(3) Functions

(a) A speed change is made in the following procedure.

1) The speed changing flag (M2021 to M2028/M2061 to M2092) corresponding to the axis specified at (S1) is turned ON.
2) The speed of the axis specified at (S1) is changed to the speed specified at (S2).
3) The speed changing flag is turned OFF.

(b) The axis number that may be set at (S1) is within the following range.

A172SHCPUN	A173UHCPU(-S1)/ A273UHCPU-S3(32-axis feature)
1 to 8	1 to 32

For interpolation control, set any one of the interpolation axes.
When linear interpolation control is exercised, a speed change varies as described below with the positioning speed designation method set in the servo program.

Positioning Speed Designation Method	Operation
Combined speed designation	A speed change is made so that the combined speed becomes the speed specified at (S2).
Longest axis designation	A speed change is made so that the longest axis speed becomes the speed specified at (S2).
Reference axis speed designation	A speed change is made so that the reference axis speed becomes the speed specified at (S2).

(c) Operation varies with the sign of the specified speed set at (S2).

Sign of Specified Speed	Operation
Positive	Speed change
0	Temporary stop
Negative	Return

(d) The specified speed that may be set at (S2) is within the following range.
1) Real mode

	mm		inch		degree		PULSE	
	Setting range	Unit	Setting range	Unit	Setting range	Unit	Setting range	Unit
Speed change request	0 to 600000000	$\times10^{-2}$ mm/min	0 to 600000000	$\times10^{-3}$ inch/min	0 to 2147483647	$\times10^{-3}$ degree/min	0 to 10000000	PLS/s
Return request	-1 to -600000000	$\times10^{-2}$ mm/min	-1 to -600000000	$\times10^{-3}$ inch/min	-1 to -2147483647	$\times10^{-3}$ degree/min	-1 to -10000000	PLS/s

2) Virtual mode

	PULSE	
	Setting range	Unit
Speed change request	0 to 10000000	PLS/s
Return request	-1 to -10000000	PLS/s

(e) Specifying a negative speed and making a speed change request during starting allows the axis to start deceleration at that point and return in the opposite direction upon completion of deceleration.
Operation varies with the servo instruction as described below.

Control Mode	Servo Instruction	Operation
Linear control	ABS-1 INC-1 ABS-2 INC-2 ABS-3 INC-3 ABS-4 INC-4	On completion of deceleration, the axis reverses its moving direction, returns to the positioning starting point at the absolute value of the specified speed, and stops (waits) there. For circular interpolation, the axis returns in the circular path.
Circular interpolation control	ABS circular INC circular	
Fixed-pitch feed	FEED-1 FEED-2 FEED-3	
Constant-speed control	CPSTART1 CPSTART2 CPSTART3 CPSTART4	On completion of deceleration, the axis reverses its moving direction, returns to the preceding point at the absolute value of the specified speed, and stops (waits) there.
Speed control (I)	VF VR	On completion of deceleration, the axis reverses its moving direction at the absolute value of the specified speed. The axis does not stop until a stop command is input.
Speed control (II)	VVF VVR	
Speed/position control	VPF VPR VPSTART	The axis cannot return. The speed change request is regarded as a normal speed change request. Minor error 305 will occur and the axis will be controlled at the speed limit value.
Position follow-up control	PFSTART	
Speed change control	VSTART	
JOG operation		
High-speed oscillation	OSC	A speed change cannot be made. Minor error 310 will occur.
Zeroing	ZERO	A speed change cannot be made. Minor error 301 will occur.

Reference) Minor error 301: A speed change was made during zeroing.
Minor error 305: The preset speed is outside the range 0 to speed limit value.
Minor error 310: A speed change was made during high-speed oscillation.

[Controls]

(a) If a speed change is made to a negative speed, control varies with the control mode during starting as indicated in the table in Section 5.12.1(3)(e).

(b) The returning command speed is the absolute value of a new speed.

(c) When the axis is waiting at the return position

1) Signal states

• Start acceptance (M200n)	ON (unchanged from before execution of CHGV execution)
• Positioning start completion (M16m0)	ON (unchanged from before execution of CHGV execution)
• Positioning completion (M16m1)	OFF
• In-position (M16m2)	ON
• Command in-position (M16m3)	OFF
• Speed change "0" accepting flag (-)	ON

2) Make a speed change to a positive speed for a restart.
3) Turn ON the stop command to terminate positioning.
4) A negative speed change made again will be ignored.

(d) While the axis is returning in the speed control mode

1) Make a speed change to a positive speed to change the moving direction again.
2) Turn ON the stop command to make a stop.
3) A speed change is made in the opposite direction if a negative speed change is made again.

(4) Errors

(a) An operation error will occur and a speed change will not be made if:

1) The specified axis number at (S1) is outside the range; or
2) (S2) is an indirectly specified device and its device number is outside the range.

(b) A minor error will occur and a speed change will not be made if:

1) The axis specified at (S1) is zeroing; or
2) The axis specified at (S1) is decelerating (minor error 303).

(c) A minor error will occur and the axis to be controlled at the speed limit value if:

1) The absolute value of the speed specified at (S2) is greater than the speed limit value.
(Minor error 305)

POINT
If, during constant-speed control, the absolute value of a negative new speed is higher than the speed specified in the servo program, return control is exercised at the speed specified in the program (speed clamp control for a speed change during constant-speed control). At this time, an error will not occur.

(5) Program examples

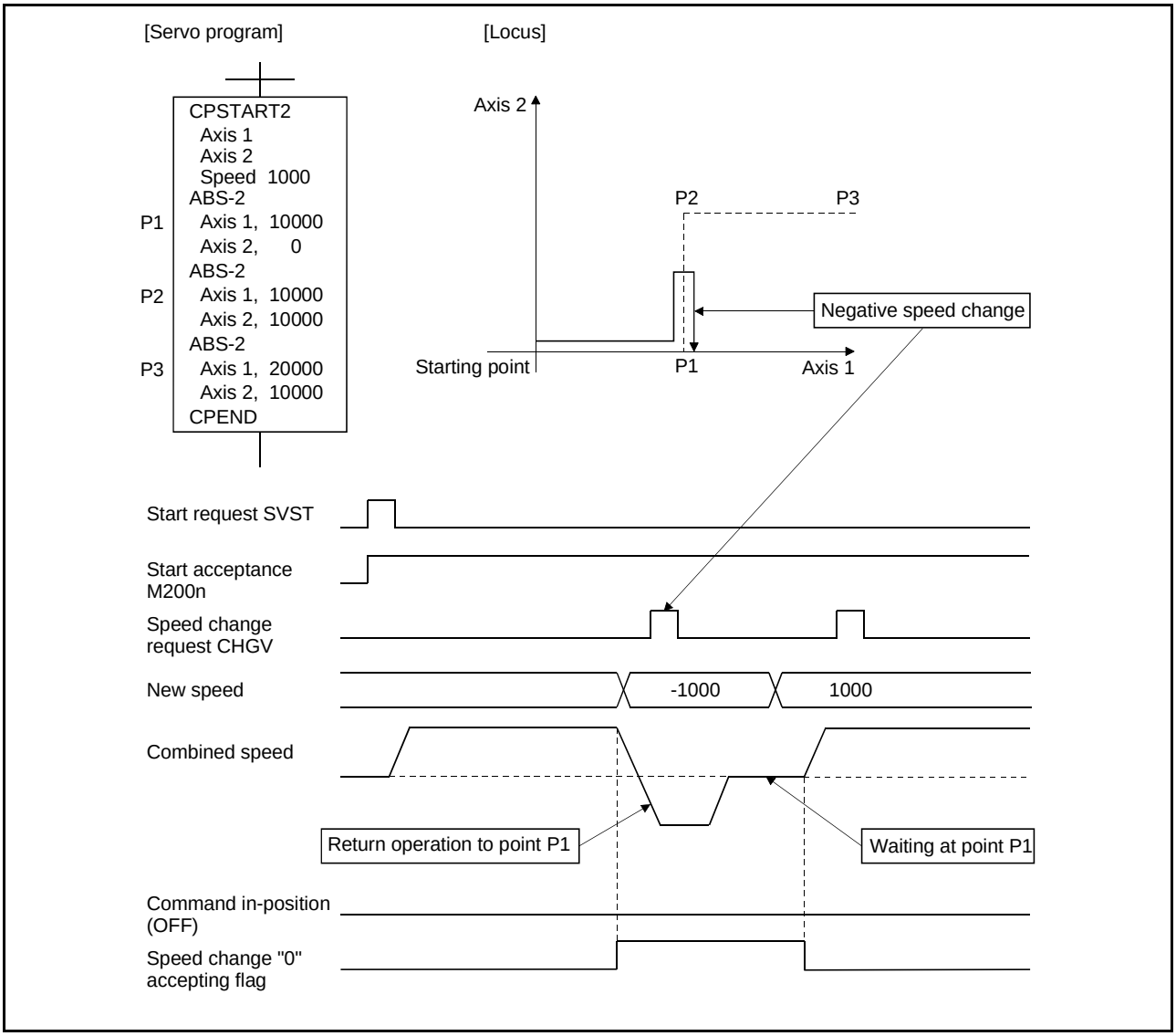
(a) Program which changes the positioning speed of axis 2

```
CHGV(K2,K10)
```

(b) Return program which changes the positioning speed of axis 1 to a negative value

```
CHGV(K1,K-1000)
```

The following operation will be performed when a return request is made in constant-speed control.



If a speed change to a negative speed is made during execution of positioning to P2 as shown above, the axis returns to P1 along the program-specified locus and waits at P1.

POINT

Speed changing instructions

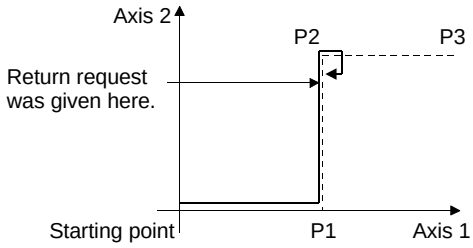
(1) A speed change may be invalid if it is made from when a servo program start request is made until the "positioning start completion signal" status changes to ON. When making a speed change at almost the same timing as a start, always create a program which will execute the speed change after the "positioning start completion signal" has turned ON.

(2) A return request, which is made while the axis is at a stop waiting for FIN using the M code FIN waiting function during constant-speed control, will be ignored.

(3) In the above example, if a return request is given right before P2 and the axis passes through P2 during deceleration, the axis will return to P2.

(4) The speed change "0" acceptance flag is not available for the positioning-dedicated devices on the A172SHCPUN.

(5) There will be a delay of time equivalent to an operation cycle at the maximum in the response time from when the CHGV instruction is executed until the speed begins to change actually.



5.12.2 Torque limit value change request : CHGT

F/FS	G
○	○

Format	Number of Basic Steps
CHGT((S1), (S2))	4

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S1)	—	—	—	—	—	○	—	—	—	—	—
(S2)	—	○	○	—	—	○	○	—	○	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S1)	Axis number to which torque limit value change request will be given	—
(S2)	Specified torque limit value	

(3) Functions

(a) The torque limit value of the axis specified at (S1) is changed to the torque limit value axis specified at (S2).

(b) In the real mode, any axis that has completed a servo startup can be changed in torque limit value any time, independently of the status, starting, stopping, servo ON or servo OFF.

(c) The axis number that may be set at (S1) is within the following range.

A172SHCPUN	A173UHCPU(-S1)/ A273UHCPU-S3(32-axes feature)
1 to 8	1 to 32

(d) The torque limit value that may be set at (S2) is within the range 1 to 500[%].

(e) The torque limit value specified here and the one specified in the servo program have the following relationships.

At start

At a normal start, the torque limit value is given to the servo of the start axis according to "P. torque" set in the servo program or the "torque limit value" of the specified parameter block.
For an interpolation start, the torque limit value is given to the number of axes to be interpolated.

↓

Executing the CHGT instruction gives the preset torque limit value to only the specified axis.

↓

Thereafter, the torque limit value given to the servo at a servo program start or JOG start is made valid only when it is lower than the torque limit value specified in CHGT.
This torque limit value clamp processing is performed per axis.

During start

1) If the following torque limit value has been set, it will not be changed to higher than the torque limit value specified in the CHGT instruction.
   - Torque limit value at a midway point in constant-speed control or speed change control
   - Torque limit value at the point of switching to position control in speed/ position changing control
   - Torque limit value in speed control II
2) The CHGT instruction accepts a torque limit value which is higher than the torque limit value set in the servo program or parameter block.

(4) Errors

(a) An operation error will occur and a torque limit value change will not be made if:
   1) The specified axis number at (S1) is outside the range; or
   2) (S2) is an indirectly specified device and its device number is outside the range.

(b) A minor error will occur and a torque limit value change will not be made if:
   1) The torque limit value specified at (S2) is outside the range 1 to 500[%] (minor error 311); or
   2) The CHGT instruction is executed for any axis that has not yet been started (minor error 312).

(5) Program examples

(a) Program which changes the torque limit value of axis 2

```
CHGT(K2,K10)
```

POINT

(1) In the virtual mode, the CHGT instruction is invalid (ignored). When changing the torque limit value during operation in the virtual mode, set the "torque limit value setting device" in the output module parameter of the machine mechanism program.

(2) There will be a delay of time equivalent to an operation cycle at the maximum in the response time from when the CHGT instruction is executed until the torque limit value is changed actually.

5.13 Other Instructions

5.13.1 Event task enable : EI

F/FS	G
○	○

Format	Number of Basic Steps
EI	1

(1) Usable data

Data to be Set	Usable Data											
		Word device				Constant						
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
—	—	—	—	—	—	—	—	—	—	—	—	—

(2) Data to be set

There are no data to be set.

(3) Functions

(a) The execution of an event task is enabled.

(b) This instruction is usable with a normal task only.

(4) Errors

(a) An operation error will occur if:

1) This instruction is used with other than a normal task.

(5) Program examples

(a) Enables the execution of an event task.

```
EI
```

5.13.2 Event task disable : DI

F/FS	G
○	○

Format	Number of Basic Steps
DI	1

(1) Usable data

Data to be Set	Usable Data											
		Word device				Constant						
	Bit device	16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)	Calculation expression	Bit conditional expression	Compar-ison conditional expression
—	—	—	—	—	—	—	—	—	—	—	—	—

(2) Data to be set
There are no data to be set.

(3) Functions
(a) The execution of an event task is disabled.

(b) If an external interrupt or PLC interrupt occurs after execution of the DI instruction, the corresponding event task is executed once at the execution of the EI instruction. (If two or more external interrupts or PLC interrupts occur during DI, the corresponding event task is executed only once at the execution of the EI instruction.)

(c) During DI, a fixed-cycle event task is not executed.

(d) The execution of an NMI task cannot be disabled.

(e) The DI status is established at power-on or when a reset is made with the RUN/STOP switch.

(4) Errors
(a) An operation error will occur if:
1) This instruction is used with other than a normal task.

(5) Program examples
(a) Program which disables the execution of an event task.

```
DI
```

5.13.3 No operation : NOP

F/FS	G
○	○

Format	Number of Basic Steps
NOP	1

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
—	—	—	—	—	—	—	—	—	—	—	—

(2) Data to be set
There are no data to be set.

(3) Functions
(a) This is a no-operation instruction and does not affect the preceding operations.

(4) Errors
(a) There are no operation errors for no operation: NOP.

5.13.4 Block transfer : BMOV

F/FS	G
○	○

Format	Number of Basic Steps
BMOV(D), (S), (n)	7

(1) Usable data

Data to be Set	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(D)	—	○	—	—	—	—	○	—	—	—	—
(S)	—	○	—	—	—	—	○	—	—	—	—
(n)	—	○	—	—	—	○	—	—	—	—	—

(Note) : When a 32-bit integer type constant is specified at (D) or (S), the number specified is the absolute address of the PCPU memory. Specify the absolute address with an even number.

(2) Data to be set

Data to be Set	Description	Data Type of Result
(D)	Transfer destination device starting number	—
(S)	Transfer source device starting number	
(n)	Number of words to be transferred	

(3) Functions

(a) n words of data in word devices, starting with the one specified at (S), are batch-transferred to n words of word devices, starting with the one specified at (D).

(b) Data can be transferred if the word devices of the transfer source and destination overlap.
Data are transferred from the devices, starting with the one at (S), for transfer of data from devices of larger numbers to those of smaller numbers, or starting with the one at (S)+(n-1) for transfer of data from devices of smaller numbers to those of larger numbers.

(c) Specifying Nn (cam number) at (D) or (S) enables batch-transfer of cam data.
In the motion controller, the cam data of the same cam number must already have been registered.
The number of transferred words specified at (n) should match the resolution of the specified cam number.
Operations performed at write and read of cam data are as described below.

At cam data write

- For A172SHCPUN
  The cam data storage area is rewritten.
- For A173UHCPU(-S1)/A273UHCPU-S3
  The cam data is stored into the extended file register (block 10 and later) but this instruction rewrites the cam data import area in the PCPU. The extended file register contents are not changed.
  Therefore, turning the cam data/limit switch output data batch-change request flag (M2056) OFF, then ON, or powering on the controller and resetting it with the key returns the cam data import area to the extended file register contents.

- Transfer of data to the cam data area is also executed during cam operation.
  Be careful not to perform write while operation is being performed with the same cam number.

At cam data read

- The cam data in the currently set status are read.

(d) The word devices that may be set at (D), (S) and (n) are as indicated below.

Data to Be Set	Word Devices			Cam Number Designation
	Dn	Wn	#n	Nn
(D)	○	○	○	○
(S)	○	○	○	○
(n)	○	○	○	—

(Note) • Nn indicates the cam number.
• You cannot use T, C and special D.
• You cannot specify the device numbers indirectly.

(e) The cam number that may be set as Nn is within the following range.

A172SHCPUN	A173UHCPU(-S1) /A273UHCPU-S3
1 to 64	1 to 64
	101 to 164
	201 to 264
	301 to 364

(4) Errors

(a) An operation error will occur if:

1) The cam data of the cam number specified at (D) or (S) are not yet registered to the motion controller;
2) The resolution of the cam number specified at (D) or (S) differs from the number of transferred words specified at (n);
3) The PCPU memory address specified at (D) or (S) is outside the SRAM range;
4) (S) to (S)+(n-1) is outside the device range;
5) (D) to (D)+(n-1) is outside the device range; or
6) (n) is 0 or a negative number.

} when (n) specified is a word device

(b) When conversion is made in program editing of the peripheral software, an error will occur if:

1) (S) to (S)+(n-1) is outside the device range;
2) (D) to (D)+(n-1) is outside the device range;
3) (n) is 0 or a negative number; or

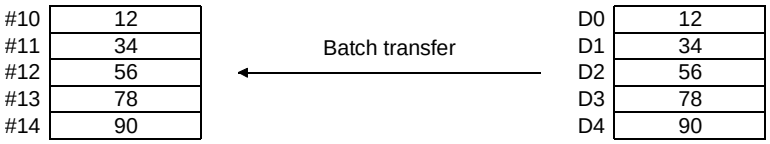
} when (n) specified is a constant

4) The 32-bit integer type constant (PCPU memory absolute address specified) specified at (D) or (S) is an odd number.

(5) Program examples

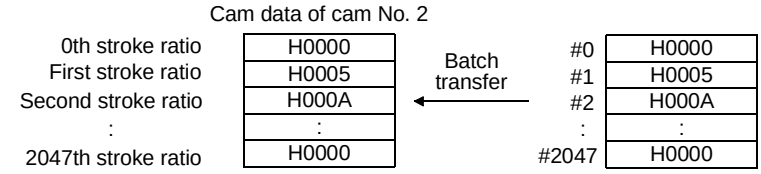
(a) Program which batch-transfers 5 words of data in devices, starting with D0, to 5 words of devices, starting with #10

```
BMOV #10,D0,K5
```



(b) Program which batch-transfers 2048 words of data in devices, starting with #0, to the data area of cam No. 2 (resolution 2048)

```
BMOV N2,#0,K2048
```



5.13.5 Time to wait : TIME

F/FS	G
—	○

Format	Number of Basic Steps
TIME(S)	7

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
(S)	—	○	○	—	—	○	○	—	—	—	—

(2) Data to be set

Data to be Set	Description	Data Type of Result
(S)	Waiting time (0 to 2147483647)ms	Logical type (true/false)

(3) Functions

(a) A wait state continues for the time specified at (S).
The result is false when the elapsed time is less than the preset time, or the result is true and execution transits when the preset time has elapsed.

(b) When a 16-bit integer type word device is used to specify any of 32768 to 65535ms at (S), convert it into an unsigned 16-bit integral value with USHORT. (Refer to the program example.)

(4) Errors

(a) An operation error will occur if:
1) (S) is an indirectly specified device and its device number is outside the range; or
2) Data (device data when device is indirectly specified) at (S) is outside the range 0 to 2147483647.

(5) Program examples

(a) Program which sets a wait of 60 seconds (when constant is specified)

```
TIME K60000
```

(b) Program for a case where there may be a wait of 32768 to 65535ms for 16-bit integer type indirect designation (#0)

```
TIME USHORT(#0)
```

(c) Program which SETS (RSTs) a bit device when the specified time has elapsed

```
SET M100 = TIME K60000
```

POINT

(1) When the waiting time setting is indirectly specified with a word device, the value imported first is used as the device value for exercising control. The set time cannot be changed if the device value is changed during a wait state.

(2) The TIME instruction is equivalent to a conditional expression, and therefore may be set on only the last line of a transition (G) program.

(3) When the transition program (Gn) of the same number having the TIME instruction setting is used in multiple SFC programs, avoid running them at the same time. (If they are run simultaneously, the waiting time in the program run first will be illegal.)

5.14 Comment Statement : //

F/FS	G
○	○

Format	Number of Basic Steps
//	—

(1) Usable data

Data to be Set	Usable Data											
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Compar-ison conditional expression
		16-bit integer type	32-bit integer type(L)	64-bit floating-point type(F)	Coasting timer	16-bit integer type (K/H)	32-bit integer type (K/H,L)	64-bit floating-point type (K)			
—	—	—	—	—	—	—	—	—	—	—	—

(2) Data to be set
There are no data to be set.

(3) Functions
(a) A character string from after // to a block end is a comment.

(4) Errors
(a) There are no operation errors for comment: //.

(5) Program examples
(a) Example which has commented an assignment program

```
D0=D1//Assigns the D0 value (16-bit integer data) to D1.
```

6. TRANSITION PROGRAMS

6.1 Transition Programs

(1) Transition programs

(a) In transition programs, you can set assignment operation expressions, motion-dedicated functions, bit device control commands and transition conditions.

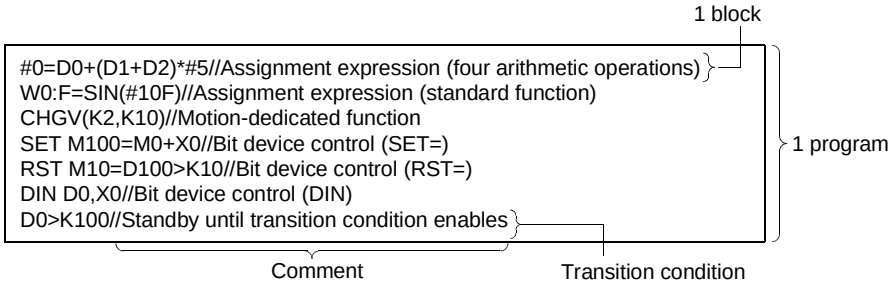
(b) You can set multiple blocks in a single transition program.

(c) There are no restrictions on the number of blocks that may be set in a single transition program.
Note that one program is within 64k bytes.

(d) The maximum number of characters in one block is 128.

(e) You must set a transition condition in the last block of a transition program.
The transition program is repeated until the transition condition enables, and when the transition condition has enabled, it shifts to the next step. The transition condition may be set only in the last block.

(f) As a special transition program, you can create a program where only no operation (NOP) is set in one block.
Use this program when you want to proceed to the next step on completion of a servo program run and there are no special conditions to be set as interlocks. For more information, refer to "4.9 Branches, Couplings".
A transition program example is given below.



What can be set as a transition condition in the last block are bit conditional expressions, comparison conditional expressions and device set (SET=)/device reset (RST=) which return logical data values (true/false). In the case of device set (SET=)/device reset (RST=), whether the bit or comparison conditional expression specified at (S) is true or false is a transition condition, and when the transition condition enables, device set/reset is carried out and execution shifts to the next step.
Transition condition description examples are given below.

Classification	Description Example
Bit conditional expression	M0
	!M0+X10*M100
Comparison conditional expression	(D0>K100)+(D100L!=K20L)
Device set (SET=)	SET Y0=M100
Device reset (RST=)	RST M10=D0==K100

POINT

(1) A transition program differs from an operation control program in that a transition condition is set in the last block.
Other settings are the same as those of the operation control program.

(2) When setting device set (SET=)/device reset (RST=) in the last block as a transition condition, you cannot omit the bit or comparison conditional expression to be specified at (S).

(3) You cannot set only the bit or comparison conditional expression in other than the last block. You can set device set (SET=)/device reset (RST=) in other than the last block.

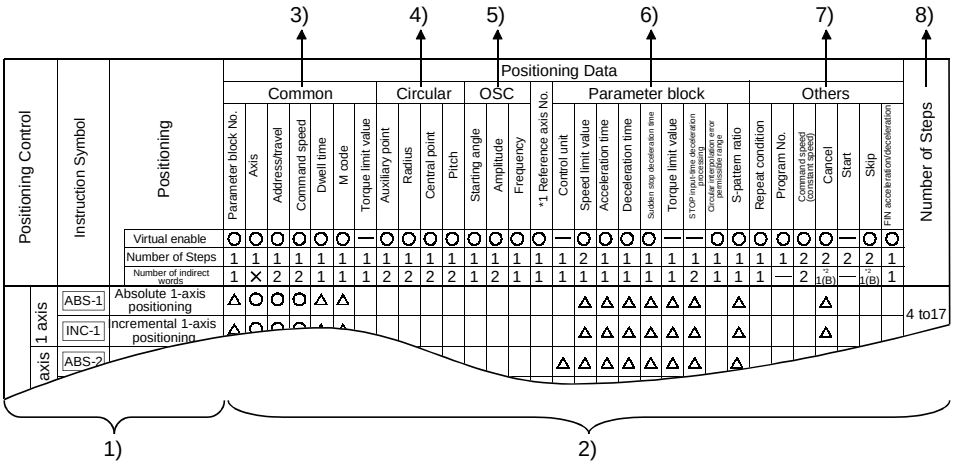
7. MOTION CONTROL PROGRAMS

7.1 Servo Instruction List

Table 7.2 lists servo instructions used in servo programs.
Refer to Sections 7.2 to 7.4 for details of the present value change control (CHGA, CHGA-E, CHGA-C) which are newly available.
For other servo instructions, refer to the "Motion Controller (SV13/SV22) Programming Manual (Real Mode)".

(1) Guide to servo instruction list

Table 7.1 Guide to Servo Instruction List



Number	Description	
1)	Instruction symbol	Gives the servo instructions usable in servo programs.
	Processing	Gives the processing outlines of the servo instructions.
2)	(1) Indicates positioning data which can be set in servo instructions. (a) ○: Item which must be set (if this data is not set, servo instruction cannot be executed) (b) △: Item which is set when required (when this data is not set, default value is used for control)	
	(2) Allows direct or indirect designation (except axis number) (a) Direct designation: Set with numerical value. (b) Indirect designation: Set with word device (D, W). • Servo program run is controlled using the preset word device contents. • Each setting item may either be 1- or 2-word data. • For 2-word data, set the first device number.	
	(3) Number of steps As there are more setting items, there are more number of instruction steps. (The number of steps is displayed when a servo program is created.) (The instruction + ○ item comprise the minimum steps, and one △ item increases the number of steps by 1.)	
3)	Items common to the servo instructions	
4)	Items set in circular interpolation starting servo programs	
5)	Items set for high-speed oscillation	
6)	Set when changing the parameter block (default value when not set) data set in the servo program to exercise control. (The parameter block data are not changed.)	
7)	Setting items other than the common, circular and parameter block items (Items to be set vary with the servo instruction.)	
8)	Indicates the number of steps of each servo instruction.	

(2) Servo instruction list

Table 7.2 indicates the servo instructions available for servo programs and the positioning data set in servo instructions.

Table 7.2 Servo Instruction List

Positioning Control		Instruction Symbol	Processing	Positioning Data: Common: Parameter block No.	Axis	Address/travel	Command speed	Dwell time	M code	Torque limit value	Circular: Auxiliary point	Radius	Central point	Pitch	OSC: Starting angle	Amplitude	Frequency	*1: Reference axis No.	Parameter block: Control unit	Speed limit value	Acceleration time	Deceleration time	Sudden stop deceleration time	Torque limit value	STOP input-time deceleration processing	Circular interpolation error permissible range	S-pattern ratio	Others: Repeat condition	Program No.	Command speed (constant speed)	Cancel	Start	Skip	FIN acceleration/deceleration	WAIT-ON/OFF	Number of Steps
			Virtual enable	○	○	○	○	○	○	—	○	○	○	○	—	—	—	○	—	○	○	○	○	—	—	○	○	○	○	○	○	—	○	○	○	
			Number of steps	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	2	2	1	2	1	2	
			Number of indirect words	1	—	2	2	1	1	1	2	2	2	1	2	2	2	1	1	2	1	1	1	1	1	2	1	1/ 1(B)	—	2	*2 1(B)	—	*2 1(B)	1	B	
Linear control	1 axis	ABS-1	Absolute 1-axis positioning	△	○	○	○	△	△											△	△	△	△	△	△		△				△					4 to 17
		INC-1	Incremental 1-axis positioning	△	○	○	○	△	△											△	△	△	△	△	△		△				△					
	2 axes	ABS-2	Absolute 2-axis linear interpolation	△	○	○	○	△	△									○	△	△	△	△	△	△	△		△				△					5 to 20
		INC-2	Incremental 2-axis linear interpolation	△	○	○	○	△	△									○	△	△	△	△	△	△	△		△				△					
	3 axes	ABS-3	Absolute 3-axis linear interpolation	△	○	○	○	△	△									○	△	△	△	△	△	△	△		△				△					7 to 21
		INC-3	Incremental 3-axis linear interpolation	△	○	○	○	△	△									○	△	△	△	△	△	△	△		△				△					
	4 axes	ABS-4	Absolute 4-axis linear interpolation	△	○	○	○	△	△									○	△	△	△	△	△	△	△		△				△					8 to 22
		INC-4	Incremental 4-axis linear interpolation	△	○	○	○	△	△									○	△	△	△	△	△	△	△		△				△					
Circular interpolation control	Auxiliary point-specified	ABS	Absolute auxiliary point-specified circular interpolation	△	○	○	○	△	△		○								△	△	△	△	△	△	△	△	△				△					7 to 22
		INC	Incremental auxiliary point-specified circular interpolation	△	○	○	○	△	△		○								△	△	△	△	△	△	△	△	△				△					
	Radius-specified	ABS	Absolute radius-specified circular interpolation less than CW 180°	△	○	○	○	△	△			○							△	△	△	△	△	△	△	△	△				△					6 to 21
		ABS	Absolute radius-specified circular interpolation CW 180° or more	△	○	○	○	△	△			○							△	△	△	△	△	△	△	△	△				△					
		ABS	Absolute radius-specified circular interpolation less than CCW 180°	△	○	○	○	△	△			○							△	△	△	△	△	△	△	△	△				△					
		ABS	Absolute radius-specified circular interpolation CCW 180° or more	△	○	○	○	△	△			○							△	△	△	△	△	△	△	△	△				△					
		INC	Incremental radius-specified circular interpolation less than CW 180°	△	○	○	○	△	△			○							△	△	△	△	△	△	△	△	△				△					
		INC	Incremental radius-specified circular interpolation CW 180° or more	△	○	○	○	△	△			○							△	△	△	△	△	△	△	△	△				△					
		INC	Incremental radius-specified circular interpolation less than CCW 180°	△	○	○	○	△	△			○							△	△	△	△	△	△	△	△	△				△					
		INC	Incremental radius-specified circular interpolation CCW 180° or more	△	○	○	○	△	△			○							△	△	△	△	△	△	△	△	△				△					
	Central point-specified	ABS	Absolute central point-specified circular interpolation CW	△	○	○	○	△	△				○							△	△	△	△	△	△	△	△	△				△					7 to 22
		ABS	Absolute central point-specified circular interpolation CCW	△	○	○	○	△	△				○							△	△	△	△	△	△	△	△	△				△					
		INC	Incremental central point-specified circular interpolation CW	△	○	○	○	△	△				○							△	△	△	△	△	△	△	△	△				△					
		INC	Incremental central point-specified circular interpolation CCW	△	○	○	○	△	△				○							△	△	△	△	△	△	△	△	△				△					

○: Item which must be set, △: Item which is set when required

*1 Only when reference axis speed is specified.

*2 (B) indicates a bit device.

Table 7.2 Servo Instruction List (Continued)

Positioning Control		Instruction Symbol	Processing	Positioning Data: Common: Parameter block No.	Common: Axis	Common: Address/travel	Common: Command speed	Common: Dwell time	Common: M code	Common: Torque limit value	Circular: Auxiliary point	Circular: Radius	Circular: Central point	Circular: Pitch	OSC: Starting angle	OSC: Amplitude	OSC: Frequency	*1: Reference axis No.	Parameter block: Control unit	Parameter block: Speed limit value	Parameter block: Acceleration time	Parameter block: Deceleration time	Parameter block: Sudden stop deceleration time	Parameter block: Torque limit value	Parameter block: STOP input-time deceleration processing	Parameter block: Circular interpolation error permissible range	Parameter block: S-pattern ratio	Others: Repeat condition	Others: Program No.	Others: Command speed (constant speed)	Others: Cancel	Others: Start	Others: Skip	Others: FIN acceleration/deceleration	Others: WAIT-ON/OFF	Number of Steps
			Virtual enable	○	○	○	○	○	○	—	○	○	○	○	—	—	—	○	—	○	○	○	○	—	—	○	○	○	○	○	○	—	○	○	○	○
			Number of steps	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	2	2	1	2	1	2	
			Number of indirect words	1	—	2	2	1	1	1	2	2	2	1	2	2	2	1	1	2	1	1	1	1	1	2	1	1/1(B)	—	2	*2 1(B)	—	*2 1(B)	1	B	
Fixed-pitch feed	1 axis	FEED-1	1-axis fixed-pitch feed start	△	○	○	○	△	△											△	△	△	△	△	△		△				△					4 to 17
Fixed-pitch feed	2 axes	FEED-2	2-axis linear interpolation fixed-pitch feed start	△	○	○	○	△	△										△	△	△	△	△	△	△		△				△					5 to 19
Fixed-pitch feed	3 axes	FEED-3	3-axis linear interpolation fixed-pitch feed start	△	○	○	○	△	△										△	△	△	△	△	△	△		△				△					7 to 21
Speed control (I)	Forward rotation	VF	Speed control (I) forward rotation start	△	○		○		△											△	△	△	△	△	△		△				△					3 to 15
Speed control (I)	Reverse rotation	VR	Speed control (I) reverse rotation start	△	○		○		△											△	△	△	△	△	△		△				△					
Speed control (II)	Forward rotation	VVF	Speed control (II) forward rotation start	△	○		○		△	△										△	△	△	△	△	△		△				△					3 to 16
Speed control (II)	Reverse rotation	VVR	Speed control (II) reverse rotation start	△	○		○		△	△										△	△	△	△	△	△		△				△					
Speed-position control	Forward rotation	VPF	Speed-position control forward rotation start	△	○	○	○	△	△	△										△	△	△	△	△	△		△				△					4 to 18
Speed-position control	Reverse rotation	VPR	Speed-position control reverse rotation start	△	○	○	○	△	△	△										△	△	△	△	△	△		△				△					
Speed-position control	Restart	VPSTART	Speed-position control restart		○																										△					2 to 4
Speed switching control		VSTART	Speed switching control start	△															△	△	△	△	△	△	△		△				△					1 to 13
Speed switching control		VEND	Speed switching control end																																	1
Speed switching control		ABS-1	Speed switching control end point address		○	○	○	△	△	△																					△					4 to 9
Speed switching control		ABS-2			○	○	○	△	△	△																					△					5 to 10
Speed switching control		ABS-3			○	○	○	△	△	△																					△					7 to 12
Speed switching control		INC-1	Travel up to speed switching control end point		○	○	○	△	△	△																					△					4 to 9
Speed switching control		INC-2			○	○	○	△	△	△																					△					5 to 10
Speed switching control		INC-3			○	○	○	△	△	△																					△					7 to 12
Speed switching control		VABS	Speed switching point absolute designation			○	○		△	△																											4 to 6
Speed switching control		VINC	Speed switching point incremental designation			○	○		△	△																											
Position follow-up control		PFSTART	Position follow-up control start	△	○	○	○		△											△	△	△	△	△	△		△				△					4 to 16
Constant-speed control		CPSTART1	1-axis constant-speed control start	△	○		○													△	△	△	△	△	△		△				△			△		3 to 15
Constant-speed control		CPSTART2	2-axis constant-speed control start	△	○		○												△	△	△	△	△	△	△	△	△				△			△		3 to 17
Constant-speed control		CPSTART3	3-axis constant-speed control start	△	○		○												△	△	△	△	△	△	△		△				△			△		4 to 17
Constant-speed control		CPSTART4	4-axis constant-speed control start	△	○		○												△	△	△	△	△	△	△		△				△			△		

○: Item which must be set, △: Item which is set when required
*1 Only when reference axis speed is specified.
*2 (B) indicates a bit device.

Table 7.2 Servo Instruction List (Continued)

Positioning Control	Instruction Symbol	Processing	Positioning Data: Common: Parameter block No.	Axis	Address/travel	Command speed	Dwell time	M code	Torque limit value	Circular: Auxiliary point	Radius	Central point	Pitch	OSC: Starting angle	Amplitude	Frequency	*1: Reference axis No.	Parameter block: Control unit	Speed limit value	Acceleration time	Deceleration time	Sudden stop deceleration time	Torque limit value	STOP input-time deceleration processing	Circular interpolation error permissible range	S-pattern ratio	Others: Repeat condition	Program No.	Command speed (constant speed)	Cancel	Start	Skip	FIN acceleration/deceleration	WAIT-ON/OFF	Number of Steps
		Virtual enable	○	○	○	○	○	○	—	○	○	○	○	—	—	—	○	—	○	○	○	○	—	—	○	○	○	○	○	○	—	○	○	○	
		Number of steps	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	1	1	1	1	1	1	1	1	2	2	1	2	1	2		
		Number of indirect words	1	—	2	2	1	1	1	2	2	2	1	2	2	2	1	1	2	1	1	1	1	1	2	1	1/1(B)	—	2	2/1(B)	—	*2 1(B)	1	B	
Constant-speed control	ABS-1	Constant-speed control passing point absolute designation		○	○			△	△																			△			△		△	2 to 10
	ABS-2			○	○			△	△																			△			△		△	3 to 11
	ABS-3			○	○			△	△																			△			△		△	4 to 12
	ABS-4			○	○			△	△																			△			△		△	5 to 13
	ABS			○	○			△	△	○																		△			△		△	5 to 14
	ABS			○	○			△	△		○																	△			△		△	4 to 13
	ABS			○	○			△	△		○																	△			△		△	
	ABS			○	○			△	△		○																	△			△		△	
	ABS			○	○			△	△		○																	△			△		△	
	ABS			○	○			△	△				○															△			△		△	5 to 14
	ABS			○	○			△	△				○															△			△		△	
	INC-1	Constant-speed control passing point incremental designation		○	○			△	△																			△			△		△	2 to 10
	INC-2			○	○			△	△																			△			△		△	3 to 11
	INC-3			○	○			△	△																			△			△		△	4 to 12
	INC-4			○	○			△	△																			△			△		△	5 to 13
	INC			○	○			△	△	○																		△			△		△	5 to 14
	INC			○	○			△	△		○																	△			△		△	4 to 13
	INC			○	○			△	△		○																	△			△		△	
	INC			○	○			△	△		○																	△			△		△	
	INC			○	○			△	△		○																	△			△		△	
	ABS			○	○			△	△				○															△			△		△	5 to 14
	INC			○	○			△	△				○															△			△		△	
	CPEND	Constant-speed control end					△																												1 to 2
Repetition of same control (used in speed switching control, constant-speed control)	FOR-TIMES	Repeat range start setting																									○								2
	FOR-ON																										○								
	FOR-OFF																										○								
	NEXT	Repeat range end setting																																	3
Simultaneous start	START	Simultaneous start																										○							2 to 3
Zeroing	ZERO	Zeroing start		○																															2
High-speed oscillation	OSC	High-speed oscillation start	△	○				△						○	○	○								△						△					5 to 10
Current value change: Servo	CHGA	Servo/virtual servo current value change		○	○																														3
Current value change: Encoder	CHGA-E	Encoder current value change		○	○																														
Current value change: Cam	CHGA-C	Cam shaft current value change		○	○																														

○: Item which must be set, △: Item which is set when required
*1 Only when reference axis speed is specified.
*2 (B) indicates a bit device.

7.2 Servo Motor/Virtual Servo Motor Shaft Current Value Change

In the real mode, the current value of the specified axis is changed.
In the virtual mode, the current value of the specified virtual servo motor shaft is changed.

Servo Instruction	Positioning Method	Number of Control Axes	Items Set on Peripheral Device																					Speed Change
			Common							Circular			Parameter block										Others		
			Parameter block No.	Axis	Address/travel	Command speed	Dwell time	M code	Torque limit value	Auxiliary point	Radius	Central point	Control unit	Speed limit value	Acceleration time	Deceleration time	Sudden stop deceleration time	Torque limit value	STOP input-time deceleration processing	Circular interpolation error permissible range	S-pattern ratio	Cancel	Start	
CHGA-C	Absolute	1		○	○																			Disable

○: Item which must be set
△: Item which is set when required

[Controls]

Control using CHGA instruction

(1) Executing the CHGA instruction changes the current value in the following procedure.
  (a) The start acceptance flag (M2001 to M2008/M2001 to M2032) corresponding to the specified axis is turned ON.
  (b) The current value of the specified axis is changed to the specified address.
  (c) Start acceptance is turned OFF on completion of the current value change.

(2) In the real mode, the current value of the specified axis is changed.

(3) In the virtual mode, the current value of the specified virtual servo motor shaft is changed.

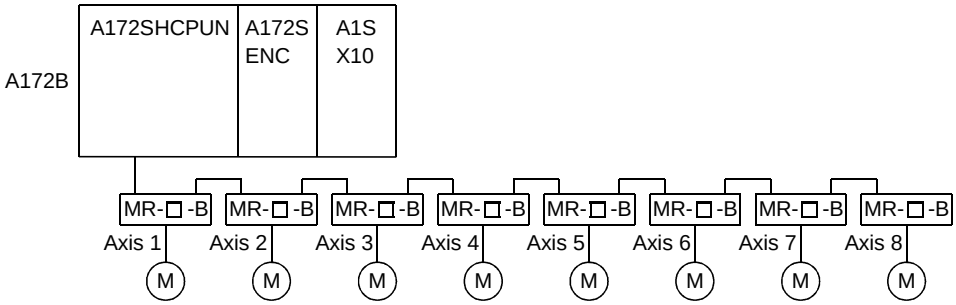
(4) The axis number used can be set within the following range.

A172SHCPUN	A173UHCPU(-S1)/ A273UHCPU-S3(32-axis feature)
Axis 1 to axis 8	Axis 1 to axis 32

[Program example]

A program for exercising current value change control in the real mode will be described under the following conditions.

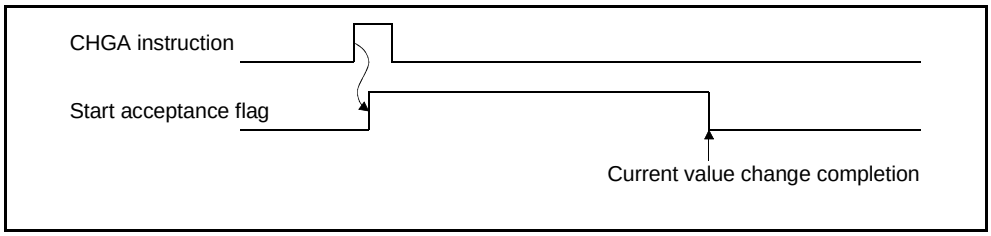
(1) System configuration
  The current value change control of axis 2 is performed.



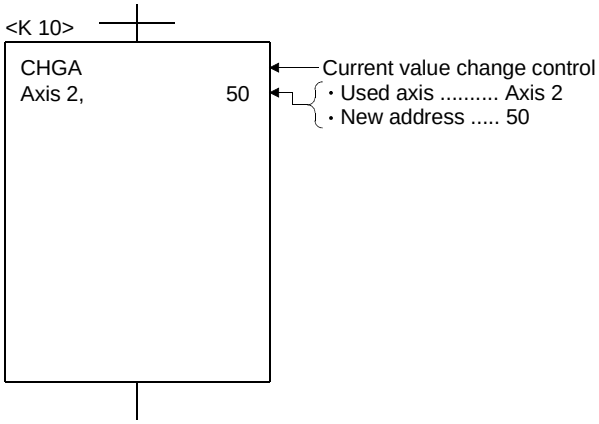
(2) In the real mode, the current value of the specified axis is changed.
(a) The current value change control conditions are indicated below.

Item	Setting
Servo program number	No.10
Control axis	Axis 2
New address	50

(3) Operation timing



(4) The axis number used can be set within the following range.



POINT

(1) Current value changing instructions

- When PLC ready (M2000) or PCPU ready (M9074) is OFF, a minor error 100 occurs and a current value change is not made.
- This change is made only during a stop. If a current value change is made while the specified axis is starting, a minor error 101 (start acceptance signal of the corresponding axis is ON) occurs and the current value change is not made.
- If the servo of the corresponding axis is not READY, a major error 1004 occurs and the current value change is not made.
- If the corresponding axis is in a servo error, a major error 1005 occurs and the current value change is not made.

For SV22

- Set the virtual servo motor shaft's current value change program within the virtual mode program number range set in "program mode assignment".
- Set the servo motor (output) shaft's current value change program within the real mode program number range.
- If a virtual servo motor shaft current value change is executed in the real mode, a servo program error 903 occurs and the current value change is not made.
- If a servo motor (output) shaft current value change is executed in the virtual mode, a servo program error 904 occurs and the current value change is not made.
- If a current value change is made during mode changing, a servo program error 907 (real→virtual changing) or 908 (virtual→real switching) occurs and the current value change is not made.

7.3 Synchronous Encoder Shaft Current Value Change Control (SV22 Only)

The current value of the synchronous encoder shaft specified in the virtual mode is changed.

Servo Instruction	Positioning Method	Number of Control Axes	Items Set on Peripheral Device																						Speed Change
			Common							Circular			Parameter block										Others		
			Parameter block No.	Axis	Address/travel	Command speed	Dwell time	M code	Torque limit value	Auxiliary point	Radius	Central point	Control unit	Speed limit value	Acceleration time	Deceleration time	Sudden stop deceleration time	Torque limit value	STOP input-time deceleration processing	Circular interpolation error permissible range	S-pattern ratio	Cancel	Start	
CHGA-E	Absolute	1		○	○																				Disable

○: Item which must be set
△: Item which is set when required

[Controls]

Control using CHGA-E instruction

(1) Executing the CHGA-E instruction changes the current value of the synchronous encoder shaft in the following procedure.
  (a) The synchronous encoder shaft current value changing flag (M2031/M2101 to M2112) corresponding to the specified synchronous encoder shaft is turned ON.
  (b) The current value of the specified synchronous encoder shaft is changed to the specified address.
  (c) The synchronous encoder shaft current value changing flag is turned OFF on completion of the current value change.

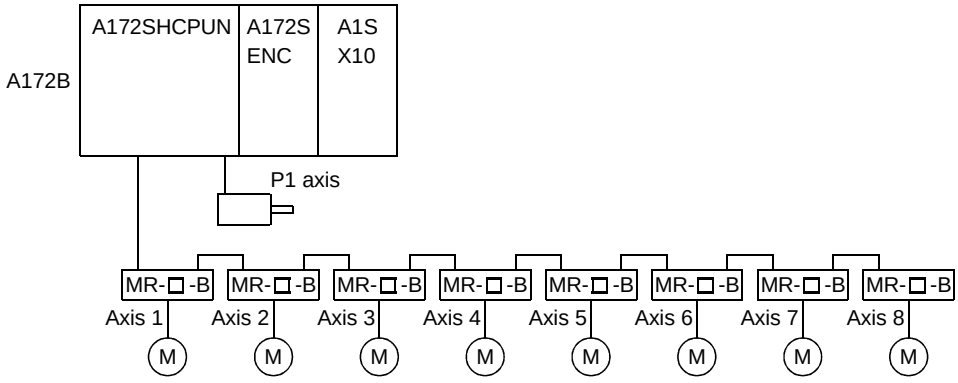
(2) The axis number used can be set within the following range.

A172SHCPUN	A173UHCPU(-S1)	A273UHCPU-S3 (32-axes feature)
Axis 1	Axis 1 to axis 4	Axis 1 to axis 12

[Program example]

A program for exercising the current value change control of the synchronous encoder shaft will be described under the following conditions.

(1) System configuration
  The current value change control of the synchronous encoder P1 axis is performed.

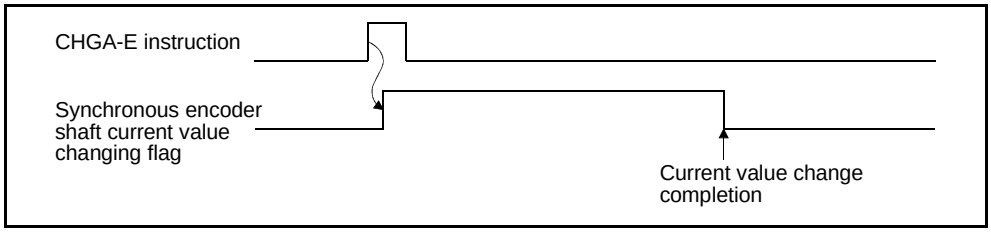


(2) Current value change control conditions

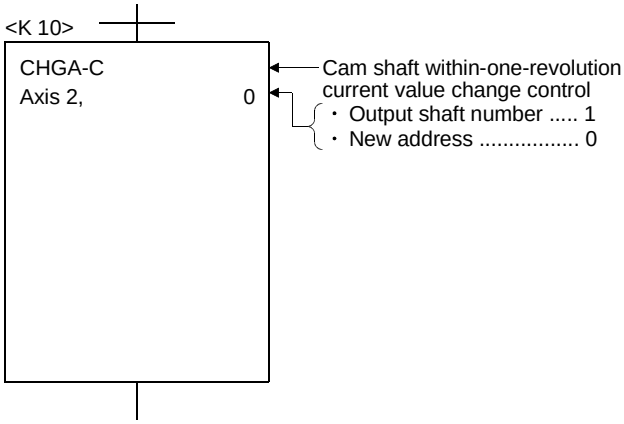
(a) The current value change control conditions are indicated below.

Item	Setting
Servo program number	No.10
Control axis.	1
New address	Indirect designation using D10, D11

(3) Operation timing



(4) Servo program



POINT

(1) Synchronous encoder current value changing instructions

- The current value change of a synchronous encoder is executed if operation is being performed in the virtual mode (during pulse input from the synchronous encoder).
  When the current value is changed, the feed current value of the synchronous encoder continues from the new value.
- The current value change of the synchronous encoder does not affect the current value of the output module.
- Set the synchronous encoder shaft's current value change program within the virtual mode program number range set in "program mode assignment".
- When PLC ready (M2000) or PCPU ready (M9074) is OFF, a minor error 100 occurs and a current value change is not made.
- If a synchronous encoder current value change is executed in the real mode, a servo program error 903 or 905 occurs and the current value change is not made. (903 when the current value change servo program is set to within the virtual mode program number range, or 905 when it is set to within the real mode program number range.)
- If a current value change is made during mode changing, a servo program error 907 (real→virtual changing) or 908 (virtual→real switching) occurs and the current value change is not made.

7.4 Cam Shaft Within-One-Revolution Current Value Change Control (SV22 Only)

Servo Instruction	Positioning Method	Number of Control Axes	Items Set on Peripheral Device																							Speed Change
			Common							Circular			Parameter block											Others		
			Parameter block No.	Axis	Address/travel	Command speed	Dwell time	M code	Torque limit value	Auxiliary point	Radius	Central point	Control unit	Speed limit value	Acceleration time	Deceleration time	Sudden stop deceleration time	Torque limit value	STOP input-time deceleration processing	Circular interpolation error permissible range	S-pattern ratio	Cancel	Start	
CHGA-C	Absolute	1		○	○																					Disable

○: Item which must be set
△: Item which is set when required

[Controls]

Control using CHGA-C instruction

(1) Executing the CHGA-C instruction changes the within-one-revolution current value of the specified cam shaft to the specified address.

(2) The cam shaft may be starting.

(3) The axis number used can be set within the following range.

A172SHCPUN	A173UHCPU(-S1)/ A273UHCPU-S3(32-axes feature)
Axis 1 to axis 8	Axis 1 to axis 32

[Program example]

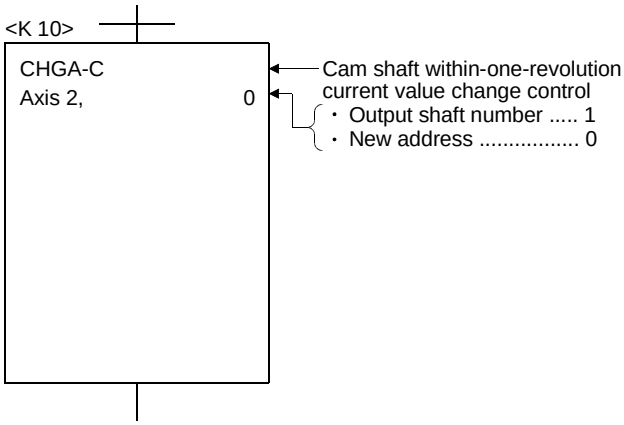
A program for exercising the within-one-revolution current value change control of the cam shaft will be described under the following conditions.

(1) Current value change control conditions

(a) The current value change control conditions are indicated below.

Item	Setting
Servo program number	No.10
Output shaft number	2
New address	0

(2) Servo program



POINT

(1) Cam shaft within-one-revolution current value changing instructions

- If a new within-one-revolution current value is outside the range 0 to (one-revolution pulse count - 1), a minor error 6120 occurs and a current value change is not made.
- Set the cam shaft within-one-revolution current value change program within the virtual mode program number range set in "program mode assignment".
- When PLC ready (M2000) or PCPU ready (M9074) is OFF, a minor error 100 occurs and a current value change is not made.
- If a cam shaft within-one-revolution current value change is executed in the real mode, a servo program error 903 or 905 occurs and the current value change is not made. (903 when the current value change servo program is set to within the virtual mode program number range, or 905 when it is set to within the real mode program number range.)
- If a current value change is made during mode changing, a servo program error 907 (real→virtual changing) or 908 (virtual→real switching) occurs and the current value change is not made.

8. MOTION DEVICES

The motion registers (#0 to #8191) and coasting timer (FT) are available as motion CPU (PCPU)-dedicated devices.
They can be used in operation control (F/FS) programs or transition (G) programs. They cannot be accessed directly from the PLC. When using them on the PLC side, assign them to the PLC devices.

8.1 Motion Registers (#0 to #8191)

Motion Device	Item	A172SHCPUN/A173UHCPU(-S1)/A273UHCPU-S3
Motion register (#)	Number of points	8192 points (#0 to #8191)
	Data size	16 bits/point
	Latch	Latched with the exception of SFC-dedicated devices. (All points are cleared by latch clear operation.)
	Usable tasks	Normal, event, NMI
	Access	Read and write enabled in whole range

(1) Motion register list

(a) Common to all Operating Systems.

Device Number	Application	Remarks
#0	User devices (8000 points)	Cleared by latch clear.
#8000	SFC dedicated devices (192 points)	Cleared at power-on or key-reset only. (66 points)
#8066 #8191		Not cleared (126 points)

POINT
The motion registers (#) cannot be set as indirectly specified devices of mechanical system programs.

(2) SFC-dedicated devices (#8000 to #8191)

The SFC-dedicated devices are indicated below.

The device's refresh cycle is indicated when the signal direction is "status", or its import cycle when the signal direction is "command".

Device Number	Signal Name		Signal Direction		Refresh Cycle	Import Cycle
			Status	Command		
#8000	Seventh error information in past (Oldest error information)	SFC error history (8 errors) (64 points)	○	—	At error occurrence	—
#8008	Sixth error information in past					
#8016	Fifth error information in past					
#8024	Fourth error information in past					
#8032	Third error information in past					
#8040	Second error information in past					
#8048	First error information in past					
#8056	Latest error information					
#8064 to #8191	User unusable (128 points)					

(3) SFC error history devices

The error information which occurred after power-on of the CPU is stored as a history of up to eight past errors. The latest error is stored in #8056 to #8063. All errors, including the SFC control errors and the conventional minor, major, servo, servo program and mode changing errors, have been integrated into this history.

At error occurrence, the "SFC error detection signal M2039" is also set.

The error information is as indicated below.

NO.	Signal Name		Description	
			SFC control errors	Conventional errors
+0	Error SFC program number		0 to 255 : SFC program number in error -1 : Independent of SFC program	-1
1	Error type		1 :F/FS 2 :G -2 :SFC chart -1 :K or other (not any of F/FS, G and SFC chart)	3: Minor/major error (Output module in real mode/virtual mode (SV22 only)) 4: Minor/major error (virtual servo motor shaft) (SV22 only) 5: Minor/major error (synchronous encoder shaft) (SV22 only) 6: Servo error 7: Servo program error 8: Mode change error (SV22 only) 9: Manual pulse generator axis setting error 10: Test mode request error 11: PCPU WDT error 12: Personal computer link communication error
2	Error program number		0 to 4095 : F/FS, G, K program number 0 to 255 : GSUB program number -1 : Independent of F/FS, G, K, GSUB	0 to 4095 : Servo program number when error type is "3 (in real mode)", "4" or "7" -1 : Others (including no start, JOG, manual pulse generator or test mode zeroing/servo startup/servo diagnostic start)
3	Error block number/SFC list line number/axis number		0 to 8191 : F/FS or G program's block number (line number) when error type is "1" or "2" 0 to 8188 : SFC list line number when error type is "-2" -1 : Independent of block when error type is "-1" or error type is "1" or "2"	1 to 32 : Corresponding axis number when error type is any of "3" to "6" -1 : Others
4	Error code		16000 and later (Refer to "15. Error List".)	• Conventional error code (less than 16000) when error type is any of "3" to "6" • Error code stored in D9190 when error type is "7" • Error code stored in D9193 (A273UH-S3)/D9195 (A172SH) when error type is "8" • -1 when error type is "9" or "10" • Error code stored in D9184 when error type is "11" • Error code stored in D9196 when error type is "12"
5	Error occurrence time	Year /month	The PLC clock data (D9025, D9026, D9027) are set. (BCD code, year in its lower 2 digits)	
6		Day/ hour		
7		Minute/ second		

(4) SFC error detection signal (M2039)
(Refresh cycle UH: 10ms, SH: PLC scan time)
The SFC error detection signal (M2039) turns ON when any of the errors detected by the motion CPU occurs.
At error occurrence, data are set to the error devices in the following procedure.
(a) The error code is set to the corresponding axis or error devices.

(b) The error detection signal of the corresponding axis or error is turned ON.

(c) Error information is set to the above "SFC error history devices (#8000 to #8063)".

(d) The SFC error detection signal (M2039) is turned ON.

In the user program, when the "SFC error detection signal (M2039)" turns ON, read the error history, and then reset the "SFC error detection signal (M2039)". After that, at occurrence of a new error, "SFC error detection signal (M2039)" turns ON again.

POINT
• Resetting the "SFC error detection signal (M2039)" will not reset (clear to zero) the "SFC error history devices (#8000 to #8063)". After power-on, they always manage the error history continuously. • Set the clock data and clock data read request (M9028) in the user program.

8.2 Coasting Timer (FT)

Motion Device	Item	A273UHCPU-S3/A172SHCPUN
Coasting timer (FT)	Number of points	1 point (FT)
	Data size	32 bits/point (-2147483648 to 2147483647)
	Latch	No latch. Cleared to zero at power-on or key-reset.
	Usable tasks	Normal, event, NMI
	Access	Read only enabled
	Timer specifications	888μs timer (Current value (FT) is incremented by 1 per 888μs.)

9. TASK OPERATIONS

9.1 Task Definitions

When to execute SFC program processing can be set only once in the program parameter (refer to Chapter 11 SFC Parameters) per program.
These processing timing brackets are called tasks.
Roughly classified, there are the following three different tasks.

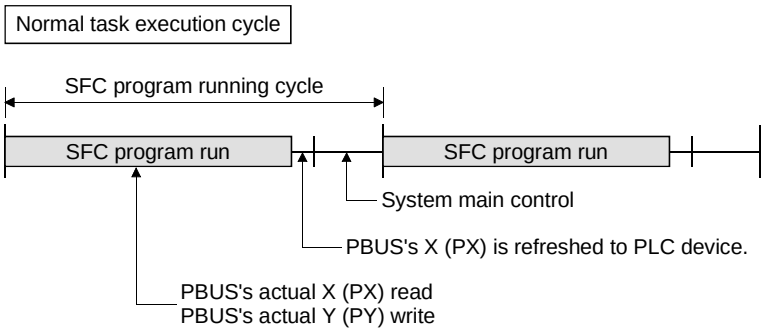
Task Type	Description
Normal task	Executed in motion main cycle (free time).
Event task	1. Executed in fixed cycle (1.7ms, 3.5ms, 7.1ms, 14.2ms). 2. Executed when the input set to the event task factor among external interrupts (16 points of AI61) turns ON. 3. Executed by an interrupt from the PLC.
NMI task	Executed when the input set to the NMI task factor among external interrupts (16 points of AI61) turns ON.

(1) Normal task

[Operations]

An SFC program is run in the main cycle (free time) of the motion side CPU (PCPU) processing.

The processing is outlined as follows.



[Points]

(a) The SFC program which includes motion control steps should be set to a normal task.

(b) During execution of an event or NMI task, the execution of the normal task is suspended.
Note that since the normal task allows the event task disable instruction (DI) to be described in an operation control step, the event task can be disabled in the area enclosed by the event task disable instruction (DI) and event task enable instruction (EI).

(2) Event task

[Operations]

An event task runs an SFC program at occurrence of an event.
There are the following events.

(a) Fixed cycle
An SFC program is run periodically in any of 1.7ms, 3.5ms, 7.1ms and 14.2ms cycles.

(b) External interrupt (16 points of I0 to I15)
Among 16 points of the AI61 (16-point interrupt module) loaded in the motion slot, an SFC program is run when the input set for an event task turns ON.

(c) PLC interrupt
An SFC program is run when the ITP instruction is executed in the sequence program.

[Points]

(a) You can set plural events to one SFC program. However, you cannot set plural fixed cycles.

(b) Multiple SFC programs can be run by one event.

(c) Motion control steps cannot be executed during the event task.

(d) The event task cannot be executed when it is disabled by the normal task. The event that occurred during event task disable is executed the moment the event task is enabled.

[Errors]

An attempt to execute a motion control step in an SFC program set to the event task will result in an SFC program error 16113 and stop the SFC program that is running.

(3) NMI task

[Operations]

An SFC program is run as soon as the input set to the NMI task factor among the external interrupts (16 points of the AI61) turns ON.

[Points]

(a) Among the normal, event and NMI tasks, the NMI task has the highest priority.

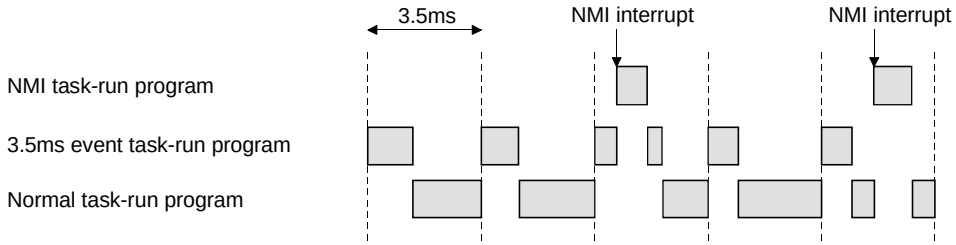
(b) If the event task is disabled (DI) by the normal task, the interruption of the NMI task is executed, without being masked.

[Errors]

During an NMI task, a motion control step cannot be executed.
Presence of a motion control step during an NMI task will result in an SFC program error 16113 and stop the SFC program which is running.

9.2 Task Execution Status

The following example gives how the SFC programs run by multiple tasks are executed.



When there are programs which are run by the NMI task, 3.5ms fixed-cycle even task and normal task,

(1) The 3.5ms fixed-cycle event task run its program at intervals of 3.5ms;
(2) The NMI task runs its program with the highest priority when an NMI interrupt is input; and
(3) The normal task runs its program at free time.
as shown above.

[Points]
A single SFC program can be run partially by another task by setting the area to be executed by another task as a subroutine and setting a subroutine running task as another task.
Example) No. 0 Main SFC program Normal task
No. 1 Subroutine Event task (3.5ms cycle)

10. PROGRAMMING INSTRUCTIONS

10.1 Task Definitions

Note the following points when SET/RST/DOUT of the bit devices which are enabled for SET/RST/DOUT from a sequence program, e.g. M devices, is executed in an SFC program.

(1) The bit devices which are SET/RST/DOUT in an SFC program should not be SET/RST/OUT in a sequence program.

(2) Reversely, the bit devices which are SET/RST/OUT in a sequence program should not be SET/RST/DOUT in an SFC program.

(3) The above exclusive control should be exercised for each bit device in increments of consecutive 16 points, starting with the device number which begins with a multiple of 16.

[Points]

(1) The user should predetermine how to use bit devices, e.g.
M112 to M127: SET/RST executed on the SFC program side
M128 to M143: SET/RST executed on the sequence program side

(2) Care should be taken since the first command device of each axis does not begin with a multiple of 16.

10.2 SET/RST Response Delays of Motion-Dedicated Bit Devices

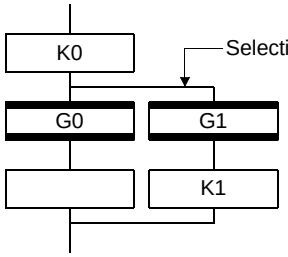
When command devices among the following motion-dedicated bit devices are SET/RST in an SFC program, there will be a delay in refresh time as indicated below.
There will also be a refresh delay when devices are SET/RST in an SFC program and their results are used in the SFC program.

	10ms	END(Note)
A172SHCPUN	M2000 to M2015	M2016 to M2047
A173UHCPU(-S1) A273UHCPU-S3(32-axes feature)	M2000 to M2047	M2048 to M2095

(Note): END indicates a "sequence program scan time".

10.3 Cancel Start

When a cancel start has been set in the setting items of the servo program which was started at the motion control step of an SFC program, the cancel of the running servo program is valid but the servo program specified to start after a cancel is ignored, without being started.
The following example shows an SFC program which exercises control equivalent to a cancel start.



Providing transition G1 with cancel device condition specified in servo program K0 will cancel run of servo program K0 and allow servo program K1 to start.

10.4 Indirect Designation using Motion Devices

The motion registers #0 to #8191 cannot be used to make indirect designation in servo and mechanical system programs.
When using the motion register values in servo or mechanical system programs, assign them to PLC devices.

10.5 Sequence Programs

(1) You cannot use the SVST, CHGV, CHGA and CHGT (DSFRP and DSFLP also included when the A172SHCPUN is used) motion-dedicated instructions in sequence programs.
Doing so will cause an error (INSTRUCT CODE ERR.) in the PLC.

(2) When reading and using 2-word monitor data, such as a feed current value, or 2-word data written with an SFC program, always import it into a user device once using the "DMOV(P)" instruction, and perform magnitude comparison or similar operation using the device that imported the data.

11. SFC PARAMETERS

Two different SFC parameters are available: "task parameters" designed to control the tasks (normal task, event task, NMI task) and "program parameters" to be set per SFC program.
Their details will be explained below.

11.1 Task Parameters

No.	Item		Setting Item	Initial Value	Remarks
1	Number of consecutive transitions	Normal task	1 to 30	3	These parameters are imported when PLC ready (M2000) turns from OFF to ON and used for control thereafter. When setting/changing the values of these parameters, turn PLC ready (M2000) OFF.
2	Interrupt setting		Set whether the event task or NMI task is used for external interrupt inputs (I0 to I15).	Event task	

(1) Number of consecutive transitions

[Description]
With "execution of active step → judgment of next transition condition → transition processing performed when condition enables (transition of active step)" defined as a single basic operation of SFC program running control in the execution cycle of the corresponding task, this operation is performed for the number of active steps to terminate processing once.
The same operation is then repeated in the next cycle to perform processing.
In this case, the transition destination step is executed in the next cycle when the transition condition enables.

Consecutive transition control indicates that transition destination steps are executed one after another in the same one execution cycle when their transition conditions have enabled (single basic operation is performed consecutively).
Set the number of consecutive transitions in this case.
Control exercised is common to the SFC programs run by normal tasks.

POINT
Set the number of consecutive transitions to each of the SFC programs run by event and NMI tasks.

[Errors]
These parameters are imported and checked when PLC ready (M2000) turns from OFF to ON.
When the value that was set is outside the setting range, the following SFC error is set and the initial value is used to exercise control.

Error Code	Error Factor		Error Processing	Corrective Action
	Name	Definition		
17000	Normal task consecutive transition count error	The normal task's consecutive transition count of the SFC program run by the normal task is outside the range 1 to 30.	The initial value of 3 is used for control.	Turn PLC ready (M2000) OFF, make correction to set the value of within the range, and write it to the CPU.

(2) Interrupt setting

[Description]
Set whether 16 interrupt input points (I0 to I15) of the AI61 interrupt input module loaded in the motion slot are used as NMI or event task inputs.
Setting can be made freely per point.
All points default to event tasks.

[Errors]
None.

11.2 Program Parameters

Set the following parameters per SFC program.

No.	Item	Setting Range	Initial Value	Remarks
1	Start setting	Automatically started or not	Not	These parameters are imported when PLC ready (M2000) turns from OFF to ON and used for control thereafter. When setting/changing the values of these parameters, turn PLC ready (M2000) OFF. POINT The settings of "executed task", "number of consecutive transitions" and "END operation" are invalid for the subroutine called program. For the subroutine called program, "executed task" and "number of consecutive transitions" depend on the call source program setting. "END operation" is controlled as "end".
2	Executed task	Only one of normal, event and NMI tasks	Normal task	
		When you have set the event task, further set the event which will be enabled. Always set any one of the following 1 to 3. 1. Fixed cycle One or none of 1.7ms, 3.5ms, 7.1ms and 14.2ms. 2. External interrupt (make selection from those set to event task) More than one interrupt can be set from among I0, I1, I2, I3, I4, I5, I6, I7, I8, I9, I10, I11, I12, I13, I14 and I15. 3. PLC interrupt OR may also be used to set 1 to 3. The same event may be shared among multiple SFC programs.	None	
		When you have set the NMI task, further set the interrupt input which will be enabled. 1. External interrupt (make selection from those set to NMI task) More than one interrupt can be set from among I0, I1, I2, I3, I4, I5, I6, I7, I8, I9, I10, I11, I12, I13, I14 and I15.		
3	Number of consecutive transitions	1 to 10 Set the number of consecutive transitions to the program set to the event or NMI task.	1	
4	END operation	End/continue Set the operation mode of the END step to the program set to the event or NMI task.	End	

(1) Start setting

[Description]
The following control is governed by "automatically started or not" setting.

• Program run by normal task

No.	Item	When "automatically started"	When "not automatically started"
1	Start control	In the main cycle after PLC ready (M2000) has turned from OFF to ON, the program is run from the initial (first) step in accordance with the normal task's consecutive transition count.	The program is started by the SFC start instruction (SFCS) from the PLC or by a subroutine call/start (GSUB) made from within the SFC program. • When started by the SFCS instruction In the main cycle after execution of the SFCS instruction, the program is run from the initial (first) step in accordance with the normal task's consecutive transition count. • When subroutine started In the (next) main cycle after execution of GSUB, the program is run from the first step in accordance with the normal task's consecutive transition count. • When subroutine called The program is run in the same cycle from the first step.
		After that, in the motion main cycle, the program is run continuously by the number of consecutive transitions of the normal task. (The settings of "executed task" and "number of consecutive transitions" of the subroutine called program are invalid. The normal task is used for control.)	
2	End control	Ends its own program. The program is started when it is started again by the SFC start instruction (SFCS) from the PLC or by a subroutine call/start (GSUB) made from within the SFC program.	

• Program run by event task

No.	Item	When "automatically started"	When "not automatically started"
1	Start control	At occurrence of a valid event after PLC ready (M2000) has turned from OFF to ON, the program is run from the initial (first) step in accordance with the number of consecutive transitions of the corresponding program.	The program is started by the SFC start instruction (SFCS) from the PLC or by a subroutine call/start (GSUB) made from within the SFC program. • When started by the SFCS instruction At occurrence of a valid event after execution of the SFCS instruction, the program is run from the initial (first) step in accordance with the number of consecutive transitions of the corresponding program. • When subroutine started At occurrence of a valid event after execution of GSUB, the program is run from the first step in accordance with the number of consecutive transitions of the corresponding program. • When subroutine called The program is run immediately from the first step.
		After that, at occurrence of a valid event, the program is run continuously by the number of consecutive transitions of the corresponding program. (The subroutine called program is controlled in accordance with the "executed task" and "number of consecutive transitions" of the call source program.)	
2	End control	As specified for END operation.	

• Program run by NMI task

No.	Item	When "automatically started"	When "not automatically started"
1	Start control	At occurrence of a valid event after PLC ready (M2000) has turned from OFF to ON, the program is run from the initial (first) step in accordance with the number of consecutive transitions of the corresponding program.	The program is started by the SFC start instruction (SFCS) from the PLC or by a subroutine call/start (GSUB) made from within the SFC program. • When started by the SFCS instruction At occurrence of a valid event after execution of the SFCS instruction, the program is run from the initial (first) step in accordance with the number of consecutive transitions of the corresponding program. • When subroutine started At occurrence of a valid event after execution of GSUB, the program is run from the first step in accordance with the number of consecutive transitions of the corresponding program. • When subroutine called The program is run immediately from the first step.
		After that, at occurrence of a valid event, the program is run continuously by the number of consecutive transitions of the corresponding program. (The subroutine called program is controlled in accordance with the "executed task" and "number of consecutive transitions" of the call source program.)	
2	End control	As specified for END operation.	

[Errors]
None.

POINT
When you want to automatically restart the program run by the normal task from the initial step at end of a single cycle operation, write the program so that it is not ended by END but it returns to the starting step by a jump.

(2) Executed task

[Description]
Set the timing (task) to run a program.
Specify whether the program will be run by only one of the "normal task (main cycle), event task (fixed cycle, external interrupt, SCPU interrupt) and NMI task (external interrupt)".

When you have set the event task, you can set multiple events out of the "fixed cycle, external interrupt (for event task) and SCPU interrupt".
Note that multiple fixed cycles cannot be set to a single SFC program.

Example) Interrupt setting: Inputs for event task I6, I7, I8, I9, I10, I11, I12, I13, I14 and I15
SFC program No. 10 - event: Fixed cycle (3.5ms)
SFC program No. 20 - event:
Fixed cycle (1.7ms) + external interrupt (I6)
SFC program No. 30 - event:
External interrupts (I7, I15) + SCPU interrupt

When you have set the NMI task, you can set multiple interrupt inputs out of the external interrupts (for NMI task).

Example) Interrupt setting: Inputs for NMI task I0, I1, I2, I3, I4, I5
SFC program No. 10 - NMI: I0
SFC program No. 20 - NMI: I1 + I2
SFC program No. 30 - NMI: I5

[Errors]
This parameter is imported when PLC ready (M2000) turns from OFF to ON, and is checked at an SFC program start (automatic start, start from PLC or subroutine start).
When the value is unauthorized, either of the following SFC errors is set and the initial value is used for control.

Error Code	Error Factor		Error Processing	Corrective Action
	Name	Definition		
17010	Executed task setting is illegal	Among the normal, event and NMI tasks, more than one or none of them has been set.	The initial value (normal task) is used for control.	Turn PLC ready (M2000) OFF, make correction, and write a correct value to the CPU.
17011	Executed task setting is illegal (event)	Two or more fixed cycles of the event task have been set.		

POINT

(1) Since the executed task setting can be made per SFC program number, multiple programs need not be written for single control (machine operation) to divide execution timing-based processings.
For example, this can be achieved easily by subroutine starting the areas to be run in fixed cycle and to be run by external interrupt partially in an SFC program run by the normal task.
(2) The executed task of the subroutine called program is controlled like that of the call source program.
Hence, this setting is invalid but it is recommended to make the same setting as the call source program.

(3) Number of consecutive transitions

[Description]
Set the number of consecutive transitions to each program run by the event or NMI task.
Refer to Section 11.1 for the "number of consecutive transitions".

[Errors]
This parameter is imported when PLC ready (M2000) turns from OFF to ON, and is checked at an SFC program start (automatic start, start from PLC or subroutine start).
When the value is unauthorized, either of the following SFC errors is set and the initial value is used for control.

Error Code	Error Factor		Error Processing	Corrective Action
	Name	Definition		
17001	Event task consecutive transition count error	The set number of consecutive transitions of the SFC program started by the event task is outside the range 1 to 10.	The initial value of 1 is used for control.	Turn PLC ready (M2000) OFF, make correction to set the value within the range, and write it to the CPU.
17002	NMI task consecutive transition count error	NMI task consecutive transition count error		

POINT

The number of consecutive transitions of the subroutine called program is the same as that of the call source program.
Hence, this setting is invalid but it is recommended to make the same setting as the call source program.

(4) END operation

[Description]
Set the operation to be performed at execution of the END step to the program run by the event or NMI task.
This varies the specifications for the following items.

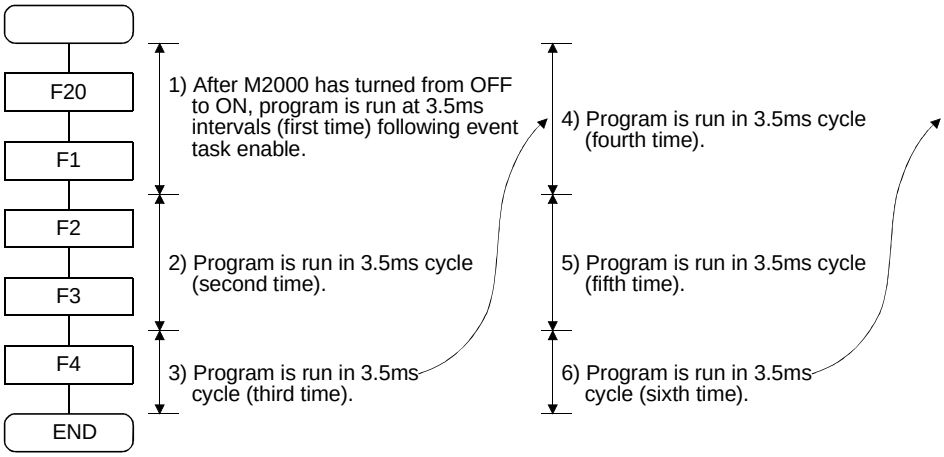
NO.	Item	When "Ended"	When "Continued"
1	Control at END execution	Ends its own program.	Ends the run of its own program with the event/interrupt made this time.
2	Restart after END execution	Started by the SFC start instruction (SFCS) from the PLC again or started by a subroutine call/start (GSUB) made from within the SFC program.	Restarted at occurrence of the next valid event/interrupt, and run from the initial (first) step in accordance with the number of consecutive transitions of the corresponding program. Thereafter, at occurrence of a valid event/interrupt, the program is controlled in accordance with the number of consecutive transitions of the corresponding program.
3	Restart after end by clear step CLR	Started by the SFC start instruction (SFCS) from the PLC again or started by a subroutine call/start (GSUB) made from within the SFC program.	

POINT

The END operation of the subroutine called program is controlled as an "end".

- The following operation example assumes that the END operation is "continued".
  Program parameters

  - Automatically started
  - Executed task = event 3.5ms
  - Number of consecutive transitions = 2
  - END operation "Continued"



12. HOW TO RUN SFC PROGRAM

12.1 How to Start SFC Program

An SFC program runs while PLC ready M2000 is ON.
An SFC program may be started by any of the following three methods.

(1) Automatic start
(2) Start from SFC program
(3) Start from PLC

Set the starting method in the program parameter per SFC program.
Refer to Chapter 11 SFC Parameters for parameter setting.

12.1.1 Automatic start

[Operations]

An automatic start is made by turning PLC ready M2000 ON.

12.1.2 Start from SFC program

[Operations]

A start is made by executing a subroutine call/start step in the SFC program.
For details of the subroutine call/start step, refer to Chapter 4 SFC Programs.

12.1.3 Start from PLC (Sequence instruction SFCS)

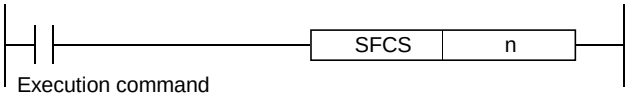
The SFC program can be started by executing the following instruction in the sequence program.

- SFC program start request instruction (SFCS)

	Usable Devices																				Digit Designation	Number of Steps	Subset	Index	Carry Flag	Error Flag	
	Bit devices							Bit devices									Const-ants		Point-ers		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
(D)																							13		○		○	○
N										○	○	○					○	○										

Sequence program

[Execution condition]

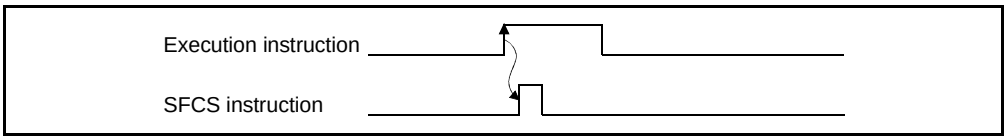


	Settings		Setting Range	
n	SFC program number to be started	SFC program number to be started	0 to 255	
		When specified indirectly (1 word)	A172SHCPUN	A173UHCPU (-S1) /A273UHCPU-S3
			D0 to D1023 W0 to W3FF R0 to R8191	D0 to D8191 W0 to W1FFF R0 to R8191

[Controls]

A request to start the specified SFC program is given on the leading edge (OFF→ON) of the SFCS instruction execution command in the sequence program. The SFC program to be started may be run by any of the normal task, event task and NMI task.

(1) This instruction is always valid in any of the real mode, virtual mode and mode changing status.



[Errors]

At occurrence of any of the following errors, an SFC error is set to the SFC-dedicated devices "SFC error history devices (#8000 to #8039)" and SFC error detection M2039, and the SFC program is not started.

<SFC error history devices>

#8056	Error SFC program number		Started SFC program number
#8057	Error type		-1
#8058	Error program number		-1
#8059	Error block number		-1
#8060	Error code		* (indicated below)
#8061	Error occurrence time	Year/month	??
#8062		Day/hour	??
#8063		Minute/ second	??

M2039	SFC error detection signal	(ON)

Error Code	Error Factor		Error Factor	Corrective Action
	Corrective Action	Definition		
16000	PLC ready OFF (SFCS)	At a start made by SFCS, PLC ready (M2000) or PCPU ready (M9074) is OFF.	The specified SFC program does not start.	Provide ON of PLC ready (M2000) and PCPU ready (M9074) as start interlocks.
16001	SFC program number error (SFCS)	At an SFC program start made by SFCS, the SFC program number specified is outside the range 0 to 255.		Check the SFC program number, and correct it to a correct sequence program.
16002	No SFC program (SFCS)	At an SFC program start made by SFCS, the specified SFC program does not exist.		Check the SFC program number, and correct it to a correct sequence program, or create an SFC program not yet created.
16003	Double start error (SFCS)	At an SFC program start made by SFCS, the same SFC program is already starting.		Double start should be managed on the user side. Provide the user's starting signal as a start interlock in the sequence program.

12.2 How to End SFC Program

[Operations]

(1) The SFC program is ended by executing END set in itself.
(2) The SFC program is stopped by turning OFF the PLC ready signal M2000.
(3) The program can be ended by the clear step.
For details of the clear step, refer to the section of the clear step in Chapter 4 SFC Programs.

[Points]

(1) Multiple ENDs can be set in a single SFC program.

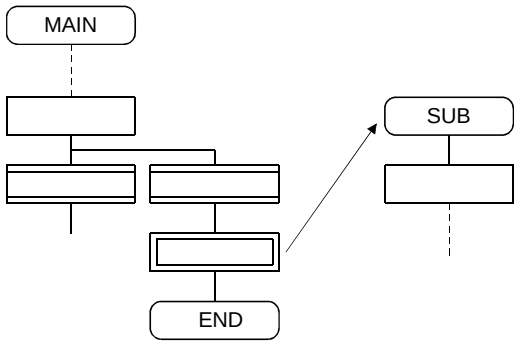
12.3 Clear Step in the SFC Program

Executing the clear step set in the SFC program stops the run of the SFC program specified in the clear step.

12.4 How to Change from One SFC Program to Another

Use a subroutine start to stop the SFC program which is running and switch it to another SFC program.

SFC program changing example using subroutine start



12.5 How to Manage the Running Programs

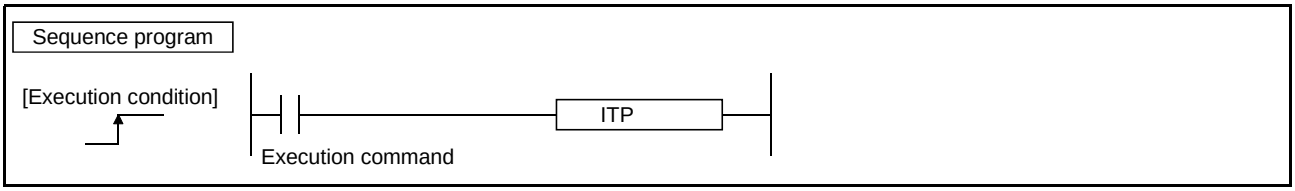
There are no specific information that indicates which SFC program is running. Use a user program (SFC program/sequence program) to manage the running program.

12.6 SCPU to PCPU Interrupt Instruction (Sequence instruction ITP)

Executing the following instruction in a sequence program generates an interruption to the motion CPU.

- PCPU interrupt instruction (ITP)

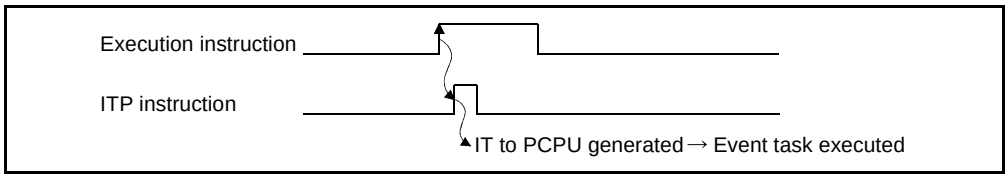
	Usable Devices																				Digit Designation	Number of Steps	Subset	Index	Carry Flag	Error Flag	
	Bit devices							Bit devices									Cons-tants		Point-ers		Level							
	X	Y	M	L	S	B	F	T	C	D	W	R	A0	A1	Z	V	K	H	P	I	N					M9012	M9010	M9011
(D)																							13					
N																												



[Controls]

An interruption to the motion CPU (PCPU) is generated on the leading edge (OFF→ON) of the ITP instruction execution command in the sequence program. When an interruption is generated by the SCPU, the motion CPU processes the active step of the SFC program to be executed at a "PLC interrupt".

(1) This instruction is always valid in any of the real mode, virtual mode and mode changing status.



(2) When the motion side is in the DI (interrupt disable) status, event processing stands by until the EI (interrupt enable) instruction is executed.

[Errors]

At occurrence of the following error, an SFC error is set to the SFC-dedicated devices "SFC error history devices (#8000 to #8039)" and SFC error detection M2039, and the SFC program's active step to be executed at a "PLC interrupt" is not processed.

<SFC error history devices>

#8056	Error SFC program number		-1
#8057	Error type		-1
#8058	Error program number		-1
#8059	Error block number		-1
#8060	Error code		* (indicated below)
#8061	Error occurrence time	Year/month	??
#8062		Day/hour	??
#8063		Minute/ second	??

M2039	SFC error detection signal	(ON)

Error Code	Error Factor		Error Processing	Corrective Action
	Name	Definition		
16004	PLC ready OFF (ITSP)	ITP was executed with PLC ready (M2000) or PCPU ready (M9074) OFF.	ITP was executed with PLC ready (M2000) or PCPU ready (M9074) OFF.	ITP was executed with PLC ready (M2000) or PCPU ready (M9074) OFF.

13. SFC PROGRAM CONTROLLING OPERATIONS

13.1 Operation Performed at CPU Power-Off or Key-Reset

When the CPU is powered off or a key reset operation is performed, SFC programs run as described below.

(1) When the CPU is powered off or a key reset operation is performed, SFC programs stop running.

(2) At CPU power-off or key-reset, the contents of the motion registers #0 to #7999 are held. Initialize them in SFC programs as required.

(3) After CPU power-on or key reset processing, SFC programs run as described below.
- The SFC programs set to start automatically are run from the beginning by turning PLC ready M2000 ON in the sequence program.
- The other SFC programs are also run from the beginning when started.

13.2 Operation Performed when CPU Is Put in RUN Mode

When the CPU is set to the RUN mode, the following operation is performed.

(1) When PLC ready M2000 is ON
- The SFC programs set to start automatically run from the initial step.
- The output states are governed by the PLC side "STOP→RUN time output mode" parameter setting.

(2) When PLC ready M2000 is OFF
- The SFC programs do not run until M2000 is turned ON.
- The output states are governed by the PLC side "STOP→RUN time output mode" parameter setting.

13.3 Operation Performed when CPU Is Switched from RUN to STOP

(1) When the CPU is placed in the STOP mode, SFC programs stop.

(2) When SFC programs are stopped in the STOP mode, all outputs turn OFF.

13.4 Operation Performed when CPU is set to PAUSE or STEP-RUN

When the CPU is set to PAUSE or STEP-RUN, SFC programs continue processing without stopping.

13.5 Operation Performed when PLC Ready (M2000) Turns OFF/ON

[M2000 OFF→ON]

If there is no fault when PLC ready (M2000) turns from OFF to ON, the PCPU ready flag M9074 turns ON.
When this PCPU ready flag M9074 turns ON, SFC programs can be run.

[M2000 ON→OFF]

When PLC ready (M2000) turns OFF, SFC programs stop running and the PCPU ready flag M9074 turns OFF.
Since outputs are held, turn OFF necessary outputs in the sequence program after the PCPU ready flag M9074 has turns OFF.

[Points]

(1) While the PCPU ready flag M9074 is ON, the outputs Y of the PBUS do not provide data to actual outputs if write is performed from the sequence program. However, while the PCPU ready flag M9074 is OFF, the outputs Y of the PBUS provide data to actual outputs when write is performed from the sequence program.

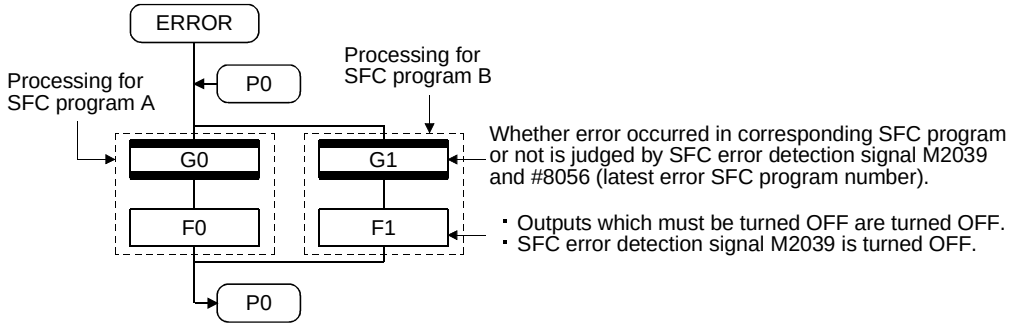
POINT
When the PLC ready signal M2000 turns OFF, SFC programs stop but outputs Y in the SFC programs do not turn OFF. Turn them OFF as required in the sequence program.

(2) SFC program run governed by RUN/STOP and M2000 ON/OFF

		PLC Ready M2000 Is ON	PLC Ready M2000 Is OFF
RUN	SFC programs	Run	SFC programs stop
	Outputs	Turn ON/OFF according to program	Outputs held
STOP	SFC programs	SFC programs stop	←
	Outputs	All outputs turn OFF	←
STOP ↓ RUN	SFC programs	SFC programs set to start automatically restart from beginning	SFC programs remain stopped
	Outputs	As set in sequence parameter "STOP→RUN time output mode"	As set in sequence parameter "STOP→RUN time output mode"

13.6 Error-Time Operation

Outputs are held if SFC programs stop due to error occurrence.
To turn OFF outputs at error occurrence, run the following SFC program.



At SCPU WDT ERROR occurrence, all SFC programs running stop and all outputs turn OFF.

14. USER FILES

This chapter provides a user file list and directory structure.

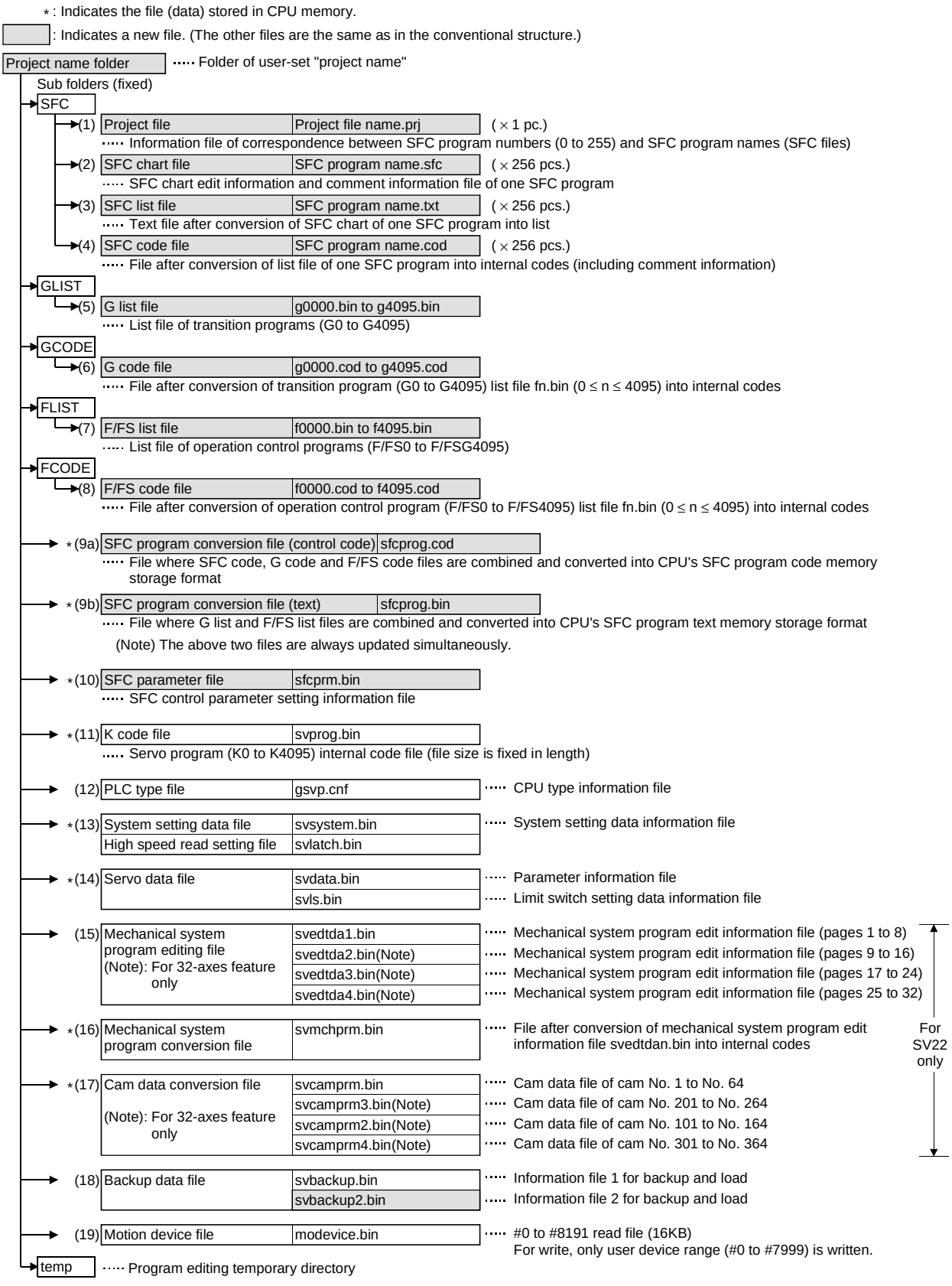
14.1 Projects

User files are managed on a "project" basis.
When you set a "project name", a "project name" folder is created as indicated on the next page, and under that, sub folders (SFC, GLIST, GCODE, FLIST, FCODE) classified by file types are created.
Also, under the SFC sub folders, initial files of the "project file (project name.prj)" and an editing folder (temp) are created.

POINT
(1) Set the "project name" on the project management screen. (2) The "project name" is restricted to 256 characters in length. (3) The "project path name" + "project name" are restricted to 256 characters in length. ((Example) "C:\Usr\.....\project name\")

14.2 User File List

A user file list is indicated below.



15. ERROR LISTS

Eight errors that occurred in the past during SFC control are stored into the "error history devices (#8000 to #8039)" of the motion registers.
The "error codes" in them indicate the following definitions.
The conventional minor errors, major errors, servo errors, servo program errors, mode change errors (SV22 only) and similar errors remain unchanged.

15.1 SFC Program Errors

Table 15.1 SFC Program Start Errors (16000 to 16099)

Error Code	Error Factor		Error Processing	Corrective Action
	Name	Definition		
16000	PLC ready OFF (SFCS)	At a start made by SFCS, PLC ready (M2000) or PCPU ready (M9074) is OFF.	The specified SFC program does not start.	Provide ON of PLC ready (M2000) and PCPU ready (M9074) as start interlocks.
16001	SFC program number error (SFCS)	At an SFC program start made by SFCS, the SFC program number specified is outside the range 0 to 255.		Check the SFC program number, and correct it to a correct sequence program.
16002	No SFC program (SFCS)	At an SFC program start made by SFCS, the specified SFC program does not exist.		Check the SFC program number, and correct it to a correct sequence program, or create an SFC program not yet created.
16003	Double start error (SFCS)	At an SFC program start made by SFCS, the same SFC program is already starting.		Double start should be managed on the user side. Provide the user's starting signal as a start interlock in the sequence program.
16004	PLC ready OFF (ITSP)	ITP was executed with PLC ready (M2000) or PCPU ready (M9074) OFF.	The SFC program's active step to be executed at a "PLC interrupt" is not processed.	Provide ON of PLC ready (M2000) and PCPU ready (M9074) as ITP execution interlocks.
16005	No SFC program	At an SFC program start made by automatic start setting or GSUB, the specified SFC program does not exist.	The specified SFC program does not start. When started by GSUB, the start source SFC program being run also stops.	Check the SFC program number, and correct it to a correct program, or create an SFC program not yet created.
16006	Double start error	At an SFC program start made by automatic start setting or GSUB, the same SFC program is already starting.		Double start should be managed on the user side. Provide the user's starting signal as a transition condition.

Table 15.2 SFC Interpreter Detection Errors (16100 to 16199)

Error Code	Error Factor		Error Processing	Corrective Action
	Name	Definition		
16100	SFC program error (grammatical error)	The code exists but is grammatically erroneous. • Though not within branch-coupling, a label/jump code within selective branch-coupling or a label/jump code within parallel branch-coupling exists.	The corresponding SFC program No. being run stops. For the subroutine called program, the call source program being run also stops.	The SFC program code is corrupted. Turn PLC ready (M2000) OFF and write the SFC program again, or change the battery (A6BAT) if it has reached the end of its life.
16101		• Selective branch destinations are all headed by other than SFT or WAIT transitions.		
16102		• WAITON/WAITOFF is not followed by a motion control step. (However, this is permitted to a pointer (Pn) or jump (Pn).)		
16103		• A parallel branch is followed by an END step without a parallel coupling.		
16104	SFC code error	• An impossible code is used. The internal code is corrupted.		
16105	Jump code error 1	• Internal code (list code) error in jump destination information		
16106	Jump code error 2	• Internal code (label information) error in jump destination information		
16107	Jump code error 3	• Internal code (label number) error in jump destination information		
16108	Jump code error 4	• Internal code (label address) error in jump destination information		
16109	Jump destination error	• The specified pointer does not exist at the jump destination.		
16110	GSUB setting error 1	• Its own program was called/started by GSUB.		GSUB cannot call its own or main program. Correct the SFC program.
16111	GSUB setting error 2	• The main program was called/started by GSUB.		
16112	Parallel branch nesting excess	• Nesting of parallel branches within a parallel branch route exceeded four levels.		The nesting of parallel branch is up to four levels. Subroutine the branch destination processing and correct the program.
16113	Executed task error	• An attempt was made to execute a motion control step K with an event or NMI task.		Motion control steps cannot be set in SFC programs run by the event and NMI tasks. Correct the SFC program or change the "executed task" setting of the SFC parameter to a normal task.
16120	Simultaneously active step count excess	• The number of simultaneously active steps exceeded 256 during execution.		The max. number of simultaneously active steps is 256. Reexamine the SFC program.

Table 15.3 SFC Program Run Errors (16200 to 16299)

Error Code	Error Factor		Error Processing	Corrective Action
	Name	Definition		
16200	No specified program (Kn)	The servo program (Kn) specified at the motion control step does not exist.	The corresponding SFC program being run stops. For the subroutine called program, the call source program being run also stops.	Create the specified servo program.
16201	No specified program (Fn/FSn)	The program (Fn/FSn) specified at the operation control step does not exist.		Create the specified operation control program.
16202	No specified program (Gn)	The program (Gn) specified at the transition does not exist.		Create the specified transition program.
16203	No specified program (SFC)	The SFC program specified at the clear step does not exist.		Correct the specified SFC program name or create the specified SFC program.
16204	No setting of operation expression/conditional expression	The program (Gn) specified at the transition does not have a conditional expression setting.		Always set a conditional expression in the last block of the transition program.
16205	Fn/FSn program code error	Internal code error in the operation program (Fn/FSn)		The SFC program code is corrupted. Turn PLC ready (M2000) OFF and write the SFC program again, or change the battery (A6BAT) if it has reached the end of its life.
16206	Gn program code error	Internal code error in the transition program (Gn)		

Table 15.4 Operation Control/Transition Execution Errors (16300 to 16599)

Error Code	Error Factor		Error Processing	Corrective Action
	Name	Definition		
16301	Event task enable (EI) execution error	Event task enable was executed in other than the normal task.	The block processing in execution is stopped and the next block is executed.	Event task enable may be executed in the normal task only. Correct the program.
16302	Event task disable (DI) execution error	Event task disable was executed in other than the normal task.		Event task disable may be executed in the normal task only. Correct the program.
16303	Block transfer (BMOV) execution error	The cam data of the cam No. specified at (D) or (S) is not yet registered to the motion controller. The resolution of the cam No. specified at (D) or (S) differs from the number of transferred words specified at (n). The PCPU memory address specified at (D) or (S) is outside the SRAM range. (S) to (S)+(n-1) is outside the device range. (D) to (D)+(n-1) is outside the device range. (n) is 0 or a negative number.		Correct the program so that cam data is that of the already registered cam No. Correct the program to match (n) with the cam resolution. Correct the program to specify the PCPU memory address with an even number. Change (n) so that the block transfer range is within the device range. Change (n) to a positive number.
16304	Time to wait (TIME) execution error	The device number which indirectly specifies (S) is illegal. The (S) data is outside the range 0 to 2147483647.		Correct the program so that the device number which indirectly specifies (S) is proper. Correct the program so that the (S) data is within the range 0 to 2147483647.
16308	Speed change request (CHGV) execution error	The specified axis number is outside the range.		Correct the program so that the specified axis number is within the range.
16309	Torque limit value change request (CHGT) execution error			
16316	Assignment (=) execution error	The (S) data is outside the range of the data type of (D). The device number which indirectly specifies (D) is illegal.		Correct the program so that the (S) data is within the range of the data type of (D). Correct the program so that the device number which indirectly specifies (D) is proper.
16320	Operation (/) execution error	The divisor is 0.		Correct the program so that the divisor is other than 0.
16321	Remainder (%) execution error			
16332	Device set (SET) execution error	The device number which indirectly specifies (D) is illegal. (D) is a device which is write-disabled.		Correct the program so that the device number which indirectly specifies (D) is proper. Correct the program to set a write-enabled device at (D).
16333	Device reset (RST) execution error			
16334	Device set (SET=) execution error			
16335	Device reset (RST=) execution error			
16336	Device output (DOUT) execution error			

Table 15.4 Operation Control/Transition Execution Errors (16300 to 16599) (Continued)

Error Code	Error Factor		Error Processing	Corrective Action
	Name	Definition		
16337	Device input (DIN) execution error	The device number which indirectly specifies (D) is illegal.	The block processing in execution is stopped and the next block is executed.	Correct the program so that the device number which indirectly specifies (D) is proper.
16380	Signed 16-bit integral value conversion (SHORT) execution error	The (S) data is outside the signed 16-bit integral value range.		Correct the program so that the (S) data is within the signed 16-bit integral value range.
16381	Unsigned 16-bit integral value conversion (USHORT) execution error	The (S) data is outside the unsigned 16-bit integral value range.		Correct the program so that the (S) data is within the unsigned 16-bit integral value range.
16382	Signed 32-bit integral value conversion (LONG) execution error	The (S) data is outside the signed 32-bit integral value range.		Correct the program so that the (S) data is within the signed 32-bit integral value range.
16383	Unsigned 32-bit integral value conversion (ULONG) execution error	The (S) data is outside the unsigned 32-bit integral value range.		Correct the program so that the (S) data is within the unsigned 32-bit integral value range.
16398	Tangent (TAN) execution error	(S) is 90+(180*n). (n is an integer)		Correct the program so that (S) is not 90+(180*n). (n is an integer)
16399	Arcsine (ASIN) execution error	(S) is outside the range -1.0 to 1.0.		Correct the program so that (S) is within the range -1.0 to 1.0.
16400	Arccosine (ACOS) execution error			
16402	Square root (SQRT) execution error	(S) is a negative number.		Correct the program so that (S) is a positive number.
16403	BCD→BIN conversion (BIN) execution error	Any digit of (S) has a value other than 0 to 9.		Correct the program so that each digit of (S) is 0 to 9.
16404	BIN→BCD conversion (BCD) execution error	The (S) value is outside the range where BIN data can be converted into BCD data.		Correct the program so that the (S) value is within the range.
16405	Natural logarithm (LN) execution error	(S) is 0 or a negative number.		Correct the program so that (S) is a positive number.
16462	Indirectly specified 16-bit motion device (#(n)) read error	The indirectly specified device number is outside the range.		Correct the program so that the indirectly specified device number is proper.
16463	Indirectly specified 32-bit motion device (#(n)L) read error	The indirectly specified device number is outside the range or an odd number.		
16464	Indirectly specified 64-bit motion device (#(n)F) read error			
16465	Indirectly specified 16-bit data register (D(n)) read error	The indirectly specified device number is outside the range.		
16466	Indirectly specified 32-bit data register (D(n)L) read error	The indirectly specified device number is outside the range or an odd number.		
16467	Indirectly specified 64-bit data register (D(n)F) read error			

Table 15.4 Operation Control/Transition Execution Errors (16300 to 16599) (Continued)

Error Code	Error Factor		Error Processing	Corrective Action
	Name	Definition		
16468	Indirectly specified 16-bit link register (W(n)) read error	The indirectly specified device number is outside the range.	The block processing in execution is stopped and the next block is executed.	Correct the program so that the indirectly specified device number is proper.
16469	Indirectly specified 32-bit link register (W(n)L) read error	The indirectly specified device number is outside the range or an odd number.		
16470	Indirectly specified 64-bit link register (W(n):F) read error			
16471	Indirectly specified 16-bit timer present value (T(n)) read error	The indirectly specified device number is outside the range.		
16472	Indirectly specified 16-bit counter present value (C(n)) read error			
16486	Indirectly specified input relay (X(n)) read error			
16487	Indirectly specified output relay (Y(n)) read error			
16488	Indirectly specified internal/latch relay (M(n)/L(n)) read error			
16489	Indirectly specified link relay (B(n)) read error			
16490	Annunciator (F(n)) read error			
16491	Timer contact (TT(n)) read error			
16492	Timer coil (TC(n)) read error			
16493	Counter contact (CT(n)) read error			
16494	Counter coil (CC(n)) read error			
16516	Indirectly specified 16-bit batch input relay (X(n)) read error	The indirectly specified device number is outside the range or is not a multiple of 16.		
16517	Indirectly specified 32-bit batch input relay (X(n)) read error			
16518	Indirectly specified 16-bit batch output relay (Y(n)) read error			
16519	Indirectly specified 32-bit batch output relay (Y(n)) read error			
16520	Indirectly specified 16-bit batch internal/latch relay (M(n)/L(n)) read error			

Table 15.4 Operation Control/Transition Execution Errors (16300 to 16599) (Continued)

Error Code	Error Factor		Error Processing	Corrective Action
	Name	Definition		
16521	Indirectly specified 32-bit batch internal/latch relay (M(n)/L(n)) read error	The indirectly specified device number is outside the range or is not a multiple of 16.	The block processing in execution is stopped and the next block is executed.	Correct the program so that the indirectly specified device number is proper.
16522	Indirectly specified 16-bit batch link relay (B(n)) read error			
16523	Indirectly specified 32-bit batch link relay (B(n)) read error			
16524	Indirectly specified 16-bit batch annunciator (F(n)) read error			
16525	Indirectly specified 32-bit batch annunciator (F(n)) read error			
16526	Indirectly specified 16-bit batch timer contact (TT(n)) read error			
16527	Indirectly specified 32-bit batch timer contact (TT(n)) read error			
16528	Indirectly specified 16-bit batch timer coil (TC(n)) read error			
16529	Indirectly specified 32-bit batch timer coil (TC(n)) read error			
16530	Indirectly specified 16-bit batch counter contact (CT(n)) read error			
16531	Indirectly specified 32-bit batch counter contact (CT(n)) read error			
16532	Indirectly specified 16-bit batch counter coil (CT(n)) read error			
16533	Indirectly specified 32-bit batch counter coil (CC(n)) read error			

15.2 SFC Parameter Errors

Table 15.5 PLC Ready (M2000) OFF→ON Errors (17000 to 17009)

Error Code	Error Factor		Error Processing	Corrective Action
	Name	Definition		
17000	Normal task consecutive transition count error	The normal task's consecutive transition count of the SFC program started by the normal task is outside the range 1 to 30.	The initial value of 3 is used for control.	Turn PLC ready (M2000) OFF, make correction to set the value within the range, and write it to the CPU.
17001	Event task consecutive transition count error	The set number of consecutive transitions of the SFC program started by the event task is outside the range 1 to 10.	The initial value of 1 is used for control.	
17002	NMI task consecutive transition count error	The set number of consecutive transitions of the SFC program started by the NMI task is outside the range 1 to 10.		

Table 15.6 SFC Program Start Errors (Error Code 17010 to 17019)

Error Code	Error Factor		Error Processing	Corrective Action
	Name	Definition		
17010	Executed task setting is illegal	Among the normal, event and NMI tasks, more than one or none of them has been set.	The initial value (normal task) is used for control.	Turn PLC ready (M2000) OFF, make correction, and write a correct value to the CPU.
17011	Executed task setting is illegal (event)	Two or more fixed cycles of the event task have been set.		

16. LIMIT SWITCH OUTPUT FUNCTION

The limit switch output function is designed to output the ON/OFF signal corresponding to the data range of the watch data set per output device.
You can set up to 32 points of output devices.

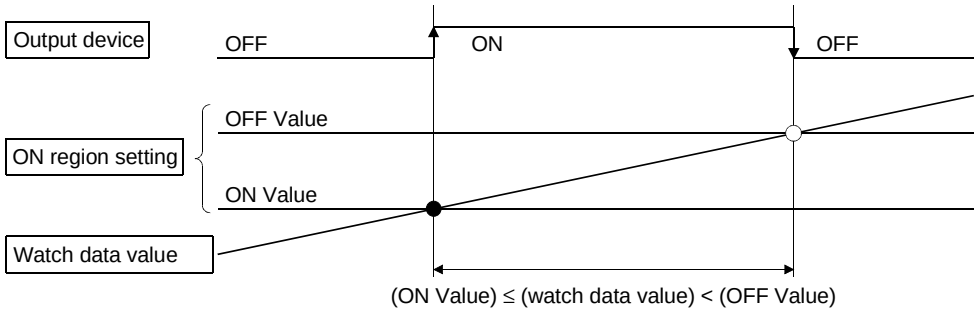
16.1 Operations

(1) The limit switch output function provides an ON output to an output device while the watch data value is in the ON output region set with (ON Value) and (OFF Value).

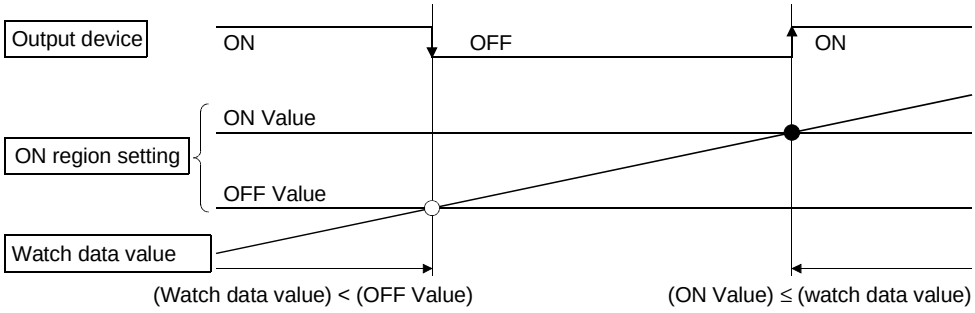
(a) The (ON Value), (OFF Value) and watch data value are handled as signed data.
The ON output region where an ON output is provided to the output device is governed by the magnitude relationship between (ON Value) and (OFF Value) as indicated below.

Relationship between (ON Value) and (OFF Value)	ON Output Region
(ON Value) < (OFF Value)	(ON Value) ≤ (watch data value) < (OFF Value)
(ON Value) > (OFF Value)	(ON Value) ≤ (watch data value) (Watch data value) < (OFF Value)
(ON Value) = (OFF Value)	Output OFF in whole region

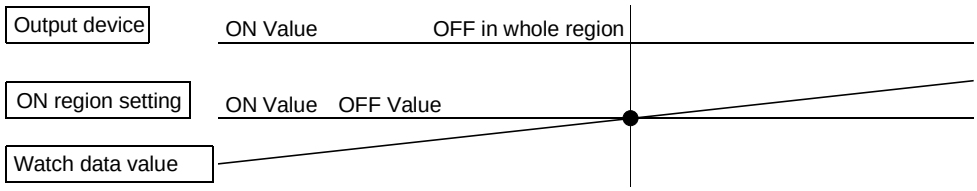
1) If (ON Value) < (OFF Value)



2) If (ON Value) > (OFF Value)

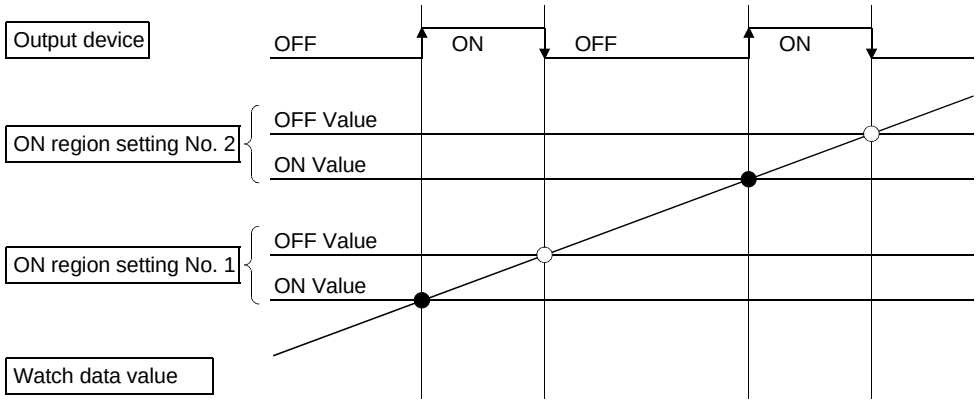


3) If (ON Value) = (OFF Value)



(b) The limit switch outputs are controlled on the basis of each watch data in the PCPU ready status (M9047: ON) after PLC ready (M2000) has turned from OFF to ON.
When the PCPU ready flag (M9047) turns OFF, all points turn OFF.
When (ON Value) and (OFF Value) are specified with word devices, the word device contents are imported to the internal area when PLC ready (M2000) turns from OFF to ON.
Thereafter, in each motion operation cycle, the word device contents are imported to control the limit switch outputs.

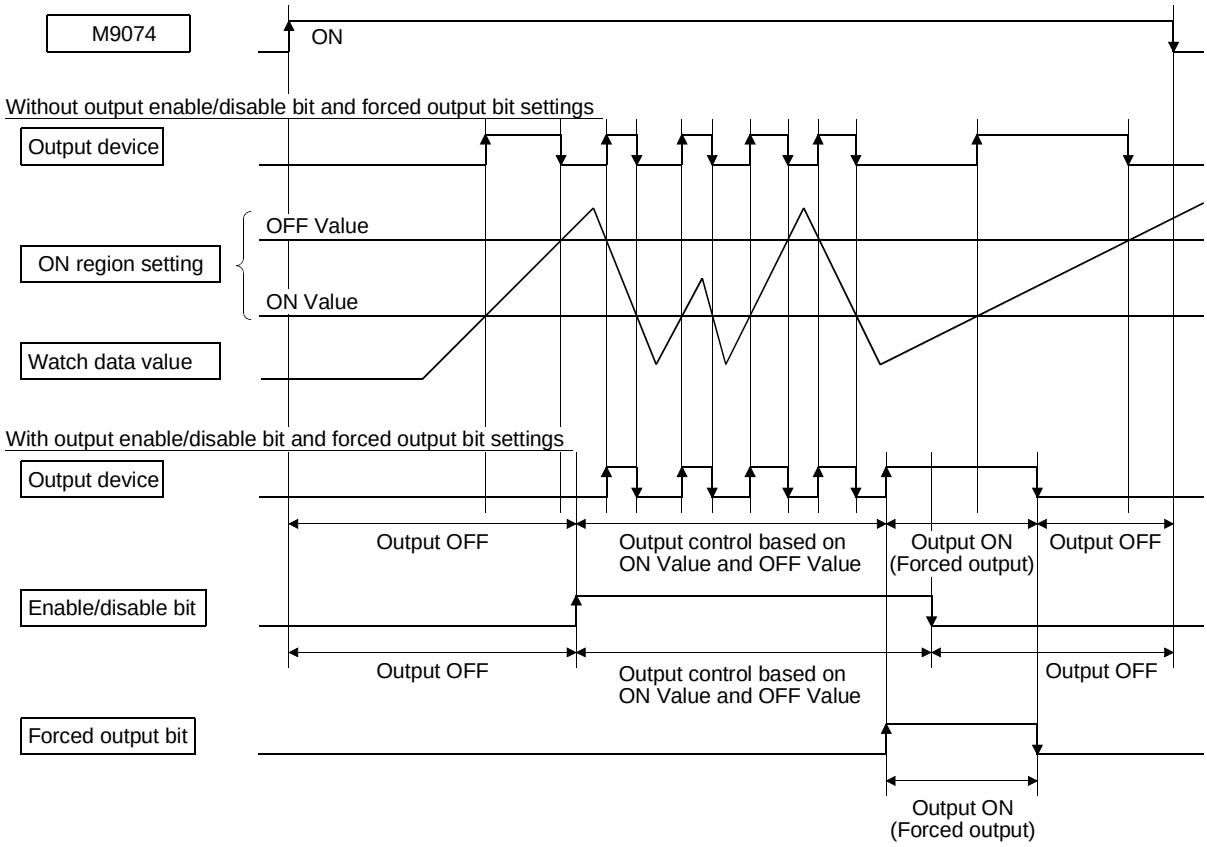
(c) You can set multiple outputs (up to 32 points) to one piece of watch data.
In each setting, the output device may be the same.
If multiple ON region settings have been made to the same output device, the logical add of the output results in the regions is output.



(2) You can set an output enable/disable bit to enable/disable the limit switch outputs point-by-point.
Limit switch output control is exercised when the output enable/disable bit is ON, and the output is OFF when it is OFF.
When there is no setting, the outputs are always enabled.

(3) You can set a forced output bit to forcibly provide (turn ON) the limit switch outputs point-by-point.
The output is ON when the forced output bit is ON.
This setting overrides OFF (disable) of the "output enable/disable bit".
When there is no setting, no forced outputs are always provided.

(4) When multiple pieces of watch data, ON region, output enable/disable bit and forced output bit are set to the same output device, the logical add of the output results of the settings is output.



(5) The conventional limit output function cannot be used.
The following settings and devices are all invalid.

- "Limit output module" in system settings
- "Limit output used/unused" setting in fixed parameters
- "Limit switch output enable" of each axis command device
- Limit switch output disable setting registers

A172SHCPUN	: D1008 to D1011
A173UHCPU (-S1)	: D760 to D775
A273UHCPU-S3	: D760 to D775

- Limit switch output status registers

A172SHCPUN	: D9180 to D9183
A173UHCPU (-S1)	: D776 to D791
A273UHCPU-S3	: D776 to D791

- "Limit switch output used/unused" setting in mechanical system output module parameters

16.2 Limit Output Setting Data

Limit output data are listed below.
You can set up to 32 points of output devices.
(The following items No. 1 to No. 6 are set together as one point.)

No.	Item		Setting Range	Import Cycle	Refresh Cycle	Remarks
1	Output device		Bit device (X, Y, M, L, B)	—	Operation cycle	
2	Watch data		Motion control data/ word device (D, W, #, absolute address) (16-bit integer type/32-bit integer type/ 64-bit floating-point type)	Operation cycle	—	
3	ON region setting	ON Value	Word device (D, W, #)/constant (K, H)	Operation cycle	—	
		OFF Value	Word device (D, W, #)/constant (K, H)	Operation cycle	—	
4	Output enable/disable bit		Bit device (X, Y, M, L, B, F, TT, TC, CT, CC, special M)/ none (default)	Operation cycle	—	ON: Enable OFF: Disable None: Always enable
5	Forced output bit		Bit device (X, Y, M, L, B, F, TT, TC, CT, CC, special M)/ none (default)	Operation cycle	—	None: No forced output always provided (OFF status)

(1) Output device

(a) Set the bit device which outputs the ON/OFF signal in response to the preset watch data.

(b) As the output device, you can use the following devices.

Item	Device Number Setting Range	
	A172SHCPUN	A173UHCPU (-S1) /A273UHCPU-S3
Input relay (Note-1)	X0 to X7FF	X0 to X1FFF
Output relay (Note-2)	Y0 to Y7FF	Y0 to Y1FFF
Internal relay (Note-3)	M0 to M2047	M0 to M8191
Latch relay (Note-3)	L0 to L2047	L0 to L8191
Link relay	B0 to B3FF	B0 to B1FFF

(Note-1): As PX is write-disabled, it cannot be used as the output device.
For X, only the free numbers within the input card non-loading range and outside the link range may be used.
Note that when the A172SHCPUN is used, there will be a read response delay as indicated below.

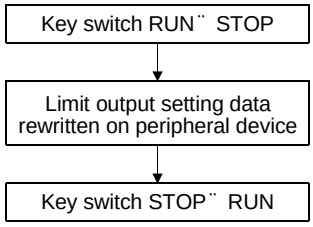
Read CPU	Response Delay
Motion CPU	1 sequence scan
PLC CPU	None

(Nore-2): Note the following points when setting Y as the output device.

- When Y is set, response will be as indicated below.

Y Classification			Response
PBUS	PLC device		Motion operation cycle
	Actual output		
SBUS	Refresh system	PLC device	Motion operation cycle
		Actual output	Motion operation cycle + PLC scan time
	Direct system (A172SHCPUN only)	PLC device	Motion operation cycle
		Actual output	Not provided

- When the STOP→RUN time output mode of the PLC is set to "before operation", performing the following operation to change the output device setting may cause the ON/OFF status of the previously set output device to be output continuously, resulting in an unexpected output status.
  If such operation must be performed, set the STOP→RUN time output mode of the PLC to "after 1 scan run".



(Note-3): As the output devices, M2001 to M2032 cannot be used with the A173UHCPU
(-S1)/A273UHCPU-S3, and M2001 to M2008 with the A172SHCPUN.
While PCPU ready (M9074) is ON, do not perform write from the sequence ladder to the 16-point range which begins with a multiple of 16, including the output device.
Such write operation will not be guaranteed.
The other devices in this range should be used in motion SFC operation control/transition control programs (SET/RST/DOUT).
(Device range example: When the output device is M10, M0 to M15 are in the corresponding range.)

(2) Watch data

(a) This data is used to perform the limit switch output function.
This data is comparison data to output the ON/OFF signal.
The output device is ON/OFF-controlled according to the ON region setting.

(b) As the watch data, motion control data or any word device data can be used.

1) Motion control data

Item	Unit	Data Type	Axis Number Setting Range	
			A172SHCPUN	A173UHCPU (-S1) /A273UHCPU-S3
Feed current value	Position command unit	32-bit integer type	1 to 8	1 to 32
Real current value				
Deviation counter value	PULSE			
Motor current (Command output voltage: ACF)	0.1% (0.01V)	16-bit integer type		
Motor speed	0.1r/min	32-bit integer type		
Cam shaft within-one-revolution current value	PULSE			
Feed current value (temporary)				
After-differential current value (temporary)				
After-differential encoder current value			1	1 to 12
Encoder current value				

2) Word device data

Item	Device Number Setting Range	
	A172SHCPUN	A173UHCPU (-S1) /A273UHCPU-S3
Data register	D0 to D1023	D0 to D8191
Link register	W0 to W3FF	W0 to W1FFF
Motion device	#0 to #8191	#0 to #8191
Absolute address *1	H0 to HFFFFFFFF	H0 to HFFFFFFFF

*1 If the specified absolute address is outside the SRAM range of the motion controller, limit switch output control for the corresponding watch data is not exercised.

3) When you have set any device data, specify the following data type as the data type to be compared.

Data Type	Remarks
16-bit integer type	Specify the absolute address as an even number.
32-bit integer type	Specify the device number as an even number. Specify the absolute address as a multiple of 4.
64-bit floating-point type	Specify the device number as an even number. Specify the absolute address as a multiple of 8.

(3) ON region setting

(a) Set the data range where the output device is turned ON/OFF in response to the watch data.

(b) The following devices can be used as the ON Value and OFF Value of the data range.
The data type of the device/constant to be set is the same as the type of the watch data.

Item	Device Number Setting Range	
	A172SHCPUN	A173UHCPU (-S1) /A273UHCPU-S3
Data register	D0 to D1023	D0 to D8191
Link register	W0 to W3FF	W0 to W1FFF
Motion device	#0 to #8191	#0 to #8191
Constant	Hn/Kn	Hn/Kn

(4) Output enable/disable bit

(a) Set the status of the output enable/disable bit when you want to disable the limit switch outputs during operation.

1) The following control is exercised.

Output Enable/Disable Bit Status		Control
Without setting (always enable)		Limit switch outputs are turned ON/OFF on the basis of the ON region setting (ON Value, OFF Value).
With setting	ON (enable)	
	OFF (disable)	Limit switch outputs are turned OFF.

(b) Usable devices

Item	Device Number Setting Range	
	A172SHCPUN	A173UHCPU (-S1) /A273UHCPU-S3
Input relay	X0 to X7FF	X0 to X1FFF
Output relay	Y0 to Y7FF	Y0 to Y1FFF
Internal relay	M0 to M2047	M0 to M8191
Latch relay	L0 to L2047	L0 to L8191
Link relay	B0 to B3FF	B0 to B1FFF
Annunciator	F0 to F255	F0 to F2047
Timer contact	TT0 to TT255	TT0 to TT2047
Timer coil	TC0 to TC255	TC0 to TC2047
Counter contact	CT0 to CT255	CT0 to CT1023
Counter coil	CC0 to CC255	CC0 to CC1023
Special relay	M9000 to M9255	M9000 to M9255

(5) Forced output bit

(a) Set the "forced output bit" when you want to forcibly provide the limit switch outputs during operation.
This setting overrides OFF (disable) of the above "output enable/disable bit".

1) The following control is exercised.

Forced Output Bit		Control
Without setting		Limit switch outputs are turned ON/OFF on the basis of the "output enable/disable bit" and ON region setting (ON Value, OFF Value).
With setting	ON (enable)	
	OFF (disable)	Limit switch outputs are turned ON.

(b) Usable devices

Item	Device Number Setting Range	
	A172SHCPUN	A173UHCPU (-S1) /A273UHCPU-S3
Input relay	X0 to X7FF	X0 to X1FFF
Output relay	Y0 to Y7FF	Y0 to Y1FFF
Internal relay	M0 to M2047	M0 to M8191
Latch relay	L0 to L2047	L0 to L8191
Link relay	B0 to B3FF	B0 to B1FFF
Annunciator	F0 to F255	F0 to F2047
Timer contact	TT0 to TT255	TT0 to TT2047
Timer coil	TC0 to TC255	TC0 to TC2047
Counter contact	CT0 to CT255	CT0 to CT1023
Counter coil	CC0 to CC255	CC0 to CC1023
Special relay	M9000 to M9255	M9000 to M9255

APPENDICES

APPENDIX 1 PROCESSING TIMES

Appendix 1.1 Operation Control/Transition Instruction Processing Times

(1) Operation instructions

Classification	Symbol	Instruction	Operation Expression	A172SHCPUN/ A173UHCPU(-S1) Unit (μs)	A273UHCPU-S3 Unit (μs)
Binary operation	=	Substitution	#0=#1	16.9	20.9
			D0=D1	24.6	27.9
			#0L=#2L	24.3	30.1
			D0L=D2L	38.9	51.6
			#0F=#4F	26.0	32.0
			D0F=D4F	42.9	55.7
	+	Addition	#0=#1+#2	21.3	25.4
			D0=D1+D2	30.1	33.0
			#0L=#2L+#4L	30.5	37.1
			D0L=D2L+D4L	45.8	60.9
			#0F=#4F+#8F	37.3	41.7
			D0F=D4F+D8F	55.8	68.3
	–	Subtraction	#0=#1–#2	21.2	25.4
			D0=D1–D2	30.1	33.0
			#0L=#2L–#4L	30.6	36.5
			D0L=D2L–D4L	45.3	59.5
			#0F=#4F–#8F	37.6	41.7
			D0F=D4F–D8F	55.6	67.6
	*	Multiplication	#0=#1＊#2	21.1	25.4
			D0=D1＊D2	30.2	33.0
			#0L=#2L＊#4L	30.9	36.5
			D0L=D2L＊D4L	46.8	59.5
			#0F=#4F＊#8F	38.0	41.7
			D0F=D4F＊D8F	56.7	67.6
	/	Division	#0=#1/#2	25.4	32.7
			D0=D1/D2	34.1	41.7
			#0L=#2L/#4L	34.8	44.9
			D0L=D2L/D4L	51.1	69.0
			#0F=#4F/#8F	43.7	44.3
			D0F=D4F/D8F	61.1	70.1
	%	Remainder	#0=#1%#2	24.0	32.5
			D0=D1%D2	32.9	40.1
			#0L=#2L%#4L	34.2	43.6
			D0L=D2L%D4L	51.0	66.7
Bit operation	~	Bit inversion (complement)	#0=~#1	18.8	22.6
			D0=~D1	26.7	31.1
			#0L=~#2L	26.4	32.4
			D0L=~D2L	41.8	54.9
	&	Bit logical AND	#0=#1&#2	20.8	25.2
			D0=D1&D2	28.5	32.7
			#0L=#2L&#4L	30.6	36.2
			D0L=D2L&D4L	46.1	59.2
	|	Bit logical OR	#0=#1|#2	20.8	25.2
			D0=D1|D2	29.1	32.7
			#0L=#2L|#4L	30.0	36.2
			D0L=D2L|D4L	45.5	59.2

Classification	Symbol	Instruction	Operation Expression	A172SHCPUN/ A173UHCPU(-S1) Unit (μs)	A273UHCPU-S3 Unit (μs)
Bit operation	^	Bit exclusive OR	#0=#1^#2	21.2	25.2
			D0=D1^D2	29.4	32.7
			#0L=#2L^#4L	30.3	36.2
			D0L=D2L^D4L	45.7	59.7
	>>	Bit right shift	#0=#1>>#2	21.4	25.6
			D0=D1>>D2	30.2	33.2
			#0L=#2L>>#4L	30.8	37.7
			D0L=D2L>>D4L	46.9	61.8
	<<	Bit left shift	#0=#1<<#2	21.8	25.6
			D0=D1<<D2	30.5	33.2
			#0L=#2L<<#4L	31.7	36.8
			D0L=D2L<<D4L	47.4	59.8
Sign	–	Sign inversion (Complement of 2)	#0=–#1	18.5	22.4
			D0=–D12	26.4	29.4
			#0L=–#2L	26.1	31.8
			D0L=–D2L	41.5	53.3
			#0F=–#4F	27.8	35.5
			D0F=–D4F	44.6	59.2
Standard function	SIN	Sine	#0F=SIN (#4F)	59.6	68.1
			D0F=SIN (D4F)	76.4	92.2
	COS	Cosine	#0F=COS (#4F)	61.2	88.5
			D0F=COS (D4F)	77.9	111.1
	TAN	Tangent	#0F=TAN (#4F)	90.7	98.1
			D0F=TAN (D4F)	108.0	121.0
	ASIN	Arcsine	#0F=ASIN (#4F)	86.2	72.3
			D0F=ASIN (D4F)	103.5	95.9
	ACOS	Arccosine	#0F=ACOS (#4F)	89.5	75.1
			D0F=ACOS (D4F)	107.0	100.3
	ATAN	Arctangent	#0F=ATAN (#4F)	70.9	86.9
			D0F=ATAN (D4F)	88.9	110.5
	SQRT	Square root	#0F=SQRT (#4F)	45.1	70.0
			D0F=SQRT (D4F)	62.2	94.0
	LN	Natural logarithm	#0F=LN (#4F)	55.8	78.3
			D0F=LN (D4F)	73.8	102.0
	EXP	Exponential operation	#0F=EXP (#4F)	47.0	85.3
			D0F=EXP (D4F)	65.3	108.7
	ABS	Absolute value	#0F=ABS (#4F)	30.0	37.6
			D0F=ABS (D4F)	47.0	62.0
	RND	Round-off	#0F=RND (#4F)	41.1	53.5
			D0F=RND (D4F)	59.1	77.9
	FIX	Round-down	#0F=FIX (#4F)	31.3	46.1
			D0F=FIX (D4F)	48.4	70.6
	FUP	Round-up	#0F=FUP (#4F)	30.2	48.2
			D0F=FUP (D4F)	46.7	71.7
	BIN	BCD→BIN conversion	#0=BIN (#1)	21.1	25.0
			DO=BIN (D1)	29.0	33.5
			#0L=BIN (#2L)	29.2	35.5
			D0L=BIN (D2L)	44.9	56.8
	BCD	BIN→BCD conversion	#0=BCD (#1)	31.6	55.4
			DO=BCD (D1)	39.4	63.9
			#0L=BCD (#2L)	45.6	106.1
			D0L=BCD (D2L)	62.5	128.7

Classification	Symbol	Instruction	Operation Expression	A172SHCPUN/ A173UHCPU(-S1) Unit (μs)	A273UHCPU-S3 Unit (μs)
Type conversion	SHORT	Conversion into 16-bit integer type (signed)	#0=SHORT (#2L)	22.8	26.9
			#0=SHORT (#4F)	31.0	47.5
			D0=SHORT (D2L)	33.5	39.9
			D0=SHORT (D4F)	42.2	57.7
	USHORT	Conversion into 16-bit integer type (unsigned)	#0=USHORT (#2L)	23.3	27.0
			#0=USHORT (#4F)	30.4	47.1
			D0=USHORT (D2L)	33.9	39.9
			D0=USHORT (D4F)	41.5	57.2
	LONG	Conversion into 32-bit integer type (signed)	#0L=LONG (#2)	24.4	28.5
			#0L=LONG (#4F)	35.4	50.4
			D0L=LONG (D2)	40.6	50.2
			D0L=LONG (D4F)	52.2	74.5
	ULONG	Conversion into 32-bit integer type (unsigned)	#0L=ULONG (#2)	25.9	28.8
			#0L=ULONG (#4F)	47.2	62.9
			D0L=ULONG (D2)	41.6	50.1
			D0L=ULONG (D4F)	55.9	72.0
	FLOAT	Conversion into 64-bit floating-point type (signed)	#0F=FLOAT (#4)	26.6	32.2
			#0F=FLOAT (#4L)	29.3	35.7
			D0F=FLOAT (D4)	42.6	55.9
			D0F=FLOAT (D4L)	45.7	58.8
	UFLOAT	Conversion into 64-bit floating-point type (unsigned)	#0F=UFLOAT (#4)	26.8	32.3
			#0F=UFLOAT (#4L)	29.3	36.5
			D0F=UFLOAT (D4)	43.4	54.7
			D0F=UFLOAT (D4L)	45.8	60.1
Bit device status	(None)	ON (normally open contact) (when condition enables)	SET M1000 = M0	39.5	39.7
			SET M1000 = X100	39.8	42.0
			SET M1000 = PX0	50.8	44.5
	!	OFF (normally closed contact) (when condition enables)	SET M1000 = !M0	41.3	41.3
			SET M1000 = !X100	42.1	43.6
			SET M1000 = !PX0	47.7	46.1
Bit device control	SET	Device set	SET M1000	28.3	33.2
			SET Y100	29.9	34.5
			SET PY0	28.6	31.9
	RST	Device reset	RST M1000	28.4	32.5
			RST Y100	29.5	33.3
			RST PY0	27.6	31.6
	DOUT	Device output	DOUT M0, #0	29.6	33.9
			DOUT M0, #0L	34.5	37.9
			DOUT Y100, #0	31.7	35.3
			DOUT Y100, #0L	38.3	40.4
			DOUT PY0, #0	36.4	38.8
			DOUT PY0, #0L	45.7	49.1
	DIN	Device input	DIN #0, M0	26.8	30.8
			DIN #0L, M0	32.5	33.9
			DIN #0, X0	16.7	15.3
			DIN #0L, X0	30.5	34.1
			DIN #0, PX0	34.1	36.1
			DIN #0L, PX0	42.2	45.9

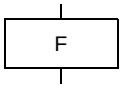
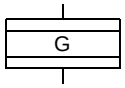
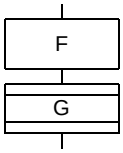
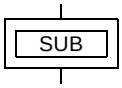
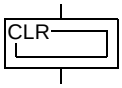
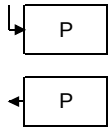
Classification	Symbol	Instruction	Operation Expression	A172SHCPUN/ A173UHCPU(-S1) Unit (μs)	A273UHCPU-S3 Unit (μs)
Logical operation	*	Logical AND	SET M1000 = M0 * M1	52.5	55.4
			SET M1000 = X100 * X101	53.7	58.1
			SET M1000 = PX0 * PX1	47.2	63.4
	+	Logical OR	SET M1000 = M0+M1	53.1	55.4
			SET M1000 = X100+X101	55.5	58.1
			SET M1000 = PX0+PX1	47.4	63.4
Comparison operation	==	Equal to (when condition enables)	SET M1000 = #0==#1	40.1	40.5
			SET M1000 = D0==D1	41.1	42.3
			SET M1000 = #0L==#2L	43.3	45.3
			SET M1000 = D0L==D2L	45.6	51.3
			SET M1000 = #0F==#4F	45.9	49.4
			SET M1000 = D0F==D4F	49.4	57.3
	!=	Not equal to (when condition enables)	SET M1000 = #0!=#1	39.5	40.5
			SET M1000 = D0!=D1	41.9	42.3
			SET M1000 = #0L!=#2L	43.5	45.3
			SET M1000 = D0L!=D2L	45.8	51.3
			SET M1000 = #0F!=#4F	46.2	49.4
			SET M1000 = D0F!=D4F	49.7	57.3
	<	Less than (when condition enables)	SET M1000 = #0<#1	39.9	40.5
			SET M1000 = D0<D1	41.2	42.2
			SET M1000 = #0L<#2L	43.4	45.3
			SET M1000 = D0L<D2L	45.3	51.3
			SET M1000 = #0F<#4F	46.6	49.4
			SET M1000 = D0F<D4F	49.0	57.3
	<=	Less than or equal to (when condition enables)	SET M1000 = #0<=#1	39.8	40.5
			SET M1000 = D0<=D1	41.3	42.3
			SET M1000 = #0L<=#2L	42.3	45.3
			SET M1000 = D0L<=D2L	45.2	51.3
			SET M1000 = #0F<=#4F	44.9	49.4
			SET M1000 = D0F<=D4F	48.9	57.3
	>	More than (when condition enables)	SET M1000 = #0>#1	38.7	40.5
			SET M1000 = D0>D1	41.1	42.3
			SET M1000 = #0L>#2L	42.4	45.3
			SET M1000 = D0L>D2L	44.8	51.4
			SET M1000 = #0F>#4F	44.7	49.9
			SET M1000 = D0F>D4F	49.1	57.3
	=>	More than or equal to (when condition enables)	SET M1000 = #0=>#1	39.7	40.5
			SET M1000 = D0=>D1	41.0	42.3
			SET M1000 = #0L=>#2L	43.5	45.3
			SET M1000 = D0L=>D2L	45.0	51.3
			SET M1000 = #0F=>#4F	45.7	49.4
			SET M1000 = D0F=>D4F	49.5	57.8
Motion-dedicated function	CHGV	Speed change request	CHGV (K1, #0)	18.4	22.9
			CHGV (K1, D0)	21.0	27.6
			CHGV (K1, #0L)	20.0	25.3
			CHGV (K1, D0L)	28.7	37.4
	CHGT	Torque limit value change request	CHGT (K1, #0)	17.9	22.1
			CHGT (K1, D0)	20.5	24.7
			CHGT (K1, #0L)	20.8	26.0
			CHGT (K1, D0L)	30.1	38.1

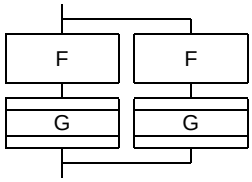
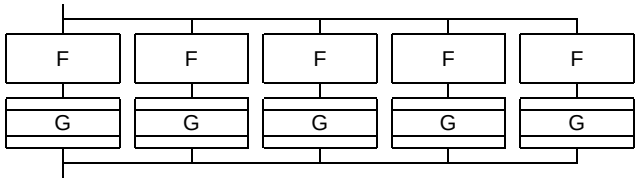
Classification	Symbol	Instruction	Operation Expression	A172SHCPUN/ A173UHCPU(-S1) Unit (μs)	A273UHCPU-S3 Unit (μs)
Others	EI	Event task enable	EI	5.3	7.6
	DI	Event task disable	DI	5.5	6.8
	NOP	No operation	NOP	1.5	1.7
	BMOV	Block move	BMOV #0, #100, K10	25.5	29.4
			BMOV D0, D100, K10	33.8	38.4
			BMOV #0, #100, K100	71.6	85.4
			BMOV D0, D100, K100	138.1	168.0
			BMOV N1, #0, K512	54.3	64.5
			BMOV N1, D0, K512	53.5	64.7
	TIME	Time to wait		—	—

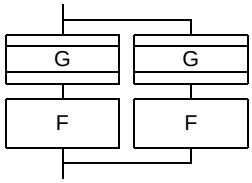
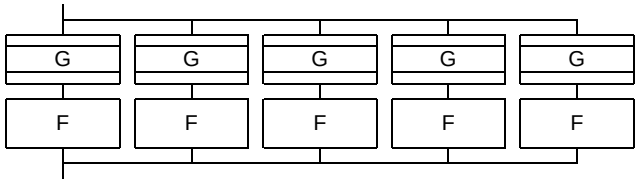
(2) Transition conditional expressions

Classification	Symbol	Instruction	Operation Expression	A172SHCPUN/ A173UHCPU(-S1) Single Operation Expression Processing Time (μs)	A273UHCPU-S3 Single Operation Expression Processing Time (μs)
Bit device status	(None)	ON (normally open contact) (when condition enables)	M0	24.7	20.7
			X100	22.3	17.7
			PX0	25.9	17.5
	!	OFF (normally closed contact) (when condition enables)	!M0	26.4	22.4
			!X100	24.1	19.3
			!PX0	27.6	19.5
Logical operation	*	Logical AND	M0*M1	31.8	26.7
			X100*X101	29.9	23.8
			PX0*PX1	36.2	23.4
	+	Logical OR	M0+M1	31.8	27.1
			X100+X101	30.0	24.0
			PX0+PX1	36.2	24.0
Comparison operation	==	Equal to (when condition enables)	#0==#1	9.5	8.8
			D800==D801	19.2	16.6
			#0L==#2L	25.0	19.1
			D800L==D802L	38.4	29.3
			#0F==#4F	29.7	22.2
			D800F==D804F	44.3	33.4
	!=	Not equal to (when condition enables)	#0!=#1	9.5	8.8
			D800!=D801	19.2	16.2
			#0L!=#2L	25.0	19.3
			D800L!=D802L	38.4	29.6
			#0F!=#4F	29.9	22.0
			D800F!=D804F	44.5	33.2
	<	Less than (when condition enables)	#0<#1	9.5	8.8
			D800<D801	19.2	16.4
			#0L<#2L	25.0	19.1
			D800L<D802L	38.4	29.3
			#0F<#4F	29.9	22.0
			D800F<D804F	44.5	33.1
	<=	Less than or equal to (when condition enables)	#0<=#1	9.5	8.8
			D800<=D801	19.2	16.5
			#0L<=#2L	25.0	19.3
			D800L<=D802L	38.4	29.3
			#0F<=#4F	29.9	22.2
			D800F<=D804F	44.5	33.3
	>	More than (when condition enables)	#0>#1	9.5	8.8
			D800>D801	19.2	16.1
			#0L>#2L	25.0	19.1
			D800L>D802L	38.6	29.1
			#0F>#4F	29.9	22.1
			D800F>D804F	44.5	33.3
	>=	More than or equal to (when condition enables)	#0>=#1	9.5	8.8
			D800>=D801	19.2	16.1
			#0L>=#2L	25.0	19.3
			D800L>=D802L	38.4	29.2
			#0F>=#4F	29.9	22.2
			D800F>=D804F	44.5	33.0

(3) Processing time taken when F and G are combined (program described in F/G is NOP)

		F Alone	G Alone	F+G	GSUB	CLR	JMP/Coupling
					 (Note)	 (Note)	
A172SHCPUN A173UHCPU(-S1)	μs	48	40	51	103	48	37
A273UHCPU-S3	μs	43	35	48	109	48	37

		Parallel Branch (2 Pcs.)		Parallel Branch (5 Pcs.)	
					
		At branch	At coupling	At branch	At coupling
A172SHCPUN A173UHCPU(-S1)	μs	111	75	232	140
A273UHCPU-S3	μs	118	80	252	152

		Selective Branch (2 Pcs.)	Selective Branch (5 Pcs.)
			
A172SHCPUN A173UHCPU(-S1)	μs	142	185
A273UHCPU-S3	μs	163	227

(Note) Varies greatly with the started/cleared program.

POINT
Long processing time may cause a PCPU WDT error or servo fault. Especially for SFC programs run by event/NMI tasks, take care so that the processing time will not be too long.

Appendix 1.2 Motion Operation Cycles (msec)

CPU	A172SHCPUN			A173UHCPU(-S1)						A273UHCPU-S3					
OS type	SV13	SV22		SV13			SV22			SV13			SV22		
Set axis count	1 to 8	1 to 6	7 to 8	1 to 17	18 to 29	30 to 32	1 to 10	11 to 18	19 to 32	1 to 10	11 to 21	22 to 32	1 to 6	7 to 13	14 to 32
Operation cycle	3.5ms	3.5ms	7.1ms	3.5ms	7.1ms	14.2ms	3.5ms	7.1ms	14.2ms	3.5ms	7.1ms	14.2ms	3.5ms	7.1ms	14.2ms

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