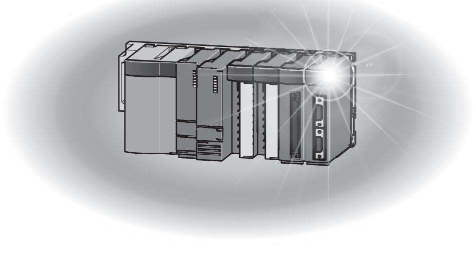
Motion Controller

MELSEC Q series

Q173HCPU/Q172HCPU
Motion Controller (SV13/SV22)
Programming Manual (Motion SFC)

-Q172HCPU
-Q173HCPU



● SAFETY PRECAUTIONS ●

(Read these precautions before using.)

When using this equipment, thoroughly read this manual and the associated manuals introduced in this manual. Also pay careful attention to safety and handle the module properly.
These precautions apply only to this equipment. Refer to the Q173HCPU/Q172HCPU Users manual for a description of the Motion controller safety precautions.
These SAFETY PRECAUTIONS classify the safety precautions into two categories: "DANGER" and "CAUTION".

◇! DANGER	Indicates that incorrect handling may cause hazardous conditions, resulting in death or severe injury.
⚠ CAUTION	Indicates that incorrect handling may cause hazardous conditions, resulting in medium or slight personal injury or physical damage.

Depending on circumstances, procedures indicated by ⚠ CAUTION may also be linked to serious results.
In any case, it is important to follow the directions for usage.

Store this manual in a safe place so that you can take it out and read it whenever necessary. Always forward it to the end user.

For Safe Operations

1. Prevention of electric shocks

◇! DANGER

- Never open the front case or terminal covers while the power is ON or the unit is running, as this may lead to electric shocks.
- Never run the unit with the front case or terminal cover removed. The high voltage terminal and charged sections will be exposed and may lead to electric shocks.
- Never open the front case or terminal cover at times other than wiring work or periodic inspections even if the power is OFF. The insides of the Motion controller and servo amplifier are charged and may lead to electric shocks.
- When performing wiring work or inspections, turn the power OFF, wait at least ten minutes, and then check the voltage with a tester, etc.. Failing to do so may lead to electric shocks.
- Be sure to ground the Motion controller, servo amplifier and servomotor. (Ground resistance : 100 Ω or less) Do not ground commonly with other devices.
- The wiring work and inspections must be done by a qualified technician.
- Wire the units after installing the Motion controller, servo amplifier and servomotor. Failing to do so may lead to electric shocks or damage.
- Never operate the switches with wet hands, as this may lead to electric shocks.
- Do not damage, apply excessive stress, place heavy things on or sandwich the cables, as this may lead to electric shocks.
- Do not touch the Motion controller, servo amplifier or servomotor terminal blocks while the power is ON, as this may lead to electric shocks.
- Do not touch the built-in power supply, built-in grounding or signal wires of the Motion controller and servo amplifier, as this may lead to electric shocks.

2. For fire prevention

⚠ CAUTION

- Install the Motion controller, servo amplifier, servomotor and regenerative resistor on inflammable material. Direct installation on flammable material or near flammable material may lead to fire.
- If a fault occurs in the Motion controller or servo amplifier, shut the power OFF at the servo amplifier's power source. If a large current continues to flow, fire may occur.
- When using a regenerative resistor, shut the power OFF with an error signal. The regenerative resistor may abnormally overheat due to a fault in the regenerative transistor, etc., and may lead to fire.
- Always take heat measures such as flame proofing for the inside of the control panel where the servo amplifier or regenerative resistor is installed and for the wires used. Failing to do so may lead to fire.

3. For injury prevention

⚠ CAUTION

- Do not apply a voltage other than that specified in the instruction manual on any terminal. Doing so may lead to destruction or damage.
- Do not mistake the terminal connections, as this may lead to destruction or damage.
- Do not mistake the polarity (+ / -), as this may lead to destruction or damage.
- Do not touch the servo amplifier's heat radiating fins, regenerative resistor and servomotor, etc., while the power is ON and for a short time after the power is turned OFF. In this timing, these parts become very hot and may lead to burns.
- Always turn the power OFF before touching the servomotor shaft or coupled machines, as these parts may lead to injuries.
- Do not go near the machine during test operations or during operations such as teaching. Doing so may lead to injuries.

4. Various precautions

Strictly observe the following precautions.

Mistaken handling of the unit may lead to faults, injuries or electric shocks.

(1) System structure

⚠ CAUTION

- Always install a leakage breaker on the Motion controller and servo amplifier power source.
- If installation of an electromagnetic contactor for power shut off during an error, etc., is specified in the instruction manual for the servo amplifier, etc., always install the electromagnetic contactor.
- Install the emergency stop circuit externally so that the operation can be stopped immediately and the power shut off.
- Use the Motion controller, servo amplifier, servomotor and regenerative resistor with the combinations listed in the instruction manual. Other combinations may lead to fire or faults.
- If safety standards (ex., robot safety rules, etc.,) apply to the system using the Motion controller, servo amplifier and servomotor, make sure that the safety standards are satisfied.
- Construct a safety circuit externally of the Motion controller or servo amplifier if the abnormal operation of the Motion controller or servo amplifier differ from the safety directive operation in the system.
- In systems where coasting of the servomotor will be a problem during the forced stop, emergency stop, servo OFF or power supply OFF, use dynamic brakes.
- Make sure that the system considers the coasting amount even when using dynamic brakes.
- In systems where perpendicular shaft dropping may be a problem during the forced stop, emergency stop, servo OFF or power supply OFF, use both dynamic brakes and electromagnetic brakes.
- The dynamic brakes must be used only on errors that cause the forced stop, emergency stop, or servo OFF. These brakes must not be used for normal braking.

⚠ CAUTION

- The brakes (electromagnetic brakes) assembled into the servomotor are for holding applications, and must not be used for normal braking.
- The system must have a mechanical allowance so that the machine itself can stop even if the stroke limits switch is passed through at the max. speed.
- Use wires and cables that have a wire diameter, heat resistance and bending resistance compatible with the system.
- Use wires and cables within the length of the range described in the instruction manual.
- The ratings and characteristics of the parts (other than Motion controller, servo amplifier and servomotor) used in a system must be compatible with the Motion controller, servo amplifier and servomotor.
- Install a cover on the shaft so that the rotary parts of the servomotor are not touched during operation.
- There may be some cases where holding by the electromagnetic brakes is not possible due to the life or mechanical structure (when the ball screw and servomotor are connected with a timing belt, etc.). Install a stopping device to ensure safety on the machine side.

(2) Parameter settings and programming

⚠ CAUTION

- Set the parameter values to those that are compatible with the Motion controller, servo amplifier, servomotor and regenerative resistor model and the system application. The protective functions may not function if the settings are incorrect.
- The regenerative resistor model and capacity parameters must be set to values that conform to the operation mode, servo amplifier and servo power supply module. The protective functions may not function if the settings are incorrect.
- Set the mechanical brake output and dynamic brake output validity parameters to values that are compatible with the system application. The protective functions may not function if the settings are incorrect.
- Set the stroke limit input validity parameter to a value that is compatible with the system application. The protective functions may not function if the setting is incorrect.
- Set the servomotor encoder type (increment, absolute position type, etc.) parameter to a value that is compatible with the system application. The protective functions may not function if the setting is incorrect.
- Set the servomotor capacity and type (standard, low-inertia, flat, etc.) parameter to values that are compatible with the system application. The protective functions may not function if the settings are incorrect.
- Set the servo amplifier capacity and type parameters to values that are compatible with the system application. The protective functions may not function if the settings are incorrect.
- Use the program commands for the program with the conditions specified in the instruction manual.

⚠ CAUTION

- Set the sequence function program capacity setting, device capacity, latch validity range, I/O assignment setting, and validity of continuous operation during error detection to values that are compatible with the system application. The protective functions may not function if the settings are incorrect.
- Some devices used in the program have fixed applications, so use these with the conditions specified in the instruction manual.
- The input devices and data registers assigned to the link will hold the data previous to when communication is terminated by an error, etc. Thus, an error correspondence interlock program specified in the instruction manual must be used.
- Use the interlock program specified in the special function module's instruction manual for the program corresponding to the special function module.

(3) Transportation and installation

⚠ CAUTION

- Transport the product with the correct method according to the mass.
- Use the servomotor suspension bolts only for the transportation of the servomotor. Do not transport the servomotor with machine installed on it.
- Do not stack products past the limit.
- When transporting the Motion controller or servo amplifier, never hold the connected wires or cables.
- When transporting the servomotor, never hold the cables, shaft or detector.
- When transporting the Motion controller or servo amplifier, never hold the front case as it may fall off.
- When transporting, installing or removing the Motion controller or servo amplifier, never hold the edges.
- Install the unit according to the instruction manual in a place where the mass can be withstood.
- Do not get on or place heavy objects on the product.
- Always observe the installation direction.
- Keep the designated clearance between the Motion controller or servo amplifier and control panel inner surface or the Motion controller and servo amplifier, Motion controller or servo amplifier and other devices.
- Do not install or operate Motion controller, servo amplifiers or servomotors that are damaged or that have missing parts.
- Do not block the intake/outtake ports of the servomotor with cooling fan.
- Do not allow conductive matter such as screw or cutting chips or combustible matter such as oil enter the Motion controller, servo amplifier or servomotor.
- The Motion controller, servo amplifier and servomotor are precision machines, so do not drop or apply strong impacts on them.

⚠ CAUTION

- Securely fix the Motion controller and servo amplifier to the machine according to the instruction manual. If the fixing is insufficient, these may come off during operation.
- Always install the servomotor with reduction gears in the designated direction. Failing to do so may lead to oil leaks.
- Store and use the unit in the following environmental conditions.

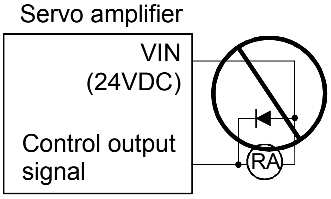
Environment	Conditions	
	Motion controller/Servo amplifier	Servomotor
Ambient temperature	According to each instruction manual.	0°C to +40°C (With no freezing) (32°F to +104°F)
Ambient humidity	According to each instruction manual.	80% RH or less (With no dew condensation)
Storage temperature	According to each instruction manual.	-20°C to +65°C (-4°F to +149°F)
Atmosphere	Indoors (where not subject to direct sunlight). No corrosive gases, flammable gases, oil mist or dust must exist	
Altitude	1000m (3280.84ft.) or less above sea level	
Vibration	According to each instruction manual	

- When coupling with the synchronization encoder or servomotor shaft end, do not apply impact such as by hitting with a hammer. Doing so may lead to detector damage.
- Do not apply a load larger than the tolerable load onto the servomotor shaft. Doing so may lead to shaft breakage.
- When not using the module for a long time, disconnect the power line from the Motion controller or servo amplifier.
- Place the Motion controller and servo amplifier in static electricity preventing vinyl bags and store.
- When storing for a long time, please contact with our sales representative.

(4) Wiring

⚠ CAUTION

- Correctly and securely wire the wires. Reconfirm the connections for mistakes and the terminal screws for tightness after wiring. Failing to do so may lead to run away of the servomotor.
- After wiring, install the protective covers such as the terminal covers to the original positions.
- Do not install a phase advancing capacitor, surge absorber or radio noise filter (option FR-BIF) on the output side of the servo amplifier.
- Correctly connect the output side (terminals U, V, W). Incorrect connections will lead the servomotor to operate abnormally.
- Do not connect a commercial power supply to the servomotor, as this may lead to trouble.
- Do not mistake the direction of the surge absorbing diode installed on the DC relay for the control signal output of brake signals, etc. Incorrect installation may lead to signals not being output when trouble occurs or the protective functions not functioning.



- Do not connect or disconnect the connection cables between each unit, the encoder cable or PLC expansion cable while the power is ON.
- Securely tighten the cable connector fixing screws and fixing mechanisms. Insufficient fixing may lead to the cables combing off during operation.
- Do not bundle the power line or cables.

(5) Trial operation and adjustment

⚠ CAUTION

- Confirm and adjust the program and each parameter before operation. Unpredictable movements may occur depending on the machine.
- Extreme adjustments and changes may lead to unstable operation, so never make them.
- When using the absolute position system function, on starting up, and when the Motion controller or absolute value motor has been replaced, always perform a home position return.

(6) Usage methods

⚠ CAUTION

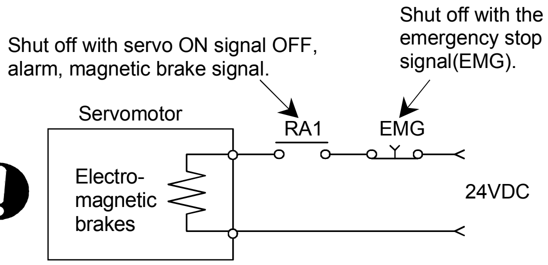
- ● Immediately turn OFF the power if smoke, abnormal sounds or odors are emitted from the Motion controller, servo amplifier or servomotor.
- ● Always execute a test operation before starting actual operations after the program or parameters have been changed or after maintenance and inspection.
- ● The units must be disassembled and repaired by a qualified technician.
- ● Do not make any modifications to the unit.
- ● Keep the effect or electromagnetic obstacles to a minimum by installing a noise filter or by using wire shields, etc. Electromagnetic obstacles may affect the electronic devices used near the Motion controller or servo amplifier.
- ● When using the CE Mark-compliant equipment, refer to the "EMC Installation Guidelines" (data number IB(NA)-67339) for the Motion controllers and refer to the corresponding EMC guideline information for the servo amplifiers, inverters and other equipment.
- ● Use the units with the following conditions.

Item	Conditions					
	Q61P-A1	Q61P-A2	Q61P	Q62P	Q63P	Q64P
Input power	100 to 120VAC $^{+10\%}_{-15\%}$ (85 to 132VAC)	200 to 240VAC $^{+10\%}_{-15\%}$ (170 to 264VAC)	100 to 240VAC $^{+10\%}_{-15\%}$ (85 to 264VAC)		24VDC $^{+30\%}_{-35\%}$ (15.6 to 31.2VDC)	100 to 120VAC $^{+10\%}_{-15\%}$ / 200 to 240VAC $^{+10\%}_{-15\%}$ (85 to 132VAC/ 170 to 264VAC)
Input frequency	50/60Hz ±5%					
Tolerable momentary power failure	20ms or less					

(7) Corrective actions for errors

⚠ CAUTION

- ● If an error occurs in the self diagnosis of the Motion controller or servo amplifier, confirm the check details according to the instruction manual, and restore the operation.
- ● If a dangerous state is predicted in case of a power failure or product failure, use a servomotor with electromagnetic brakes or install a brake mechanism externally.
- ● Use a double circuit construction so that the electromagnetic brake operation circuit can be operated by emergency stop signals set externally.



⚠ CAUTION

- If an error occurs, remove the cause, secure the safety and then resume operation after alarm release.
- The unit may suddenly resume operation after a power failure is restored, so do not go near the machine. (Design the machine so that personal safety can be ensured even if the machine restarts suddenly.)

(8) Maintenance, inspection and part replacement

⚠ CAUTION

- Perform the daily and periodic inspections according to the instruction manual.
- Perform maintenance and inspection after backing up the program and parameters for the Motion controller and servo amplifier.
- Do not place fingers or hands in the clearance when opening or closing any opening.
- Periodically replace consumable parts such as batteries according to the instruction manual.
- Do not touch the lead sections such as ICs or the connector contacts.
- Do not place the Motion controller or servo amplifier on metal that may cause a power leakage or wood, plastic or vinyl that may cause static electricity buildup.
- Do not perform a megger test (insulation resistance measurement) during inspection.
- When replacing the Motion controller or servo amplifier, always set the new module settings correctly.
- When the Motion controller or absolute value motor has been replaced, carry out a home position return operation using one of the following methods, otherwise position displacement could occur.
    1) After writing the servo data to the Motion controller using programming software, switch on the power again, then perform a home position return operation.
    2) Using the backup function of the programming software, load the data backed up before replacement.
- After maintenance and inspections are completed, confirm that the position detection of the absolute position detector function is correct.
- Do not short circuit, charge, overheat, incinerate or disassemble the batteries.
- The electrolytic capacitor will generate gas during a fault, so do not place your face near the Motion controller or servo amplifier.
- The electrolytic capacitor and fan will deteriorate. Periodically replace these to prevent secondary damage from faults. Replacements can be made by our sales representative.

(9) About processing of waste

When you discard Motion controller, servo amplifier, a battery (primary battery) and other option articles, please follow the law of each country (area).

⚠ CAUTION

- ● This product is not designed or manufactured to be used in equipment or systems in situations that can affect or endanger human life.
- ● When considering this product for operation in special applications such as machinery or systems used in passenger transportation, medical, aerospace, atomic power, electric power, or submarine repeating applications, please contact your nearest Mitsubishi sales representative.
- ● Although this product was manufactured under conditions of strict quality control, you are strongly advised to install safety devices to forestall serious accidents when it is used in facilities where a breakdown in the product is likely to cause a serious accident.

(10) General cautions

⚠ CAUTION

- ● All drawings provided in the instruction manual show the state with the covers and safety partitions removed to explain detailed sections. When operating the product, always return the covers and partitions to the designated positions, and operate according to the instruction manual.

REVISIONS

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

Print Date	∗ Manual Number	Revision
Jun., 2005	IB(NA)-0300112-A	First edition
Sep., 2006	IB(NA)-0300112-B	[Additional model] Q61P, MR-J3-□B-RJ006 [Additional function] Control loop changing command, Control loop monitor status, Bit device setting by Motion SFC instruction [Additional correction/partial correction] About Manuals, Restriction by the version, Device lists, User file list, etc.

Japanese Manual Number IB(NA)-0300092

INTRODUCTION

Thank you for choosing the Q173HCPU/Q172HCPU Motion Controller.
Please read this manual carefully so that equipment is used to its optimum.

CONTENTS

5. OPERATION CONTROL PROGRAMS 5- 1 to 5-94

11. ERROR CODE LISTS 11- 1 to 11-12

APPENDICES APP- 1 to APP-31

About Manuals

The following manuals are related to this product.
Referring to this list, please request the necessary manuals.

Related Manuals

(1) Motion controller

Manual Name	Manual Number (Model Code)
Q173HCPU/Q172HCPU Motion controller User's Manual This manual explains specifications of the Motion CPU modules, Q172LX Servo external signal interface module, Q172EX Serial absolute synchronous encoder interface module, Q173PX Manual pulse generator interface module, Teaching units, Power supply modules, Servo amplifiers, SSCNETⅢ cables, synchronous encoder cables and others. (Optional)	IB-0300110 (1XB910)
Q173HCPU/Q172HCPU Motion controller Programming Manual (COMMON) This manual explains the Multiple CPU system configuration, performance specifications, common parameters, auxiliary/applied functions and others. (Optional)	IB-0300111 (1XB911)
Q173HCPU/Q172HCPU Motion controller (SV13/SV22) Programming Manual (REAL MODE) This manual explains the servo parameters, positioning instructions, device list, error list and others. (Optional)	IB-0300113 (1XB913)
Q173HCPU/Q172HCPU Motion controller (SV22) Programming Manual (VIRTUAL MODE) This manual describes the dedicated instructions use to the synchronous control by virtual main shaft, mechanical system program create mechanical module. This manual explains the servo parameters, positioning instructions, device list, error list and others. (Optional)	IB-0300114 (1XB914)
Q173HCPU/Q172HCPU Motion controller (SV43) Programming Manual This manual describes the dedicated instructions to execute the positioning control by Motion program of EIA language (G-code). This manual explains the servo parameters, positioning instructions, device list, error list and others. (Optional)	IB-0300115 (1XB915)

(2) PLC

Manual Name	Manual Number (Model Code)
QCPU User's Manual (Hardware Design, Maintenance and Inspection) This manual explains the specifications of the QCPU modules, power supply modules, base modules, extension cables, memory card battery and others. (Optional)	SH-080483ENG (13JR73)
QCPU User's Manual (Function Explanation, Program Fundamentals) This manual explains the functions, programming methods and devices and others to create programs with the QCPU. (Optional)	SH-080484ENG (13JR74)
QCPU User's Manual (Multiple CPU System) This manual explains the functions, programming methods and cautions and others to construct the Multiple CPU system with the QCPU. (Optional)	SH-080485ENG (13JR75)
QCPU (Q Mode)/QnACPU Programming Manual (Common Instructions) This manual explains how to use the sequence instructions, basic instructions, application instructions and micro computer program. (Optional)	SH-080039 (13JF58)
QCPU (Q Mode)/QnACPU Programming Manual (PID Control Instructions) This manual explains the dedicated instructions used to exercise PID control. (Optional)	SH-080040 (13JF59)
QCPU (Q Mode)/QnACPU Programming Manual (SFC) This manual explains the system configuration, performance specifications, functions, programming, debugging, error codes and others of MELSAP3. (Optional)	SH-080041 (13JF60)
I/O Module Type Building Block User's Manual This manual explains the specifications of the I/O modules, connector, connector/terminal block conversion modules and others. (Optional)	SH-080042 (13JL99)

(3) Servo amplifier

Manual Name	Manual Number (Model Code)
MR-J3-☐B Servo amplifier Instruction Manual This manual explains the I/O signals, parts names, parameters, start-up procedure and others for MR-J3-☐B Servo amplifier. (Optional)	SH-030051 (1CW202)
Fully Closed Loop Control MR-J3-☐B-RJ006 Servo amplifier Instruction Manual This manual explains the I/O signals, parts names, parameters, start-up procedure and others for Fully Closed Loop Control MR-J3-☐B-RJ006 Servo amplifier. (Optional)	SH-030056 (1CW304)

1. OVERVIEW

1.1 Overview

This programming manual describes the Motion SFC program and Multiple CPU system of the operating system software packages "SW6RN-SV13Q☐", "SW6RN-SV22Q☐" for Motion CPU module(Q173HCPU/Q172HCPU).
In this manual, the following abbreviations are used.

Generic term/Abbreviation	Description
Q173HCPU/Q172HCPU or Motion CPU (module)	Q173HCPU/Q172HCPU/Q173HCPU-T/Q172HCPU-T Motion CPU module
Q172LX/Q172EX/Q173PX or Motion module	Q172LX Servo external signals interface module/ Q172EX-S2/-S3 Serial absolute synchronous encoder interface module(Note-1)/ Q173PX(-S1) Manual pulse generator interface module
MR-J3-☐B	Servo amplifier model MR-J3-☐B
AMP or Servo amplifier	General name for "Servo amplifier model MR-J3-☐B"
QCPU, PLC CPU or PLC CPU module	Qn(H)CPU
Multiple CPU system or Motion system	Abbreviation for "Multiple PLC system of the Q series"
CPUn	Abbreviation for "CPU No.n (n= 1 to 4) of the CPU module for the Multiple CPU system"
Programming software package	General name for "MT Developer" and "GX Developer"
Operating system software	General name for "SW☐RN-SV☐Q☐"
SV13	Operating system software for conveyor assembly use (Motion SFC) : SW6RN-SV13Q☐
SV22	Operating system software for automatic machinery use (Motion SFC) : SW6RN-SV22Q☐
MT Developer	Abbreviation for Integrated start-up support software package "MT Developer (Version 00K or later)"
GX Developer	Abbreviation for MELSEC PLC programming software package "GX Developer (Version 6 or later)"
Manual pulse generator or MR-HDP01	Abbreviation for "Manual pulse generator (MR-HDP01)"
Serial absolute synchronous encoder or Q170ENC	Abbreviation for "Serial absolute synchronous encoder (Q170ENC)"
SSCNETⅢ(Note-2)	High speed synchronous network between Motion controller and servo amplifier
SSCNET(Note-2)	High speed serial communication between Motion controller and servo amplifier
Absolute position system	General name for "system using the servomotor and servo amplifier for absolute position"
Battery holder unit	Battery holder unit (Q170HBATC)
External battery	General name for "Q170HBATC" and "Q6BAT"
A☐0BD-PCF	A10BD-PCF/A30BD-PCF SSC I/F board
SSC I/F communication cable	Abbreviation for "Cable for SSC I/F board/card"
Teaching Unit or A31TU-D3K☐/A31TU-DNK☐	A31TU-D3☐/A31TU-DN☐ Teaching unit(Note-3)

Generic term/Abbreviation	Description
Intelligent function module	Abbreviation for "MELSECNET/H module/Ethernet module/CC-Link module/ Serial communication module"

(Note-1) : Q172EX can be used in SV22.

(Note-2) : SSCNET: Servo System Controller NETwork

(Note-3) : Teaching unit can be used in SV13.

REMARK

For information about the each module, design method for program and parameter, refer to the following manuals relevant to each module.

Item		Reference Manual
Motion CPU module/Motion unit		Q173HCPU/Q172HCPU User's Manual
PLC CPU, peripheral devices for PLC program design, I/O modules and intelligent function module		Manual relevant to each module
Operation method for MT Developer		Help of each software
SV13/SV22	• Multiple CPU system configuration • Performance specification • Design method for common parameter • Auxiliary and applied functions (common)	Q173HCPU/Q172HCPU Motion controller Programming Manual (COMMON)
	• Design method for positioning control program in the real mode • Design method for positioning control parameter	Q173HCPU/Q172HCPU Motion controller (SV13/SV22) Programming Manual (REAL MODE)
SV22 (Virtual mode)	• Design method for mechanical system program	Q173HCPU/Q172HCPU Motion controller (SV13/SV22) Programming Manual (VIRTUAL MODE)

1.2 Features

The Motion CPU and Motion SFC program have the following features.

1.2.1 Features of Motion SFC programs

(1) Since a program intelligible for anyone can be created in flow chart form by macking a sequence of machine operation correspond to each operation step, maintenance nature improves.

(2) Since transition conditions are judged with Motion CPU side and positioning starts, there is not dispersion in the response time influenced by PLC scan time.

(3) High speed and high response processing is realizable with the step processing method (only active steps) of Motion SFC.

(4) Not only positioning control but also numerical operations, device SET/RST, etc. can be processed with Motion CPU side, making via PLC CPU is unnecessary and a tact time can be shortened.

(5) By transition condition description peculiar to Motion SFC, the instructions to servo amplifier is possible at completion of starting condition.

(6) By transition condition description peculiar to Motion SFC, after starting, transition to next step is possible without waiting for positioning completion.

(7) Motion SFC program that responds and executes it at high speed for interrupt input from external source can be set.

(8) Motion SFC program executed in the fixed cycle (0.88ms, 1.77ms, 3.55ms, 7.11ms, 14.2ms) by synchronizing to the Motion operation cycle can be set.

1.2.2 Performance specifications

(1) Basic specifications of Q173HCPU/Q172HCPU

(a) Motion control specifications

Item		Q173HCPU	Q173HCPU-T	Q172HCPU	Q172HCPU-T
Number of control axes		Up to 32 axes		Up to 8 axes	
Operation cycle (default)	SV13	0.44ms/ 1 to 3 axes 0.88ms/ 4 to 10 axes 1.77ms/11 to 20 axes 3.55ms/21 to 32 axes		0.44ms/ 1 to 3 axes 0.88ms/ 4 to 8 axes	
	SV22	0.88ms/ 1 to 5 axes 1.77ms/ 6 to 14 axes 3.55ms/15 to 28 axes 7.11ms/29 to 32 axes	—	0.88ms/ 1 to 4 axes 1.77ms/ 5 to 8 axes	—
Interpolation functions		Linear interpolation (Up to 4 axes), Circular interpolation (2 axes), Helical interpolation (3 axes)			
Control modes		PTP(Point to Point) control, Speed control, Speed-position control, Fixed-pitch feed, Constant speed control, Position follow-up control, Speed control with fixed position stop, Speed switching control, High-speed oscillation control, Synchronous control (SV22)			
Acceleration/ deceleration control		Automatic trapezoidal acceleration/deceleration, S-curve acceleration/deceleration			
Compensation		Backlash compensation, Electronic gear, Phase compensation (SV22)			
Programming language		Motion SFC, Dedicated instruction, Mechanical support language (SV22)			
Servo program capacity		14k steps			
Number of positioning points		3200 points (Positioning data can be designated indirectly)			
Programming tool		IBM PC/AT			
Peripheral I/F		USB/SSCNET			
Teaching operation function		None	Provided (SV13 use)	None	Provided (SV13 use)
Home position return function		Proximity dog type (2 types), Count type (3 types), Data set type (2 types), Dog cradle type, Stopper type (2 types), Limit switch combined type (Home position return re-try function provided, home position shift function provided)			
JOG operation function		Provided			
Manual pulse generator operation function		Possible to connect 3 modules			
Synchronous encoder operation function		Possible to connect 12 modules		Possible to connect 8 modules	
M-code function		M-code output function provided M-code completion wait function provided			
Limit switch output function		Number of output points 32 points Watch data: Motion control data/Word device			
Absolute position system		Made compatible by setting battery to servo amplifier. (Possible to select the absolute data method or incremental method for each axis)			

Motion control specifications (continued)

Item	Q173HCPU	Q173HCPU-T	Q172HCPU	Q172HCPU-T
Number of SSCNETⅢ systems (Note-1)	2 systems		1 system	
Motion related interface module	Q172LX : 4 modules usable Q172EX : 6 modules usable Q173PX : 4 modules usable (Note-2)		Q172LX : 1 module usable Q172EX : 4 modules usable Q173PX : 3 modules usable (Note-2)	

(Note-1) : The servo amplifiers for SSCNET cannot be used.

(Note-2) : When using the incremental synchronous encoder (SV22 use), you can use above number of modules. When connecting the manual pulse generator, you can use only 1 module.

(b) Motion SFC Performance Specifications

Item				Q173HCPU/Q172HCPU
Motion SFC program capacity	Code total (Motion SFC chart+ Operation control+ Transition)			543k bytes
	Text total (Operation control+ Transition)			484k bytes
Motion SFC program	Number of Motion SFC programs			256 (No.0 to 255)
	Motion SFC chart size/program			Up to 64k bytes (Included Motion SFC chart comments)
	Number of Motion SFC steps/program			Up to 4094 steps
	Number of selective branches/branch			255
	Number of parallel branches/branch			255
	Parallel branch nesting			Up to 4 levels
Operation control program (F/FS) / Transition program (G)	Number of operation control programs			4096 with F(Once execution type) and FS(Scan execution type) combined. (F/FS0 to F/FS4095)
	Number of transition programs			4096(G0 to G4095)
	Code size/program			Up to approx. 64k bytes (32766 steps)
	Number of blocks (line)/program			Up to 8192 blocks (in the case of 4 steps(min)/blocks)
	Number of characters/block (line)			Up to 128 (comment included)
	Number of operand/block			Up to 64 (operand: constants, word device, bit devices)
	( ) nesting/block			Up to 32 levels
	Descriptive expression	Operation control program		Calculation expression/bit conditional expression
		Transition program		Calculation expression/bit conditional expression/ comparison conditional expression
Execute specification	Number of multi execute programs			Up to 256
	Number of multi active steps			Up to 256 steps/all programs
	Executed task	Normal task		Execute in motion main cycle
		Event task (Execution can be masked.)	Fixed cycle	Execute in fixed cycle (0.88ms, 1.77ms, 3.55ms, 7.11ms, 14.2ms)
			External interrupt	Execute when input ON is set among interrupt module QI60 (16 points).
			PLC interrupt	Execute with interrupt instruction (S(P).GINT) from PLC CPU.
		NMI task		Execute when input ON is set among interrupt module QI60 (16 points).

1.2.3 Operation control/transition control specifications

(1) Table of the operation control/transition control specifications

<table>
<tr><th>Item</th><th colspan="3">Specifications</th><th>Remark</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(D100), etc.</td></tr>
<tr><td rowspan="2">Conditional expression</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>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">Description 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>Input module non-loaded range</td><td>X</td><td>○</td><td>○</td><td rowspan="9">○</td><td rowspan="9">○</td><td rowspan="9">○</td><td>X100</td></tr>
<tr><td>Input module loaded range</td><td>PX</td><td>○</td><td>×</td><td>PX180</td></tr>
<tr><td rowspan="2">Output</td><td>Output module non-loaded range</td><td>Y</td><td>○</td><td>○</td><td>Y100</td></tr>
<tr><td>Output module loaded range</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>○</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">Special relay</td><td>M</td><td>○</td><td>○</td><td>M9000</td></tr>
</table>
○: usable
×: unusable

CAUTION
<Restrictions on write-enabled bit devices>
1) Write to device X is allowed only within the input module non-installed range.
2) Special relay has predetermined applications in the system.
Do not perform write to other than the user setting device.
(Note) : SET/RST is disabled in the following device ranges.
<table>
<tr><th>SET/RST disable range</th><th>Remark</th></tr>
<tr><td>M2001 to M2032</td><td>Start accept device</td></tr>
</table>
(Note) : DOUT output disabled in the following device ranges.
<table>
<tr><th>DOUT output disable range</th><th>Remark</th></tr>
<tr><td>Designation including M2000 to M2127</td><td>Dedicated device</td></tr>
<tr><td>M9000 to M9255</td><td>Special relay</td></tr>
</table>
</td><td>The input X/output Y are written with the actual input PX/actual output PY.
It does the layput of the I/O numbers of PX, PY by a set up of as system.
(In the operation control program/transition program, automatically represented as PX/PY according to the system setting information.)</td></tr>
</table>

Table of the operation control/transition control specification (continued)

<table>
<tr><th>Item</th><th colspan="4">Specifications</th><th>Remark</th></tr>
<tr><td>Word devices</td><td colspan="4">
<table>
<tr><th rowspan="2">Devices</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>Data register</td><td>D</td><td>○</td><td>○</td><td rowspan="5">○</td><td rowspan="5">○</td><td rowspan="5">○</td><td>DOL</td></tr>
<tr><td>Link register</td><td>W</td><td>○</td><td>○</td><td>W1F : F</td></tr>
<tr><td>Special register</td><td>D</td><td>○</td><td>○</td><td>D9000</td></tr>
<tr><td>Motion register</td><td>#</td><td>○</td><td>○</td><td>#0F</td></tr>
<tr><td>Coasting timer</td><td>FT</td><td>○</td><td>×</td><td>FT</td></tr>
</table>
○: usable
×: unusable
CAUTION
<Restrictions on write-enabled word devices>
1) Special register has predetermined applications in the system.
Do not perform write to other than the user-set device.
</td><td></td></tr>
<tr><td rowspan="5">Data type</td><td rowspan="2">(None)</td><td colspan="2">16-bit integer type (signed)</td><td>-32768 to 32767</td><td rowspan="2">K10, D100, etc.</td></tr>
<tr><td colspan="2">16-bit integer type (unsigned)</td><td>0 to 65535</td></tr>
<tr><td rowspan="2">L</td><td colspan="2">32-bit integer type (signed)</td><td>-2147483648 to 2147483647</td><td rowspan="2">2000000000, W100L, etc.</td></tr>
<tr><td colspan="2">32-bit integer type (unsigned)</td><td>0 to 4294967295</td></tr>
<tr><td>F</td><td colspan="2">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="2">Constant</td><td>K</td><td>Decimal constant</td><td colspan="2" rowspan="2">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="2">K-100, H0FFL, etc.
'K' may be omitted.</td></tr>
<tr><td>H</td><td>Hexadecimal constant</td></tr>
<tr><td rowspan="11">Number of instructions</td><td colspan="2">Binary operation</td><td>6</td><td rowspan="11">63 in total</td><td rowspan="11"></td></tr>
<tr><td colspan="2">Bit operation</td><td>6</td></tr>
<tr><td colspan="2">Sign</td><td>1</td></tr>
<tr><td colspan="2">Standard function</td><td>15</td></tr>
<tr><td colspan="2">Type conversion</td><td>6</td></tr>
<tr><td colspan="2">Bit device status</td><td>2</td></tr>
<tr><td colspan="2">Bit device control</td><td>5</td></tr>
<tr><td colspan="2">Logical operation</td><td>4</td></tr>
<tr><td colspan="2">Comparison operation</td><td>6</td></tr>
<tr><td colspan="2">Motion dedicated function</td><td>2</td></tr>
<tr><td colspan="2">Others</td><td>10</td></tr>
<tr><td rowspan="2">Read/write response of input PX, output PY</td><td colspan="2">Input response</td><td colspan="2">Direct read control at instruction execution.</td><td rowspan="2"></td></tr>
<tr><td colspan="2">Output response</td><td colspan="2">Direct write control at instruction execution.</td></tr>
</table>

(2) Table of the operation control/transition instruction

Classification	Symbol	Function	Format	Basic steps	Usable step		Y/N transition's conditional expression	Section of reference
					F/FS	G		
Binary operation	=	Substitution	(D)=(S)	4	○	○	—	5.4.1
	+	Addition	(S1)+(S2)	4	○	○	—	5.4.2
	-	Subtraction	(S1)-(S2)	4	○	○	—	5.4.3
	*	Multiplication	(S1)*(S2)	4	○	○	—	5.4.4
	/	Division	(S1)/(S2)	4	○	○	—	5.4.5
	%	Remainder	(S1)%(S2)	4	○	○	—	5.4.6
Bit operation	~	Bit inversion (complement)	~(S)	2	○	○	—	5.5.1
	&	Bit logical AND	(S1)&(S2)	4	○	○	—	5.5.2
	|	Bit logical OR	(S1)|(S2)	4	○	○	—	5.5.3
	^	Bit exclusive OR	(S1)^(S2)	4	○	○	—	5.5.4
	>>	Bit right shift	(S1)>>(S2)	4	○	○	—	5.5.5
	<<	Bit left shift	(S1)<<(S2)	4	○	○	—	5.5.6
Sign	-	Sign inversion (complement of 2)	-(S)	2	○	○	—	5.5.7
Standard function	SIN	Sine	SIN(S)	2	○	○	—	5.6.1
	COS	Cosine	COS(S)	2	○	○	—	5.6.2
	TAN	Tangent	TAN(S)	2	○	○	—	5.6.3
	ASIN	Arcsine	ASIN(S)	2	○	○	—	5.6.4
	ACOS	Arccosine	ACOS(S)	2	○	○	—	5.6.5
	ATAN	Arctangent	ATAN(S)	2	○	○	—	5.6.6
	SQRT	Square root	SQRT(S)	2	○	○	—	5.6.7
	LN	Natural logarithm	LN(S)	2	○	○	—	5.6.8
	EXP	Exponential operation	EXP(S)	2	○	○	—	5.6.9
	ABS	Absolute value	ABS(S)	2	○	○	—	5.6.10
	RND	Round-off	RND(S)	2	○	○	—	5.6.11
	FIX	Round-down	FIX(S)	2	○	○	—	5.6.12
	FUP	Round-up	FUP(S)	2	○	○	—	5.6.13
	BIN	BCD ⟶ BIN conversion	BIN(S)	2	○	○	—	5.6.14
	BCD	BIN ⟶ BCD conversion	BCD(S)	2	○	○	—	5.6.15
Type conversion	SHORT	Convert into 16-bit integer type (signed)	SHORT(S)	2	○	○	—	5.7.1
	USHORT	Convert into 16-bit integer type (unsigned)	USHORT(S)	2	○	○	—	5.7.2
	LONG	Convert into 32-bit integer type (signed)	LONG(S)	2	○	○	—	5.7.3
	ULONG	Convert into 32-bit integer type (unsigned)	ULONG(S)	2	○	○	—	5.7.4
	FLOAT	Regard as signed data and convert into 64-bit floating point type	FLOAT(S)	2	○	○	—	5.7.5
	UFLOAT	Regard as unsigned data and convert into 64-bit floating point type	UFLOAT(S)	2	○	○	—	5.7.6
Bit device status	(None)	ON (normally open contact)	(S)	2	○	○	○	5.8.1
	!	OFF (normally closed contact)	!(S)	2	○	○	○	5.8.2
Bit device control	SET	Device set	SET(D)	3	○	○	—	5.9.1
			SET(D)= (conditional expression)	4	○	○	—	
	RST	Device reset	RST(D)	3	○	○	—	5.9.2
			RST(D)=(conditional expression)	4	○	○	—	
	DOUT	Device output	DOUT(D),(S)	4	○	○	—	5.9.3
	DIN	Device input	DIN(D),(S)	4	○	○	—	5.9.4
	OUT	Bit device output	OUT(D)=(conditional expression)	4	○	○	—	5.9.5

Table of the operation control/transition instruction (continued)

Classification	Symbol	Function	Format	Basic steps	Usable step		Y/N transition's conditional expression	Section of reference
					F/FS	G		
Logical operation	(None)	Logical acknowledgment	(Conditional expression)	0	○	○	○	5.10.1
	!	Logical negation	!(Conditional expression)	2	○	○	○	5.10.2
	*	Logical AND	(Conditional expression) * (conditional expression)	4	○	○	○	5.10.3
	+	Logical OR	(Conditional expression) + (conditional expression)	4	○	○	○	5.10.4
Comparison operation	==	Equal to	(Conditional expression) == (conditional expression)	4	○	○	○	5.11.1
	!=	Not equal to	(Conditional expression) != (conditional expression)	4	○	○	○	5.11.2
	<	Less than	(Conditional expression) < (conditional expression)	4	○	○	○	5.11.3
	<=	Less than or equal to	(Conditional expression) <= (conditional expression)	4	○	○	○	5.11.4
	>	More than	(Conditional expression) > (conditional expression)	4	○	○	○	5.11.5
	>=	More than or equal to	(Conditional expression) >= (conditional expression)	4	○	○	○	5.11.6
Motion dedicated function	CHGV	Speed change request	CHGV((S1),(S2))	4	○	○	—	5.12.1
	CHGT	Torque limit value change request	CHGT((S1),(S2))	4	○	○	—	5.12.2
Others	EI	Event task enable	EI	1	○	○	—	5.13.1
	DI	Event task disable	DI	1	○	○	—	5.13.2
	NOP	No operation	NOP	1	○	○	—	5.13.3
	BMOV	Block transfer	BMOV(D),(S),(n)	6	○	○	—	5.13.4
	FMOV	Same data block transfer	FMOV(D),(S),(n)	6	○	○	—	5.13.5
	MULTW	Write device data to shared CPU memory of the self CPU	MULTW(D),(S),(n),(D1)	8	○	○	—	5.13.6
	MULTR	Read device data from shared CPU memory of the other CPU	MULTR(D),(S1),(S2),(n)	7	○	○	—	5.13.7
	TO	Write device data to intelligent function module/special function module.	TO(D1),(D2),(S),(n)	7	○	○	—	5.13.8
	FROM	Read device data from intelligent function module/special function module.	FROM(D),(S1),(S2),(n)	7	○	○	—	5.13.9
	TIME	Time to wait	TIME(S)	7	—	○	—	5.13.10

(3) Rough calculation expression of singleprogram for operation control/transition program

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)

1.2.4 Differences Between Q173HCPU/Q172HCPU and Q173CPU(N)/Q172CPU(N)

(1) Differences between Q173HCPU/Q172HCPU and Q173CPU(N)/Q172CPU(N)

Item		Q173HCPU	Q172HCPU	Q173CPU(N)	Q172CPU(N)
Number of control axes		Up to 32 axes	Up to 8 axes	Up to 32 axes	Up to 8 axes
Operation cycle (Default) (It can be set up by parameters.)	SV13	0.44ms/ 1 to 3 axes 0.88ms/ 4 to 10 axes 1.77ms/11 to 20 axes 3.55ms/21 to 32 axes	0.44ms/ 1 to 3 axes 0.88ms/ 4 to 8 axes	0.88ms/ 1 to 8 axes 1.77ms/ 9 to 16 axes 3.55ms/17 to 32 axes	0.88ms/ 1 to 8 axes
	SV22	0.88ms/ 1 to 5 axes 1.77ms/ 6 to 14 axes 3.55ms/15 to 28 axes 7.11ms/29 to 32 axes	0.88ms/ 1 to 5 axes 1.77ms/ 6 to 8 axes	0.88ms/ 1 to 4 axes 1.77ms/ 5 to 12 axes 3.55ms/13 to 24 axes 7.11ms/25 to 32 axes	0.88ms/ 1 to 4 axes 1.77ms/ 5 to 8 axes
Motion SFC program capacity		Code total : 543 kbyte Text total : 484 kbyte		Code total : 287 kbyte Text total : 224 kbyte	
Peripheral devices I/F		USB/SSCNET		USB/RS-232/SSCNET	
Servo amplifier I/F		SSCNETⅢ (Optical communication)	Q173HCPU : 2 systems Q172HCPU : 1 system	SSCNET	Q173CPU(N) : 4 systems (Note-1) Q172CPU(N) : 1 system
Fixed position stop function with speed control		○		—	
Phase compensation function		○		—	
Indirect setting of home position return data		Indirect setting with word devices (D, W, #) of Motion CPU.		Only direct setting by programming software.	
Expansion of speed setting range in the unit [degree]		• When the speed control 10×multiplier setting for degree axis is valid ; 0.01 to 21474836.47[degree/min] • When the speed control 10×multiplier setting for degree axis is invalid ; 0.001 to 2147483.647[degree/min]		0.001 to 2147483.647[degree/min] fixed	
Fetch of external signal input		Q172LX/General input of servo amplifier (Note-2)		Q172LX	
Optional data monitor function		3 points/axis (Specified device D, W, #)		—	
Minor error [303], [304]		When the speed change is executed after positioning automatic deceleration start or during deceleration by the JOG start command signal (M3202+20n, M3203+20n) OFF, since the speed change request is ignored, a minor error [303], [304] will not occur.		When the speed change is executed after positioning automatic deceleration start or during deceleration by the JOG start command signal (M3202+20n, M3203+20n) OFF, a minor error [303], [304] will occur.	
Processing with power supply OFF of servo amplifier		Servo OFF is executed for all servo amplifier connected behind servo amplifier with which the control power supply was turned OFF.		Servo OFF is executed for only servo amplifier with which the control power supply was turned OFF.	
Back-up battery for internal memory		Internal rechargeable battery (Set the external battery (Q6BAT) if continuous power off time is longer for 1 month or more.) (Note-3)		Internal rechargeable battery (Set the external battery (A6BAT/MR-BAT) if continuous power off time is longer for 1 month or more.) (Note-4)	

(Note-1) : Use the Dividing unit (Q173DV) or dividing cable (Q173J2B△CBL☐M/Q173HB△CBL☐M).

(Note-2) : When selecting the each servo amplifier input, the speed/position switching control cannot be executed. And, the external stop input cannot be used.

(Note-3) : When adding the external battery (Q6BAT), use the Q170HBATC.

(Note-4) : When adding the external battery (A6BAT/MR-BAT), use the Q173DV (Q173HCPU use) or Q170BAT (Q172HCPU use).

1.2.5 Restrictions by the version

There are restrictions in the function that can be used by the version of Motion CPU module, operating system software and programming software.
The combination of each version and a function is shown below.

Function	Operating system software version (Note-1)	Programming software version	CPU module version		Section of reference
			Q173HCPU(-T)	Q172HCPU(-T)	
Bit device setting by Motion SFC instruction (BMOV, FMOV, MULTW, MULTR, TO, FROM)	C	M	—	—	Section 5.13.4 to 5.13.9
Fully closed loop control servo amplifer MR-J3-☐B-RJ006 (Note-2)	D	N	—	—	—
Filter external setting signal	D	N	—	—	—

—: There is no restriction by the version.

(Note-1): SV13/SV22 is the completely same version.

(Note-2): When combinig with the programming software ,be sure to use the setup software "MRZJW3-SETUP221E(Version B1 or later)".

1.2.6 Positioning dedicated devices/special relays/special registers

(1) Positioning dedicated devices

The following section describes the positioning dedicated devices.

A range of up to 32 axes is valid in Q173HCPU, and a range of up to 8 axes is valid in Q172HCPU.
Refer to the "Q173HCPU/Q172HCPU Motion controller (SV13/SV22) Programming Manual (REAL MODE)", "Q173HCPU/Q172HCPU Motion controller (SV22) Programming Manual (VIRTUAL MODE)" for details of the positioning dedicated devices.

(a) Table of the internal relays

- Overall configuration

SV13		SV22	
Device No.	Purpose	Device No.	Purpose
M0 to	User device (2000 points)	M0 to	User device (2000 points)
M2000 to	Common device (320 points)	M2000 to	Common device (320 points)
M2320 to	Special relay allocated device (Status) (80 points)	M2320 to	Special relay allocated device (Status) (80 points)
M2400 to	Axis status (20 points × 32 axes)	M2400 to	Axis status (20 points × 32 axes) Real mode……Each axis Virtual mode….Output module
M3040 to	Unusable (32 points)	M3040 to	Unusable (32 points)
M3072 to	Common device (Command signal) (64 points)	M3072 to	Common device (Command signal) (64 points)
M3136 to	Special relay allocated device (Command signal) (64 points)	M3136 to	Special relay allocated device (Command signal) (64 points)
M3200 to	Axis command signal (20 points × 32 axes)	M3200 to	Axis command signal (20 points × 32 axes) Real mode……Each axis Virtual mode….Output module
M3840 to M4799	User device (960 points)	M3840 to	Unusable (Note-1) (160 points)
		M4000 to	Virtual servomotor axis status (Note-1,2) (20 points × 32 axes) (Mechanical system setting axis only)
		M4640 to	Synchronous encoder axis status(Note-2) (4 points × 12 axes)
		M4688 to M4799	Unusable (Note-1) (112 points)

- Overall configuration (Continued)

SV13		SV22	
Device No.	Purpose	Device No.	Purpose
M4800 to M8191	User device (3392 points)	M4800 to	Virtual servomotor axis command signal (Note-1,2) (20 points × 32 axes) (Mechanical system setting axis only)
		M5440 to	Synchronous encoder axis command signal (Note-2) (4 points × 12 axes)
		M5488 to	Cam axis command signal (Note-1,2) (1 point × 32 axes) (Mechanical system setting axis only)
		M5520 to	Smoothing clutch complete signal (Note-1,2) (2 points × 32 axes)
		M5584 to	Unusable (Note-1) (16 points)
		M5600 to M8191	User device (2592 points)

(Note-1) : It can be used as an user device in the SV22 real mode only.

(Note-2) : Do not set the M4000 to M5599 as a latch range in the virtual mode.

1) Table of the axis statuses (SV13/SV22)

Device No.	Signal name	Device No.	Signal name
M2400 to M2419	Axis 1 status	M2720 to M2739	Axis 17 status
M2420 to M2439	Axis 2 status	M2740 to M2759	Axis 18 status
M2440 to M2459	Axis 3 status	M2760 to M2779	Axis 19 status
M2460 to M2479	Axis 4 status	M2780 to M2799	Axis 20 status
M2480 to M2499	Axis 5 status	M2800 to M2819	Axis 21 status
M2500 to M2519	Axis 6 status	M2820 to M2839	Axis 22 status
M2520 to M2539	Axis 7 status	M2840 to M2859	Axis 23 status
M2540 to M2559	Axis 8 status	M2860 to M2879	Axis 24 status
M2560 to M2579	Axis 9 status	M2880 to M2899	Axis 25 status
M2580 to M2599	Axis 10 status	M2900 to M2919	Axis 26 status
M2600 to M2619	Axis 11 status	M2920 to M2939	Axis 27 status
M2620 to M2639	Axis 12 status	M2940 to M2959	Axis 28 status
M2640 to M2659	Axis 13 status	M2960 to M2979	Axis 29 status
M2660 to M2679	Axis 14 status	M2980 to M2999	Axis 30 status
M2680 to M2699	Axis 15 status	M3000 to M3019	Axis 31 status
M2700 to M2719	Axis 16 status	M3020 to M3039	Axis 32 status

- Detailes of each axis

Device No.	Signal name	
M2400 + 20n	Positioning start complete	
M2401 + 20n	Positioning complete	
M2402 + 20n	In-position	
M2403 + 20n	Command in-position	
M2404 + 20n	Speed controlling	
M2405 + 20n	Speed/position switching latch signal	
M2406 + 20n	Zero pass signal	
M2407 + 20n	Error detection signal	
M2408 + 20n	Servo error detection signal	
M2409 + 20n	Home position return request signal	
M2410 + 20n	Home position return completion signal	
M2411 + 20n	External signals	FLS signal
M2412 + 20n		RLS signal
M2413 + 20n		STOP signal
M2414 + 20n		DOG/CHANGE signal
M2415 + 20n	Servo ready signal	
M2416 + 20n	Torque limiting signal	
M2417 + 20n	Unusable	
M2418 + 20n	Virtual mode continuation operation disable warning signal (SV22)	
M2419 + 20n	M-code outputting signal	

(Note-1) : "n" in the above device No. shows the numerical value which correspond to axis No.

Q173HCPU : Axis No.1 to No.32 (n=0 to 31)
Q172HCPU : Axis No.1 to No.8 (n=0 to 7)

(Note-2) : Device area of 9 axes or more is unusable in the Q172HCPU.

2) Table of the axis command signals (SV13/SV22)

Device No.	Signal name	Device No.	Signal name
M3200 to M3219	Axis 1 command signal	M3520 to M3539	Axis 17 command signal
M3220 to M3239	Axis 2 command signal	M3540 to M3559	Axis 18 command signal
M3240 to M3259	Axis 3 command signal	M3560 to M3579	Axis 19 command signal
M3260 to M3279	Axis 4 command signal	M3580 to M3599	Axis 20 command signal
M3280 to M3299	Axis 5 command signal	M3600 to M3619	Axis 21 command signal
M3300 to M3319	Axis 6 command signal	M3620 to M3639	Axis 22 command signal
M3320 to M3339	Axis 7 command signal	M3640 to M3659	Axis 23 command signal
M3340 to M3359	Axis 8 command signal	M3660 to M3679	Axis 24 command signal
M3360 to M3379	Axis 9 command signal	M3680 to M3699	Axis 25 command signal
M3380 to M3399	Axis 10 command signal	M3700 to M3719	Axis 26 command signal
M3400 to M3419	Axis 11 command signal	M3720 to M3739	Axis 27 command signal
M3420 to M3439	Axis 12 command signal	M3740 to M3759	Axis 28 command signal
M3440 to M3459	Axis 13 command signal	M3760 to M3779	Axis 29 command signal
M3460 to M3479	Axis 14 command signal	M3780 to M3799	Axis 30 command signal
M3480 to M3499	Axis 15 command signal	M3800 to M3819	Axis 31 command signal
M3500 to M3519	Axis 16 command signal	M3820 to M3839	Axis 32 command signal

- Detailes of each axis

Device No.	SV13	SV22
M3200 + 20n	Stop command	Stop command
M3201 + 20n	Rapid stop command	Rapid stop command
M3202 + 20n	Forward rotation JOG start command	Forward rotation JOG start command
M3203 + 20n	Reverse rotation JOG start command	Reverse rotation JOG start comannd
M3204 + 20n	Complete signal OFF command	Complete signal OFF command
M3205 + 20n	Speed/position switching enable command	Speed/position switching enable comannd
M3206 + 20n	Unusable	Unusable
M3207 + 20n	Error reset command	Error reset command
M3208 + 20n	Servo error reset command	Servo error reset command
M3209 + 20n	External stop input disable at start command	External stop input disable at start command
M3210 + 20n	Unusable	Unusable
M3211 + 20n		
M3212 + 20n	Feed current value update request command	Feed current value update request command
M3213 + 20n	Unusable	Address clutch reference setting command
M3214 + 20n		Cam reference position setting command
M3215 + 20n	Servo OFF command	Servo OFF command
M3216 + 20n	Gain changing command	Gain changing command
M3217 + 20n	Unusable	Unusable
M3218 + 20n	Control loop changing command	Control loop changing command
M3219 + 20n	FIN signal	FIN signal

(Note-1) : "n" in the above device No. shows the numerical value which correspond to axis No.
Q173HCPU : Axis No.1 to No.32 (n=0 to 31)
Q172HCPU : Axis No.1 to No.8 (n=0 to 7)

(Note-2) : Device area of 9 axes or more is unusable in the Q172HCPU.

3) Table of the virtual servomotor axis statuses (SV22 only)

Device No.	Signal name	Device No.	Signal name
M4000 to M4019	Axis 1 status	M4320 to M4339	Axis 17 status
M4020 to M4039	Axis 2 status	M4340 to M4359	Axis 18 status
M4040 to M4059	Axis 3 status	M4360 to M4379	Axis 19 status
M4060 to M4079	Axis 4 status	M4380 to M4399	Axis 20 status
M4080 to M4099	Axis 5 status	M4400 to M4419	Axis 21 status
M4100 to M4119	Axis 6 status	M4420 to M4439	Axis 22 status
M4120 to M4139	Axis 7 status	M4440 to M4459	Axis 23 status
M4140 to M4159	Axis 8 status	M4460 to M4479	Axis 24 status
M4160 to M4179	Axis 9 status	M4480 to M4499	Axis 25 status
M4180 to M4199	Axis 10 status	M4500 to M4519	Axis 26 status
M4200 to M4219	Axis 11 status	M4520 to M4539	Axis 27 status
M4220 to M4239	Axis 12 status	M4540 to M4559	Axis 28 status
M4240 to M4259	Axis 13 status	M4560 to M4579	Axis 29 status
M4260 to M4279	Axis 14 status	M4580 to M4599	Axis 30 status
M4280 to M4299	Axis 15 status	M4600 to M4619	Axis 31 status
M4300 to M4319	Axis 16 status	M4620 to M4639	Axis 32 status

- Detailes of each axis

Device No.	Signal name
M4000 + 20n	Positioning start complete
M4001 + 20n	Positioning complete
M4002 + 20n	Unusable
M4003 + 20n	Command in-position
M4004 + 20n	Speed controlling
M4005 + 20n	Unusable
M4006 + 20n	
M4007 + 20n	Error detection
M4008 + 20n	Unusable
M4009 + 20n	
M4010 + 20n	
M4011 + 20n	
M4012 + 20n	
M4013 + 20n	
M4014 + 20n	
M4015 + 20n	
M4016 + 20n	
M4017 + 20n	
M4018 + 20n	
M4019 + 20n	M-code outputting signal

(Note-1) : "n" in the above device No. shows the numerical value which correspond to axis No.

Q173HCPU : Axis No.1 to No.32 (n=0 to 31)
Q172HCPU : Axis No.1 to No.8 (n=0 to 7)

(Note-2) : The unused axis areas in the mechanical system program can be used as an user device.

4) Table of the virtual servomotor axis command signals (SV22 only)

Device No.	Signal name	Device No.	Signal name
M4800 to M4819	Axis 1 command signal	M5120 to M5139	Axis 17 command signal
M4820 to M4839	Axis 2 command signal	M5140 to M5159	Axis 18 command signal
M4840 to M4859	Axis 3 command signal	M5160 to M5179	Axis 19 command signal
M4860 to M4879	Axis 4 command signal	M5180 to M5199	Axis 20 command signal
M4880 to M4899	Axis 5 command signal	M5200 to M5219	Axis 21 command signal
M4900 to M4919	Axis 6 command signal	M5220 to M5239	Axis 22 command signal
M4920 to M4939	Axis 7 command signal	M5240 to M5259	Axis 23 command signal
M4940 to M4959	Axis 8 command signal	M5260 to M5279	Axis 24 command signal
M4960 to M4979	Axis 9 command signal	M5280 to M5299	Axis 25 command signal
M4980 to M4999	Axis 10 command signal	M5300 to M5319	Axis 26 command signal
M5000 to M5019	Axis 11 command signal	M5320 to M5339	Axis 27 command signal
M5020 to M5039	Axis 12 command signal	M5340 to M5359	Axis 28 command signal
M5040 to M5059	Axis 13 command signal	M5360 to M5379	Axis 29 command signal
M5060 to M5079	Axis 14 command signal	M5380 to M5399	Axis 30 command signal
M5080 to M5099	Axis 15 command signal	M5400 to M5419	Axis 31 command signal
M5100 to M5119	Axis 16 command signal	M5420 to M5439	Axis 32 command signal

- Detailes of each axis

Device No.	Signal name
M4800 + 20n	Stop command
M4801 + 20n	Rapid stop command
M4802 + 20n	Forward rotation JOG start command
M4803 + 20n	Reverse rotation JOG start command
M4804 + 20n	Complete signal OFF command
M4805 + 20n	Unusable
M4806 + 20n	
M4807 + 20n	Error reset command
M4808 + 20n	Unusable
M4809 + 20n	External stop input disable at start command
M4810 + 20n	Unusable
M4811 + 20n	
M4812 + 20n	
M4813 + 20n	
M4814 + 20n	
M4815 + 20n	
M4816 + 20n	
M4817 + 20n	
M4818 + 20n	
M4819 + 20n	FIN signal

(Note-1) : "n" in the above device No. shows the numerical value which correspond to axis No.
Q173HCPU : Axis No.1 to No.32 (n=0 to 31)
Q172HCPU : Axis No.1 to No.8 (n=0 to 7)

(Note-2) : The unused axis areas in the mechanical system program can be used as an user device.

5) Table of the synchronous encoder axis statuses (SV22 only)

Device No.	Signal name	
M4640	Axis 1	Error detection
M4641		External signal TREN
M4642		Virtual mode continuation operation disable warning
M4643		Unusable
M4644	Axis 2	Error detection
M4645		External signal TREN
M4646		Virtual mode continuation operation disable warning
M4647		Unusable
M4648	Axis 3	Error detection
M4649		External signal TREN
M4650		Virtual mode continuation operation disable warning
M4651		Unusable
M4652	Axis 4	Error detection
M4653		External signal TREN
M4654		Virtual mode continuation operation disable warning
M4655		Unusable
M4656	Axis 5	Error detection
M4657		External signal TREN
M4658		Virtual mode continuation operation disable warning
M4659		Unusable
M4660	Axis 6	Error detection
M4661		External signal TREN
M4662		Virtual mode continuation operation disable warning
M4663		Unusable
M4664	Axis 7	Error detection
M4665		External signal TREN
M4666		Virtual mode continuation operation disable warning
M4667		Unusable
M4668	Axis 8	Error detection
M4669		External signal TREN
M4670		Virtual mode continuation operation disable warning
M4671		Unusable
M4672	Axis 9	Error detection
M4673		External signal TREN
M4674		Virtual mode continuation operation disable warning
M4675		Unusable
M4676	Axis 10	Error detection
M4677		External signal TREN
M4678		Virtual mode continuation operation disable warning
M4679		Unusable
M4680	Axis 11	Error detection
M4681		External signal TREN
M4682		Virtual mode continuation operation disable warning
M4683		Unusable
M4684	Axis 12	Error detection
M4685		External signal TREN
M4686		Virtual mode continuation operation disable warning
M4687		Unusable

(Note-1) : The range of axis No.1 to 8 is valid in the Q172HCPU.

(Note-2) : Device area of 9 axes or more is unusable in the Q172HCPU.

6) Table of the syncronous encoder axis command signals (SV22 only)

Device No.	Signal name	
M5440	Axis 1	Error reset
M5441		Unusable
M5442		Unusable
M5443		Unusable
M5444	Axis 2	Error reset
M5445		Unusable
M5446		Unusable
M5447		Unusable
M5448	Axis 3	Error reset
M5449		Unusable
M5450		Unusable
M5451		Unusable
M5452	Axis 4	Error reset
M5453		Unusable
M5454		Unusable
M5455		Unusable
M5456	Axis 5	Error reset
M5457		Unusable
M5458		Unusable
M5459		Unusable
M5460	Axis 6	Error reset
M5461		Unusable
M5462		Unusable
M5463		Unusable
M5464	Axis 7	Error reset
M5465		Unusable
M5466		Unusable
M5467		Unusable
M5468	Axis 8	Error reset
M5469		Unusable
M5470		Unusable
M5471		Unusable
M5472	Axis 9	Error reset
M5473		Unusable
M5474		Unusable
M5475		Unusable
M5476	Axis 10	Error reset
M5477		Unusable
M5478		Unusable
M5479		Unusable
M5480	Axis 11	Error reset
M5481		Unusable
M5482		Unusable
M5483		Unusable
M5484	Axis 12	Error reset
M5485		Unusable
M5486		Unusable
M5487		Unusable

(Note-1) : The range of axis No.1 to 8 is valid in the Q172HCPU.

(Note-2) : Device area of 9 axes or more is unusable in the Q172HCPU.

7) Table of the cam axis command signals (SV22 only)

Device No.	Signal name
M5488	Axis 1 cam/ballscrew switching
M5489	Axis 2 cam/ballscrew switching
M5490	Axis 3 cam/ballscrew switching
M5491	Axis 4 cam/ballscrew switching
M5492	Axis 5 cam/ballscrew switching
M5493	Axis 6 cam/ballscrew switching
M5494	Axis 7 cam/ballscrew switching
M5495	Axis 8 cam/ballscrew switching
M5496	Axis 9 cam/ballscrew switching
M5497	Axis 10 cam/ballscrew switching
M5498	Axis 11 cam/ballscrew switching
M5499	Axis 12 cam/ballscrew switching
M5500	Axis 13 cam/ballscrew switching
M5501	Axis 14 cam/ballscrew switching
M5502	Axis 15 cam/ballscrew switching
M5503	Axis 16 cam/ballscrew switching
M5504	Axis 17 cam/ballscrew switching
M5505	Axis 18 cam/ballscrew switching
M5506	Axis 19 cam/ballscrew switching
M5507	Axis 20 cam/ballscrew switching
M5508	Axis 21 cam/ballscrew switching
M5509	Axis 22 cam/ballscrew switching
M5510	Axis 23 cam/ballscrew switching
M5511	Axis 24 cam/ballscrew switching
M5512	Axis 25 cam/ballscrew switching
M5513	Axis 26 cam/ballscrew switching
M5514	Axis 27 cam/ballscrew switching
M5515	Axis 28 cam/ballscrew switching
M5516	Axis 29 cam/ballscrew switching
M5517	Axis 30 cam/ballscrew switching
M5518	Axis 31 cam/ballscrew switching
M5519	Axis 32 cam/ballscrew changing

(Note-1) : The range of axis No.1 to 8 is valid in the Q172HCPU.

(Note-2) : Device area of 9 axes or more is unusable in the Q172HCPU.

(Note-3) : The unused axis areas in the cam axis command signal can be used as an user device.

8) Table of the smoothing clutch complete signals (SV22 only)

Device No.	Signal name		Refresh cycle	Fetch cycle	Signal direction	Remark
M5520	Output axis 1	Main shaft side	Operation cycle		Status signal	
M5521		Auxiliary input side				
M5522	Output axis 2	Main shaft side				
M5523		Auxiliary input side				
M5524	Output axis 3	Main shaft side				
M5525		Auxiliary input side				
M5526	Output axis 4	Main shaft side				
M5527		Auxiliary input side				
M5528	Output axis 5	Main shaft side				
M5529		Auxiliary input side				
M5530	Output axis 6	Main shaft side				
M5531		Auxiliary input side				
M5532	Output axis 7	Main shaft side				
M5533		Auxiliary input side				
M5534	Output axis 8	Main shaft side				
M5535		Auxiliary input side				
M5536	Output axis 9	Main shaft side				
M5537		Auxiliary input side				
M5538	Output axis 10	Main shaft side				
M5539		Auxiliary input side				
M5540	Output axis 11	Main shaft side				
M5541		Auxiliary input side				
M5542	Output axis 12	Main shaft side				
M5543		Auxiliary input side				
M5544	Output axis 13	Main shaft side				
M5545		Auxiliary input side				
M5546	Output axis 14	Main shaft side				
M5547		Auxiliary input side				
M5548	Output axis 15	Main shaft side				
M5549		Auxiliary input side				
M5550	Output axis 16	Main shaft side				
M5551		Auxiliary input side				
M5552	Output axis 17	Main shaft side				
M5553		Auxiliary input side				
M5554	Output axis 18	Main shaft side				
M5555		Auxiliary input side				
M5556	Output axis 19	Main shaft side				
M5557		Auxiliary input side				
M5558	Output axis 20	Main shaft side				
M5559		Auxiliary input side				
M5560	Output axis 21	Main shaft side				
M5561		Auxiliary input side				
M5562	Output axis 22	Main shaft side				
M5563		Auxiliary input side				
M5564	Output axis 23	Main shaft side				
M5565		Auxiliary input side				
M5566	Output axis 24	Main shaft side				
M5567		Auxiliary input side				
M5568	Output axis 25	Main shaft side				
M5569		Auxiliary input side				
M5570	Output axis 26	Main shaft side				
M5571		Auxiliary input side				
M5572	Output axis 27	Main shaft side				
M5573		Auxiliary input side				
M5574	Output axis 28	Main shaft side				
M5575		Auxiliary input side				
M5576	Output axis 29	Main shaft side				
M5577		Auxiliary input side				
M5578	Output axis 30	Main shaft side				
M5579		Auxiliary input side				
M5580	Output axis 31	Main shaft side				
M5581		Auxiliary input side				
M5582	Output axis 32	Main shaft side				
M5583		Auxiliary input side				

(Note-1) : The range of axis No.1 to 8 is valid in the Q172HCPU.

(Note-2) : Device area of 9 axes or more is unusable in the Q172HCPU.

(Note-3) : The unused axis areas in the mechanical system program can be used as an user device.

9) Table of the common devices (SV13/SV22)

SV13		SV22		Refresh cycle	Fetch cycle	Signal direction	Remark (Note-4)
Device No.	Signal name	Device No.	Signal name				
M2000	PLC ready flag	M2000	PLC ready flag		Main cycle	Command signal (Note-1)	M3072
M2001	Axis1 Start accept flag (32 points)	M2001	Axis1 Start accept flag (32 points)	Operation cycle		Status signal (Note-2,3)	
to	to	to	to				
M2032	Axis32	M2032	Axis32				
M2033	Unusable	M2033	Unusable	—	—	—	
M2034	Personal computer link communication error flag	M2034	Personal computer link communication error flag	Operation cycle		Status signal	
M2035	Motion SFC error history clear request flag (Note-5)	M2035	Motion SFC error history clear request flag (Note-5)		Main cycle	Command signal	M3080
M2036	Unusable (3 points)	M2036	Unusable (3 points)	—	—	—	
to		to					
M2038		M2038					
M2039	Motion SFC error detection flag	M2039	Motion SFC error detection flag		Immediate	Status signal	
M2040	Speed switching point specified flag	M2040	Speed switcing point specified flag		At start	Command signal (Note-1)	M3073
M2041	System setting error flag	M2041	System setting error flag	Operation cycle		Status signal	
M2042	All axes servo ON command	M2042	All axes servo ON command		Operation cycle	Command signal (Note-1)	M3074
M2043	Unusable (4 points)	M2043	Real/virtual mode switching request		At virtual mode transition		M3075
M2044		M2044	Real/virtual mode switching status	At virtual mode transition		Status signal	
M2045		M2045	Real/virtual mode switching error detection flag				
M2046		M2046	Out-of-sync warning				
M2047	Motion slot fault detection flag	M2047	Motion slot fault detection flag	Operation cycle			
M2048	JOG operation simultaneous start command	M2048	JOG operation simultaneous start command		Main cycle	Command signal (Note-1)	M3076
M2049	All axes servo ON accept flag	M2049	All axes servo ON accept flag	Operation cycle		Status signal	
M2050	Start buffer full	M2050	Start buffer full				
M2051	Manual pulse generator 1 enable flag	M2051	Manual pulse generator 1 enable flag		Main cycle	Command signal (Note-1)	M3077
M2052	Manual pulse generator 2 enable flag	M2052	Manual pulse generator 2 enable flag				M3078
M2053	Manual pulse generator 3 enable flag	M2053	Manual pulse generator 3 enable flag				M3079

Table of the common devices (SV13/SV22) (continued)

SV13			SV22				Refresh cycle	Fetch cycle	Signal direction	Remark (Note-4)
Device No.	Signal name		Device No.	Signal name						
M2054	Operation cycle over flag		M2054	Operation cycle over flag			Operation cycle		Status signal	
M2055	Unusable (6 points)		M2055	Unusable (6 points)			—	—	—	
to			to							
M2060			M2060							
M2061	Axis 1	Speed changing flag (32 axes)	M2061	Axis 1	Speed changing flag (32 axes)		Operation cycle		Status signal (Note-2,3)	
to	to		to	to						
M2092	Axis 32		M2092	Axis 32						
M2093	Unusable (35 points)		M2093	Unusable (8 points)			—	—	—	
			to							
			M2100							
			M2101	Axis 1	Synchronous encoder current value changing flag (12 axes)		Operation cycle		Status signal (Note-2,3)	
to			to	to						
			M2112	Axis 12						
			M2113	Unusable (15 points)			—	—	—	
			to							
M2127			M2127							
M2128	Axis 1	Automatic decelerating flag (32 axes)	M2128	Axis 1	Automatic decelerating flag (32 axes)		Operation cycle		Status signal (Note-2,3)	
to	to		to	to						
M2159	Axis 32		M2159	Axis 32						
M2160	Unusable (80 points)		M2160	Output axis 1	Main shaft side	Clutch status (Note-6)				
			M2161		Auxiliary input side					
to			to	to	to					
			M2222	Output axis 32	Main shaft side					
			M2223		Auxiliary input side					
			M2224	Unusable (16 points)			—	—	—	
			to							
M2239			M2239							
M2240	Axis 1	Speed change "0" accepting flag (32 axes)	M2240	Axis 1	Speed change "0" accepting flag (32 axes)		Operation cycle		Status signal (Note-2,3)	
to	to		to	to						
M2271	Axis 32		M2271	Axis 32						
M2272	Axis 1	Control loop monitor status (32 axes)	M2272	Axis 1	Control loop monitor status (32 axes)					
to	to		to	to						
M2303	Axis 32		M2303	Axis 32						
M2304	Unusable (16 points)		M2304	Unusable (16 points)			—	—	—	
to			to							
M2319			M2319							

Explanation of the request register

No.	Function	Bit device	Request register
1	PLC ready flag	M2000	D704
2	Speed switching point specified flag	M2040	D705
3	All axes servo ON command	M2042	D706
4	Real/virtual mode switching request (SV22)	M2043	D707
5	JOG operation simultaneous start command	M2048	D708
6	Manual pulse generator 1 enable flag	M2051	D755
7	Manual pulse generator 2 enable flag	M2052	D756
8	Manual pulse generator 3 enable flag	M2053	D757

(Note-1) : Handling of D704 to D708 and D755 to D757 register
Because cannot be turn ON/OFF for every bit from the PLC CPU, the above bit devices are assigned to D register, and each bit device becomes on with the lowest rank bit 0 → 1 of each register, and each bit device becomes off with 1 → 0.
Use it when the above functions are requested from the PLC CPU using the S(P).DDRD and S(P).DDWR instruction. Refer to "3 MOTION DEDICATED PLC INSTRUCTION" for S(P).DDRD and S(P).DDWR instruction.

(Note-2) : Device area of 9 axes or more is unusable in the Q172HCPU.

(Note-3) : The range of axis No.1 to 8 is valid in the Q172HCPU.

(Note-4) : It can also be ordered the device of a remark column.

(Note-5) : M3080 does not turn off automatically. Turn it off as an user side.

(Note-6) : It is unusable in the SV22 real mode.

⚠CAUTION

- The data executed later becomes effective when the same device is executed simultaneously in the Motion SFC program and PLC program.

10) Table of the special relay allocated devices (Status) (SV13/SV22)

Device No.	Signal name	Refresh cycle	Fetch cycle	Signal direction	Remark(Note)
M2320	Fuse blown detection	Error occurrence		Status signal	M9000
M2321	AC/DC DOWN detection				M9005
M2322	Battery low				M9006
M2323	Battery low latch				M9007
M2324	Self-diagnostic error				M9008
M2325	Diagnostic error				M9010
M2326	Always ON	Main operation			M9036
M2327	Always OFF				M9037
M2328	Clock data error	Error occurrence			M9026
M2329	PCPU WDT error flag				M9073
M2330	PCPU READY complete flag	At request			M9074
M2331	Test mode ON flag				M9075
M2332	External forced stop input flag	Operation cycle			M9076
M2333	Manual pulse generator axis setting error flag	Error occurrence			M9077
M2334	TEST mode request error flag				M9078
M2335	Servo program setting error flag				M9079
M2336	CPU No.1 reset flag	At Status change			M9240
M2337	CPU No.2 reset flag				M9241
M2338	CPU No.3 reset flag				M9242
M2339	CPU No.4 reset flag				M9243
M2340	CPU No.1 error flag				M9244
M2341	CPU No.2 error flag				M9245
M2342	CPU No.3 error flag				M9246
M2343	CPU No.4 error flag				M9247
M2344	Unusable	—	—	—	—
M2345	CPU No.1 MULTR complete flag	At instruction completion		Status signal	M9216
M2346	CPU No.2 MULTR complete flag				M9217
M2347	CPU No.3 MULTR complete flag				M9218
M2348	CPU No.4 MULTR complete flag				M9219
M2349 to M2399	Unusable (51 points)	—	—	—	—

(Note) : The same status as a remark column is output.

11) Table of the common devices (Command signal) (SV13/SV22)

Device No.	Signal name	Refresh cycle	Fetch cycle	Signal direction	Remark (Note-1) , (Note-2)
M3072	PLC ready flag		Main cycle	Command signal	M2000
M3073	Speed switching point specified flag		At start		M2040
M3074	All axes servo ON command		Operation cycle		M2042
M3075	Real/virtual mode change request (SV22)		At virtual mode transition		M2043
M3076	JOG operation simultaneous start command		Main cycle		M2048
M3077	Manual pulse generator 1 enable flag				M2051
M3078	Manual pulse generator 2 enable flag				M2052
M3079	Manual pulse generator 3 enable flag				M2053
M3080	Motion SFC error history clear request flag (Note-3)				M2035
M3081 to M3135	Unusable (55 points)	—	—	—	—

(Note-1) : The device of a remarks column turns ON by OFF to ON of the above device, and the device of a remarks column turns OFF by ON to OFF of the above device. The state of a device is not in agreement when the device of a remarks column is turned on directly. In addition, when the request from a data register and the request from the above device are performed simultaneously, the request from the above device becomes effective.

(Note-2) : It can also be ordered the device of a remark column.

(Note-3) : M3080 does not turn off automatically. Turn it off as an user side.

12) Table of the special relay allocated devices (Command signal) (SV13/SV22)

Device No.	Signal name	Refresh cycle	Fetch cycle	Signal direction	Remark (Note-1), (Note-2)
M3136	Clock data set request		Main cycle	Command signal	M9025
M3137	Clock data read request				M9028
M3138	Error reset				M9060
M3139 to M3199	Unusable (61 points)	—	—	—	—

(Note-1) : The device of a remarks column turns ON by OFF to ON of the above device, and the device of a remarks column turns OFF by ON to OFF of the above device. The state of a device is not in agreement when the device of a remarks column is turned on directly.

(Note-2) : It can also be ordered the device of a remark column.

(b) Table of the data registers

- Overall configuration

SV13		SV22	
Device No.	Application	Device No.	Application
D0 to	Axis monitor device (20 points × 32 axes)	D0 to	Axis monitor device (20 points × 32 axes) Real mode……Each axis Virtual mode….Output module
D640 to	Control change register (2 points × 32 axes)	D640 to	Control change register (2 points × 32 axes)
D704 to	Common device (Command signal) (54 points)	D704 to	Common device (Command signal) (54 points)
D758 to	Common device (Monitor) (42 points)	D758 to	Common device (Monitor) (42 points)
D800 to D8191	User device (7392 points)	D800 to	Virtual servomotor axis monitor device (Note) (10 points × 32 axes) (Mechanical system setting axis only)
		D1120 to	Syncronous encoder axis monitor device (Note) (10 points × 12 axes)
		D1240 to	Cam axis monitor device (Note) (10 points × 32 axes)
		D1560 to D8191	User device (6632 points)

(Note) : It can be used as an user device in the SV22 real mode only.

1) Table of the each axis monitor devices (SV13/SV22)

Device No.	Signal name	Device No.	Signal name
D0 to D19	Axis 1 monitor device	D320 to D339	Axis 17 monitor device
D20 to D39	Axis 2 monitor device	D340 to D359	Axis 18 monitor device
D40 to D59	Axis 3 monitor device	D360 to D379	Axis 19 monitor device
D60 to D79	Axis 4 monitor device	D380 to D399	Axis 20 monitor device
D80 to D99	Axis 5 monitor device	D400 to D419	Axis 21 monitor device
D100 to D119	Axis 6 monitor device	D420 to D439	Axis 22 monitor device
D120 to D139	Axis 7 monitor device	D440 to D459	Axis 23 monitor device
D140 to D159	Axis 8 monitor device	D460 to D479	Axis 24 monitor device
D160 to D179	Axis 9 monitor device	D480 to D499	Axis 25 monitor device
D180 to D199	Axis 10 monitor device	D500 to D519	Axis 26 monitor device
D200 to D219	Axis 11 monitor device	D520 to D539	Axis 27 monitor device
D220 to D239	Axis 12 monitor device	D540 to D559	Axis 28 monitor device
D240 to D259	Axis 13 monitor device	D560 to D579	Axis 29 monitor device
D260 to D279	Axis 14 monitor device	D580 to D599	Axis 30 monitor device
D280 to D299	Axis 15 monitor device	D600 to D619	Axis 31 monitor device
D300 to D319	Axis 16 monitor device	D620 to D639	Axis 32 monitor device

- Detailes of each axis

Device No.	SV13/SV22(Real mode)	SV22(Virtual mode)	Signal derection
D0 + 20n D1 + 20n	Feed current value	Feed current value/roller cycle speed	Monitor device
D2 + 20n D3 + 20n	Real current value	Real current value	
D4 + 20n D5 + 20n	Deviation counter value	Deviation counter value	
D6 + 20n	Minor error code	Minor error code	
D7 + 20n	Major error code	Major error code	
D8 + 20n	Servo error code	Servo error code	
D9 + 20n	Home position return re-travel value	Hold	
D10 + 20n D11 + 20n	Travel value after proximity dog ON	Hold	
D12 + 20n	Execute program No.	—	
D13 + 20n	M-code	—	
D14 + 20n	Torque limit value	Torque limit value	
D15 + 20n	Data set pointer for constant-speed control	—	
D16 + 20n D17 + 20n	Travel value change register	—	Command device
D18 + 20n D19 + 20n	Real current value at stop input	Hold	Monitor device

(Note-1) : "n" in the above device No. shows the numerical value which correspond to axis No.
Q173HCPU : Axis No.1 to No.32 (n=0 to 31)
Q172HCPU : Axis No.1 to No.8 (n=0 to 7)

(Note-2) : Device area of 9 axes or more is unusable in the Q172HCPU.

MEMO

2) Table of the control change registers (SV13/SV22)

Device No.	Signal name	Device No.	Signal name
D640 D641	Axis 1 JOG speed setting register	D672 D673	Axis 17 JOG speed setting register
D642 D643	Axis 2 JOG speed setting register	D674 D675	Axis 18 JOG speed setting register
D644 D645	Axis 3 JOG speed setting register	D676 D677	Axis 19 JOG speed setting register
D646 D647	Axis 4 JOG speed setting register	D678 D679	Axis 20 JOG speed setting register
D648 D649	Axis 5 JOG speed setting register	D680 D681	Axis 21 JOG speed setting register
D650 D651	Axis 6 JOG speed setting register	D682 D683	Axis 22 JOG speed setting register
D652 D653	Axis 7 JOG speed setting register	D684 D685	Axis 23 JOG speed setting register
D654 D655	Axis 8 JOG speed setting register	D686 D687	Axis 24 JOG speed setting register
D656 D657	Axis 9 JOG speed setting register	D688 D689	Axis 25 JOG speed setting register
D658 D659	Axis 10 JOG speed setting register	D690 D691	Axis 26 JOG speed setting register
D660 D661	Axis 11 JOG speed setting register	D692 D693	Axis 27 JOG speed setting register
D662 D663	Axis 12 JOG speed setting register	D694 D695	Axis 28 JOG speed setting register
D664 D665	Axis 13 JOG speed setting register	D696 D697	Axis 29 JOG speed setting register
D666 D667	Axis 14 JOG speed setting register	D698 D699	Axis 30 JOG speed setting register
D668 D669	Axis 15 JOG speed setting register	D700 D701	Axis 31 JOG speed setting register
D670 D671	Axis 16 JOG speed setting register	D702 D703	Axis 32 JOG speed setting register

3) Table of the virtual servomotor axis monitor devices (SV22 only)

Device No.	Signal name	Device No.	Signal name
D800 to D809	Axis 1 monitor device	D960 to D969	Axis 17 monitor device
D810 to D819	Axis 2 monitor device	D970 to D979	Axis 18 monitor device
D820 to D829	Axis 3 monitor device	D980 to D989	Axis 19 monitor device
D830 to D839	Axis 4 monitor device	D990 to D999	Axis 20 monitor device
D840 to D849	Axis 5 monitor device	D1000 to D1009	Axis 21 monitor device
D850 to D859	Axis 6 monitor device	D1010 to D1019	Axis 22 monitor device
D860 to D869	Axis 7 monitor device	D1020 to D1029	Axis 23 monitor device
D870 to D879	Axis 8 monitor device	D1030 to D1039	Axis 24 monitor device
D880 to D889	Axis 9 monitor device	D1040 to D1049	Axis 25 monitor device
D890 to D899	Axis 10 monitor device	D1050 to D1059	Axis 26 monitor device
D900 to D909	Axis 11 monitor device	D1060 to D1069	Axis 27 monitor device
D910 to D919	Axis 12 monitor device	D1070 to D1079	Axis 28 monitor device
D920 to D929	Axis 13 monitor device	D1080 to D1089	Axis 29 monitor device
D930 to D939	Axis 14 monitor device	D1090 to D1099	Axis 30 monitor device
D940 to D949	Axis 15 monitor device	D1100 to D1109	Axis 31 monitor device
D950 to D959	Axis 16 monitor device	D1110 to D1119	Axis 32 monitor device

- Detailes of each axis

Device No.	Signal name
D800 + 10n D801 + 10n	Feed current value
D802 + 10n	Minor error code
D803 + 10n	Major error code
D804 + 10n	Execute program No.
D805 + 10n	M-code
D806 + 10n D807 + 10n	Current value after virtual servomotor axis main shaft's differential gear
D808 + 10n	Error search output axis No.
D809 + 10n	Data set pointer for constant-speed control

(Note-1) : "n" in the above device No. shows the numerical value which correspond to axis No.

Q173HCPU : Axis No.1 to No.32 (n=0 to 31)
Q172HCPU : Axis No.1 to No.8 (n=0 to 7)

(Note-2) : The unused axis areas in the mechanical system program can be used as an user device.

4) Table of the synchronous encoder axis monitor devices (SV22 only)

Device No.	Signal name
D1120 to D1129	Axis 1 monitor device
D1130 to D1139	Axis 2 monitor device
D1140 to D1149	Axis 3 monitor device
D1150 to D1159	Axis 4 monitor device
D1160 to D1169	Axis 5 monitor device
D1170 to D1179	Axis 6 monitor device
D1180 to D1189	Axis 7 monitor device
D1190 to D1199	Axis 8 monitor device
D1200 to D1209	Axis 9 monitor device
D1210 to D1219	Axis 10 monitor device
D1220 to D1229	Axis 11 monitor device
D1230 to D1239	Axis 12 monitor device

- Detailes of each axis

Device No.	Signal name
D1120 + 10n D1121 + 10n	Current value
D1122 + 10n	Minor error code
D1123 + 10n	Major error code
D1124 + 10n	Unusable
D1125 + 10n	
D1126 + 10n D1127 + 10n	Current value after synchronous encoder axis main shaft's differential gear
D1128 + 10n	Error search output axis No.
D1129 + 10n	Unusable

(Note-1) : "n" in the above device No. shows the numerical value which correspond to axis No.
Q173HCPU : Axis No.1 to No.12 (n=0 to 11)
Q172HCPU : Axis No.1 to No.8 (n=0 to 7)
(Note-2) : Device area of 9 axes or more is unusable in the Q172HCPU.

5) Table of the cam axis monitor devices (SV22 only)

Device No.	Signal name	Device No.	Signal name
D1240 to D1249	Axis 1 monitor device	D1400 to D1409	Axis 17 monitor device
D1250 to D1259	Axis 2 monitor device	D1410 to D1419	Axis 18 monitor device
D1260 to D1269	Axis 3 monitor device	D1420 to D1429	Axis 19 monitor device
D1270 to D1279	Axis 4 monitor device	D1430 to D1439	Axis 20 monitor device
D1280 to D1289	Axis 5 monitor device	D1440 to D1449	Axis 21 monitor device
D1290 to D1299	Axis 6 monitor device	D1450 to D1459	Axis 22 monitor device
D1300 to D1309	Axis 7 monitor device	D1460 to D1469	Axis 23 monitor device
D1310 to D1319	Axis 8 monitor device	D1470 to D1479	Axis 24 monitor device
D1320 to D1329	Axis 9 monitor device	D1480 to D1489	Axis 25 monitor device
D1330 to D1339	Axis 10 monitor device	D1490 to D1499	Axis 26 monitor device
D1340 to D1349	Axis 11 monitor device	D1500 to D1509	Axis 27 monitor device
D1350 to D1359	Axis 12 monitor device	D1510 to D1519	Axis 28 monitor device
D1360 to D1369	Axis 13 monitor device	D1520 to D1529	Axis 29 monitor device
D1370 to D1379	Axis 14 monitor device	D1530 to D1539	Axis 30 monitor device
D1380 to D1389	Axis 15 monitor device	D1540 to D1549	Axis 31 monitor device
D1390 to D1399	Axis 16 monitor device	D1550 to D1559	Axis 32 monitor device

- Detailes of each axis

Device No.	Signal name
D1240 + 10n	Unusable
D1241 + 10n	Execute cam No.
D1242 + 10n D1243 + 10n	Execute stroke amount
D1244 + 10n	Current value within 1 cam shaft revolution
D1245 + 10n	
D1246 + 10n	Unusable
D1247 + 10n	
D1248 + 10n	
D1249 + 10n	

(Note-1) : "n" in the above device No. shows the numerical value which correspond to axis No.

Q173HCPU : Axis No.1 to No.32 (n=0 to 31)
Q172HCPU : Axis No.1 to No.8 (n=0 to 7)

(Note-2) : The unused aixs areas in the mechanical system program can be used as an user device.

6) Table of the common devices (SV13/SV22)

Device No.	Signal name		Signal derecrtion
D704	PLC ready flag request		Command device
D705	Speed switching point specified flag request		
D706	All axes servo ON command request		
D707	Real/virtual mode switching request (SV22)		
D708	JOG operation simultaneous start command request		
D709	Unusable		—
D710 to D713	JOG operation simultaneous start axis setting register		Command device
D714 D715	Manual pulse generator axis 1 No. setting register		
D716 D717	Manual pulse generator axis 2 No. setting register		
D718 D719	Manual pulse generator axis 3 No. setting register		
D720	Axis 1	Manual pulse generators 1 pulse input magnification setting register (Note-1,2)	
D721	Axis 2		
D722	Axis 3		
D723	Axis 4		
D724	Axis 5		
D725	Axis 6		
D726	Axis 7		
D727	Axis 8		
D728	Axis 9		
D729	Axis 10		
D730	Axis 11		
D731	Axis 12		
D732	Axis 13		
D733	Axis 14		
D734	Axis 15		
D735	Axis 16		
D736	Axis 17		
D737	Axis 18		
D738	Axis 19		
D739	Axis 20		
D740	Axis 21	Manual pulse generators 1 pulse input magnification setting register (Note-1,2)	Command device
D741	Axis 22		
D742	Axis 23		
D743	Axis 24		
D744	Axis 25		
D745	Axis 26		
D746	Axis 27		
D747	Axis 28		
D748	Axis 29		
D749	Axis 30		
D750	Axis 32		
D751	Axis 32		
D752	Manual pulse generator 1 smoothing magnification setting register		
D753	Manual pulse generator 2 smoothing magnification setting register		
D754	Manual pulse generator 3 smoothing magnification setting register		
D755	Manual pulse generator 1 enable flag request		
D756	Manual pulse generator 2 enable flag request		
D757	Manual pulse generator 3 enable flag request		
D758	Unusable		—
D759	PCPU ready complete flag status (0 : OFF/1 : ON)		Monitor device
D760 to D789	Unusable (30 points)		—
D790 D791	Real mode axis information register (SV22)		Monitor device
D792 to D799	Unusable (8 points)		—

(Note-1) : The range of axis No.1 to 8 is valid in the Q172HCPU.

(Note-2) : Device area of 9 axes or more is unusable in the Q172HCPU.

(2) Special relays

Special relays are internal relays whose applications are fixed in the Motion CPU. For this reason, they cannot be used in the same way as the normal internal relays by the Motion SFC programs.
However, they can be turned ON/OFF as needed in order to control the Motion CPU.

The headings in the table that follows have the following meanings.

Item	Explanation
No.	• Indicates the device No. of the special relay.
Name	• Indicates the name of the special relay.
Meaning	• Indicates the nature of the special relay.
Details	• Indicates detailed information about the nature of the special relay.
Set by (When set)	• Indicates whether the relay is set by the system or user, and, if it is set by system, when setting is performed. <Set by> S : Set by system (Motion CPU) U : Set by user (Motion SFC program or test operation using a peripheral device) S/U : Set by both system (Motion CPU) and user <When set> Indicated only if setting is done by system (Motion CPU) . Main process : Set during each main processing (free time processing of the CPU) Initial process : Set only during initial processing (when power supply is turned ON, or when executed the reset) Status change : Set only when there is a change in status Error : Set when error is occurred. Request : Set only when there is a user request (Special reray, etc.) Operation cycle : Set during each operation cycle of the Motion CPU.

Table 2.1 Special relay list

No.	Name	Meaning	Details	Set by (When set)	Remark
M9000	Fuse blown detection	OFF : Normal ON : Fuse blown module detected	• Turn on when there is one or more output modules control of self CPU which fuse has been blown. Remains on if normal status is restored.	S(Occur an error)	
M9005	AC/DC DOWN detection	OFF : AC/DC DOWN not detected ON : AC/DC DOWN detected	• Turn on if a momentary power interruption of less than 20[ms] occurred during use of the AC power supply module, and reset by turning power off to on. • Turn on if a momentary power interruption of less than 10[ms] occurred during use of the DC power supply module, and reset by turning power off to on.		
M9006	Battery low	OFF : Normal ON : Battery low	• Turned on when the voltage of the external battery reduces to less than specified value. Turn off when the voltage of the external battery becomes normal. • Synchronizes with "BAT. LED" • Check the voltage of the external battery, only when it is set with "external battery use" by system setting.		
M9007	Battery low latch	OFF : Normal ON : Battery low	• Turn on when the voltage of the external battery reduces to less than specified value. Remains on if normal status is restored. • Synchronizes with "BAT. LED" • Check the voltage of the external battery, only when it is set with "external battery use" by system setting.		
M9008	Self-diagnostic error	OFF : No error ON : Error	• Turn on when error is found as a result of self-diagnosis. Remains on if normal status is restored.		
M9010	Diagnostic error	OFF : No error ON : Error	• Turn on when error is found as a result of diagnosis. Remains on if normal status is restored.		
M9025	Clock data set request	OFF : Ignored ON : Set request present used	• Write clock data stored in D9025 to D9028 to the clock element when M9025 has changed from off to on.	U	
M9026	Clock data error	OFF : No error ON : Error	• Turn on by clock data (D9025 to D9028) error.	S(Request)	
M9028	Clock data read request	OFF : Ignored ON : Read request	• Read clock data from D9025 to D9028 in BCD when M9028 is on.	U	
M9036	Always ON	ON ——— OFF	• Turn on without regard to position of RUN/STOP switch on.	S(Main processing)	
M9037	Always OFF	ON OFF ———	• Turn off without regard to position of RUN/STOP switch on.		
M9060	Error reset	OFF → ON : Error reset	• A release of the error is executed.	U	
M9073	PCPU WDT error flag	ON : Abnormal OFF : Normal	• Turn on when a "watchdog timer error" is detected by the Motion CPU self-diagnosis function. When the Motion CPU detects a WDT error, it executes an immediate stop without deceleration of the operating axes. • The error cause is stored in the "Motion CPU WDT error cause (D9184)".	S(Occur an error)	
M9074	PCPU READY complete flag	ON : PCPU READY completion OFF : PCPU READY uncompletion	• When the PLC ready flag (M2000) turn off to on, the fixed parameters, servo parameters and limit switch output data, etc., are checked, and if no error is detected this flag turns on. • Turn off when the PLC ready flag (M2000) turns off.	S(Request)	
M9075	Test mode ON flag	ON : TEST mode is in effect. OFF : TEST mode is not in effect.	• This flag status indicates whether a TEST mode established from a peripheral device is currently in effect. • If the TEST mode is not established in response to a TEST mode request from a peripheral device, the "TEST mode request error flag (M9078)" will turn on.	S(Request)	
M9076	External forced stop input flag	ON : Forced stop OFF OFF : Forced stop ON	• This flag status indicate whether the forced stop.	S(Operation cycle)	

Table 2.1 Special relay list (continued)

No.	Name	Meaning	Details	Set by (When set)	Remark
M9077	Manual pulse generator axis setting error flag	ON : At least one D714 to D719 setting is abnormal. OFF : All D714 to D719 settings are normal.	• This flag indicates whether the setting designated at the manual pulse generator axis setting register (D714 to D719) is normal or abnormal. • When this relay turns on, the error content is stored at the manual pulse generator axis setting error register (D9185 to D9187).	S(Occur an error)	
M9078	TEST mode request error flag	ON : Abnormal OFF : Normal	• Turn on if the TEST mode is not established in response to a TEST mode request from a peripheral device. • When this relay turns on, the error content is stored at the TEST mode request error register (D9182 to D9183).	S(Occur an error)	
M9079	Servo program setting error flag	ON : Abnormal OFF : Normal	• This flag status indicates whether the positioning data of the servo program(K) specified with the Motion SFC program is normal or abnormal, and if error is detected this flag turns on. • The content of a servo program setting error is stored at D9189 and D9190.	S(Occur an error)	
M9216	CPU No.1 MULTR complete flag	OFF to ON : CPU No.1 read completion	• Turn on when the data read from CPU No.1 is performed normally by MULTR instruction.	S(Read completion)	
M9217	CPU No.2 MULTR complete flag	OFF to ON : CPU No.2 read completion	• Turn on when the data read from CPU No.2 is performed normally by MULTR instruction.		
M9218	CPU No.3 MULTR complete flag	OFF to ON : CPU No.3 read completion	• Turn on when the data read from CPU No.3 is performed normally by MULTR instruction.		
M9219	CPU No.4 MULTR complete flag	OFF to ON : CPU No.4 read completion	• Turn on when the data read from CPU No.4 is performed normally by MULTR instruction.		
M9240	CPU No.1 reset flag	OFF : CPU No.1 reset release ON : CPU No.1 resetting	• Turn off at reset release of the CPU No.1. • Turn on during reset of the CPU No.1. (It also contains when a CPU is removed from the base unit.) • The other CPU is also resetting.	S(Change status)	
M9241	CPU No.2 reset flag	OFF : CPU No.2 reset release ON : CPU No.2 resetting	• Turn off at reset release of the CPU No.2. • Turn on during reset of the CPU No.2. (It also contains when a CPU is removed from the base unit.) • The error of the "MULTI CPU DOWN" (error code : 7000) occurs in the other CPU.		
M9242	CPU No.3 reset flag	OFF : CPU No.3 reset release ON : CPU No.3 resetting	• Turn off at reset release of the CPU No.3. • Turn on during reset of the CPU No.3. (It also contains when a CPU is removed from the base unit.) • The error of the "MULTI CPU DOWN" (error code : 7000) occurs in the other CPU.		
M9243	CPU No.4 reset flag	OFF : CPU No.4 reset release ON : CPU No.4 resetting	• Turn off at reset release of the CPU No.4. • Turn on during reset of the CPU No.4. (It also contains when a CPU is removed from the base unit.) • The error of the "MULTI CPU DOWN" (error code : 7000) occurs in the other CPU.		
M9244	CPU No.1 error flag	OFF : CPU No.1 normal ON : On CPU No.1 stop error	• Turn off when the CPU No.1 is normal. (It contains at continuation error.) • Turn on during stop error of the CPU No.1. (Note-1)		
M9245	CPU No.2 error flag	OFF : CPU No.2 normal ON : On CPU No.2 stop error	• Turn off when the CPU No.2 is normal. (It contains at continuation error.) • Turn on during stop error of the CPU No.2. (Note-1)		
M9246	CPU No.3 error flag	OFF : CPU No.3 normal ON : On CPU No.3 stop error	• Turn off when the CPU No.3 is normal. (It contains at continuation error.) • Turn on during stop error of the CPU No.3. (Note-1)		
M9247	CPU No.4 error flag	OFF : CPU No.4 normal ON : On CPU No.4 stop error	• Turn off when the CPU No.4 is normal. (It contains at continuation error.) • Turn on during stop error of the CPU No.4. (Note-1)		

(Note-1): The CPU No.1 is reset after the factor of the stop error is removed to cancel a stop error. → Resetting is cancelled.

(3) Special registers

Special registers are internal registers whose applications are fixed in the Motion CPU. For this reason, it is not possible to use these registers in Motion SFC programs in the same way that normal registers are used.
However, data can be written as needed in order to control the Motion CPU.
Data stored in the special registers are stored as BIN values if no special designation has been made to the contrary.

The headings in the table that follows have the following meanings.

Item	Explanation
Number	• Indicates the No. of the special register.
Name	• Indicates the name of the special register.
Meaning	• Indicates the nature of the special register.
Details	• Indicates detailed information about the nature of the special register.
Set by (When set)	• Indicates whether the register is set by the system or user, and, if it is set by system, when setting is performed. <Set by> S : Set by system (Motion CPU) U : Set by user (Motion SFC program or test operation using a peripheral device) S/U : Set by both system (Motion CPU) and user <When set> Indicated only if setting is done by system (Motion CPU) . Main process : Set during each main processing (free time processing of the CPU) Initial process : Set only during initial processing (when power supply is turned ON, or when executed the reset) Status change : Set only when there is a change in status Error : Set when error is occurred. Request : Set only when there is a user request (Special reray, etc.) Operation cycle : Set during each operation cycle of the Motion CPU.

Table 2.2 Special register list

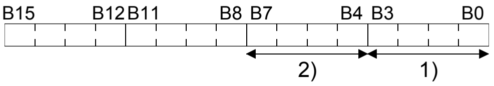
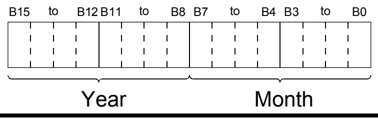
No.	Name	Meaning	Details	Set by (When set)	Remark
D9000	Fuse blown No.	Module No. with blown fuse	• When fuse blown modules are detected, the lowest I/O module No. is stored in D9000.	S(Occur an error)	
D9005	AC/DC DOWN counter No.	Number of times for AC/DC DOWN	• 1 is added to the stored value each time the input voltage becomes 85[%] (AC power supply/65[%] DC power supply) or less of the rating while the CPU module is performing an operation, and the value is stored in BIN code.		
D9008	Diagnostic error	Dignostic error number	• When error is found as a result of self-diagnosis, error No. is stored in BIN code. • Refer to "2.4 Multiple CPU Error Codes" of the "Q173HCPU/Q172HCPU Motion controller Programming Manual (COMMON)" for details of the error code.		
D9010	Diagnostic error occurrence time	Diagnostic error occurrence time	• The age (A.D, the rightmost two digits) when data on D9008 are updated, and the month stored with a BCD code two digits. B15 to B8 B7 to B0 Year(0 to 99) | Month(1 to 12) Example : October 1995 H9510		
D9011			• The day when data on D9008 are updated, and the hour stored with a BCD code two digits. B15 to B8 B7 to B0 Day(1 to 31) | Hour(0 to 23) Example : 25st, 10 a.m H2510		
D9012			• The minute when data on D9008 are updated, and the second stored with a BCD code two digits. B15 to B8 B7 to B0 Minute(0 to 59) | Second(0 to 59) Example : 35 min., 48 sec. H3548		
D9013	Error information classfication	Error information classfication code	• The classification code to judge the error information stored in the error information (D9014) is stored. • The following codes are stored. 0: None 1: Module No./CPU No./Base No. 2: Parameter No.		
D9014	Error information	Error information	• Error information to comply with the diagnostic error (D9008) is stored. There are following two types information to be stored. 1) Module No./CPU No./Base No. • Module No. or CPU No. is stored according to the error which occurred in the case of the Multiple CPU system. (Refer to each error code which is stored.) CPU No.1: 1, CPU No.2: 2, CPU No.3: 3, CPU No.4: 4 2) Parameter No.		
D9015	Operating state of CPU	Operating state of CPU	• The operation states of CPU as shown below are stored in D9015.  1) Operating state of CPU: 0: RUN, 2: STOP 2) STOP cause: 0: RUN/STOP switch, 4: Error Note: Priority is earliest first	S(Main processing)	
D9017	Scan time	Scan time (1ms units)	• Main cycle is stored in the unit 1ms. • Setting range (0 to 65535[ms])		
D9019	Maximum scan time	Maximum scan time (1ms units)	• The maximum value of the main cycle is stored in the unit 1ms. • Setting range (0 to 65535[ms])		
D9025	Clock data	Clock data (Year, month)	• Stores the year (2 lower digits) and month in BCD.  Example : July, 1993 H9307	S/U(Request)	

Table 2.2 Special register list (continued)

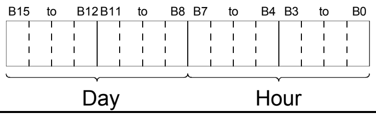
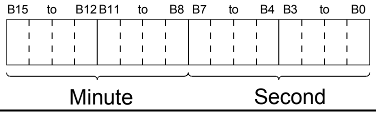
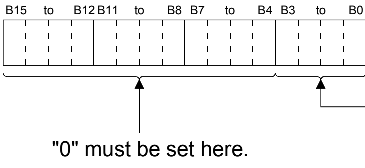
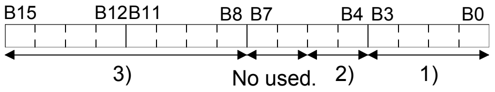
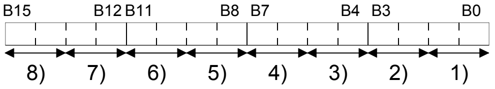
No.	Name	Meaning	Details	Set by (When set)	Remark
D9026	Clock data	Clock data (Day, hour)	• Stores the day and hour in BCD.  Example : 31st, 10 a.m. H3110	S/U(Request)	
D9027		Clock data (Minute, second)	• Stores the minute and second in BCD.  Example : 35 min., 48 sec. H3548		
D9028		Clock data (Day of week)	• Stores the day of the week in BCD.  Example : Friday H0005 Day of week 0 Sunday 1 Monday 2 Tuesday 3 Wednesday 4 Thursday 5 Friday 6 Saturday		
D9060	Error reset	Error No. of releasing an error	• Error No. of canceling error is stored.	U	
D9061	Multiple CPU No.	Multiple CPU No.	• CPU No. of the self CPU is stored.	S(Initial processing)	
D9112	Connect/ disconnect	Connect/ disconnect of SSCNET	• When the servo amplifier or SSCNETⅢ cable on the SSCNET system are exchanged or re-connected, an user side requires connect/disconnect to a system, and a system side stores the state of waiting for command accept or execution of connect/disconnect.	—	
			0 : Connect/disconnect command accept waiting -1 : Connect/disconnect execute waiting	S (Main processing)	
			1 to 32 : Disconnect command -10 : Connect command -2 : Connect/disconnect execute command	U	
D9182 D9183	Test mode request error	It is operating in requirement error occurrence of the test mode, axis information	• Each axis is stopping: 0/Operating: 1, information is stored as a bit data. D9182: b0 to b15 (Axis 1 to Axis 16) D9183: b0 to b15 (Axis 17 to Axis 32)	S(Occur an error)	
D9184	Motion CPU WDT error cause	Error meaning of WDT error occures	The following error codes are stored in D9184. 1: S/W fault 1 2: Operation cycle over 3: Q bus WDT error 4: WDT error 30: Information processor H/W error 201 to 215: Q bus H/W fault 250 to 253: Servo amplifier interface H/W fault 300: S/W fault3 301: 15 CPSTART instructions of 8 or more points were started simultaneously.		
D9185 D9186 D9187	Manual pulse generator axis setting error	Manual pulse generator axis setting error information	• Contents of the manual pulse generator axis setting error is stored when the manual pulse generator axis setting error flag (M9077) turn on. (Normal: 0/Setting error: 1) D9185: The manual pulse generator axis setting error is stored in b0 to b2 (P1 to P3). The smoothing magnification setting is stored in b3 to b5 (P1 to P3). D9186: One pulse input magnification setting error is stored in b0 to b15 (axis 1 to axis 16). D9187: One pulse input magnification setting error is stored in b0 to b15 (axis 17 to axis 32).		

Table 2.2 Special register list (continued)

No.	Name	Meaning	Details	Set by (When set)	Remark
D9188	Motion operation cycle	Motion operation cycle	• The time when the motion operation cycle is stored in the [μs] unit.	S(Operation cycle)	
D9189	Error program No.	Error program No. of servo program	When the servo program setting error flag (M9079) turns on, the erroneous servo program No. will be stored.	S(Occur an error)	
D9190	Error item information	Error code of servo program	When the servo program setting error flag (M9079) turns on, the error code corresponding to the erroneous setting item will be stored.		
D9191 D9192	Servo amplifier loading information	Servo amplifier loading information	• The loading status (loading: 1/non-loading: 0) of the servo amplifier checked in initial process, and stored as the bit data. D9191: b0 to b15 (axis 1 to axis 16) D9192: b0 to b15 (axis 17 to axis 32) • The axis which turned from non-loading to loading status after power-on is handled as loaded. (However, the axis which turned from loading to non-loading status remains as loaded.)	S(Initial processing)	
D9193 D9194 D9195	Real/virtual mode switching error information	Real/virtual mode Switching error code	• When a mode switching error occurs in real-to-virtual or virtual-to-real mode switching, or a mode continuation error occurs in the virtual mode, its error information is stored.	S(Occur an error)	
D9196	PC link communication error codes	PC link communication error codes	• The following error code is stored. 00: No error 01: Receiving timing error 02: CRC error 03: Communication response code error 04: Received frame error 05: Communication task start error (Each error code is reset to "00" when normal communication is restarted.)		
D9197	Operation cycle of the Motion CPU setting	Operation cycle of the Motion CPU setting	• The time when the setting operation cycle is stored in the [μs] unit.	S(Initial processing)	
D9200	State of switch	State of CPU switch	• The CPU switch status is stored in the following format.  1) CPU switch status 0: RUN 1: STOP 2: L.CLR 2) Memory card switch Always OFF 3) Dip switch B8 through B12 correspond to SW1 through SW5 of system setting switch 1. 0: OFF/1: ON B13 through B15 is not used.	S(Main processing)	
D9201	State of LED	State of CPU-LED	• Information concerning which of the following states the LEDs on the CPU are in is stored in the following bit patterns. • 0 is off, 1 is on, and 2 is flicker  1): RUN 5): BOOT 2): ERROR 6): No used 3): M.RUN 7): No used 4): BAT.ALARM 8): MODE Bit patterns for MODE 0: OFF 1: Green 2: Orange	S(Change status)	

MEMO

2. STRUCTURE OF THE MOTION CPU PROGRAM

Motion CPU programs is created in the Motion SFC of flowchart format. The motion control of servomotors is performed using the real-mode servo programs specified by motion-control steps in a Motion SFC program in SV13/SV22 real mode. Virtual servomotors in a mechanical system program are controlled using the virtual mode servo programs specified by motion-control steps so as to enable synchronous control in SV22 virtual mode. Refer to the documents below for the details of Motion SFC programs, motion control in real mode, and motion control in virtual mode.

Item	Reference manual
Motion SFC program	Section 4 in this manual
Motion control in SV13/SV22 real mode (Servo program)	Q173HCPU/Q172HCPU Motion controller (SV13/SV22) Programming Manual (REAL MODE)
Motion control in SV22 virtual mode (Mechanical system program)	Q173HCPU/Q172HCPU Motion controller (SV22) Programming Manual (VIRTUAL MODE)

2.1 Motion Control in SV13/SV22 Real Mode

(1) System with servomotor is controlled directly using the servo program in (SV13/SV22) real mode.

(2) Setting of the positioning parameter and creation of the servo program/ Motion SFC program are required.

(3) The procedure of positioning control is shown below:

1) Motion SFC program is requested to start using the S(P).SFCS instruction of the PLC program.
(Motion SFC program can also be started automatically by parameter setting.)

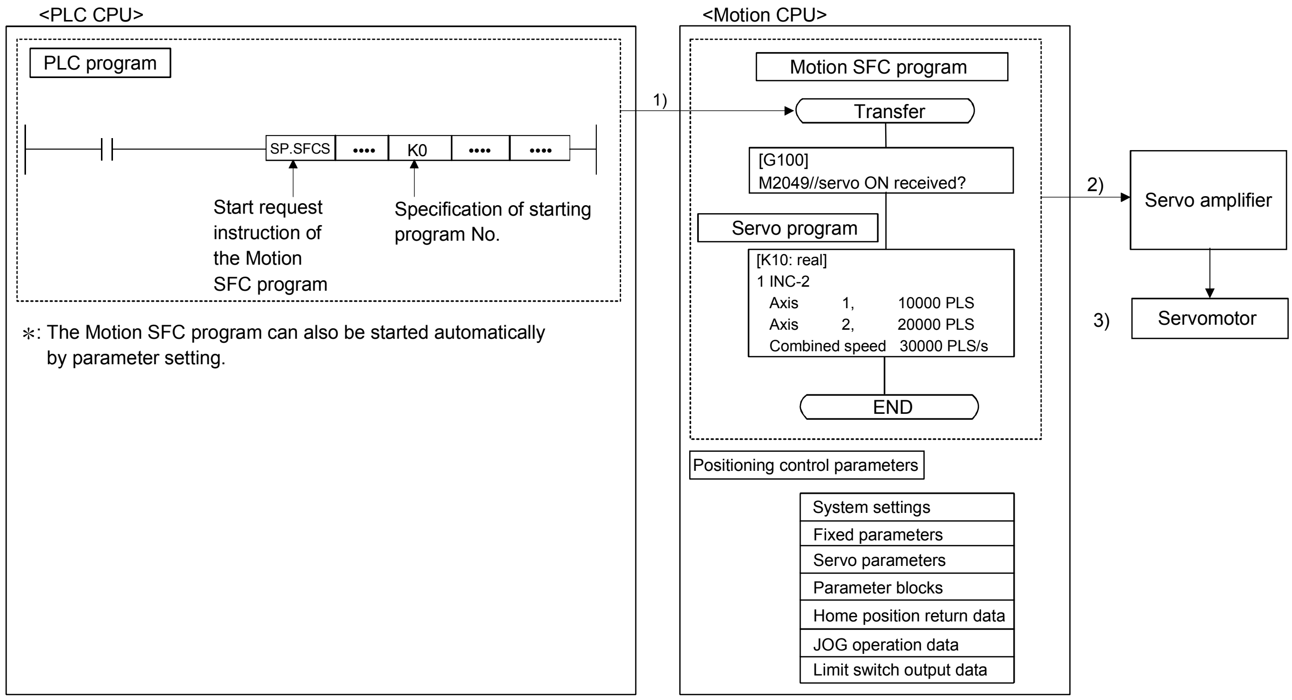
↓

2) Execute the positioning control using the specified the Motion SFC program.
(Output to the servo amplifier)

↓

3) The servomotor is controlled.

Program structure in SV13/SV22 real mode



2.2 Motion Control in SV22 Virtual Mode

(1) Software-based synchronous control is performed using the mechanical system program constructed by virtual main shaft and mechanical module in (SV22) virtual mode.

(2) Mechanical system programs is required in addition to the positioning parameter, servo program/Motion SFC program used in real mode.

(3) The procedure of positioning control in virtual model is shown below:

1) Motion SFC program for virtual mode is requested to start using the S(P).SFCS instruction of the PLC program.
(Motion SFC program can also be started automatically by parameter setting.)

↓

2) The virtual servomotor in the mechanical system program is started.

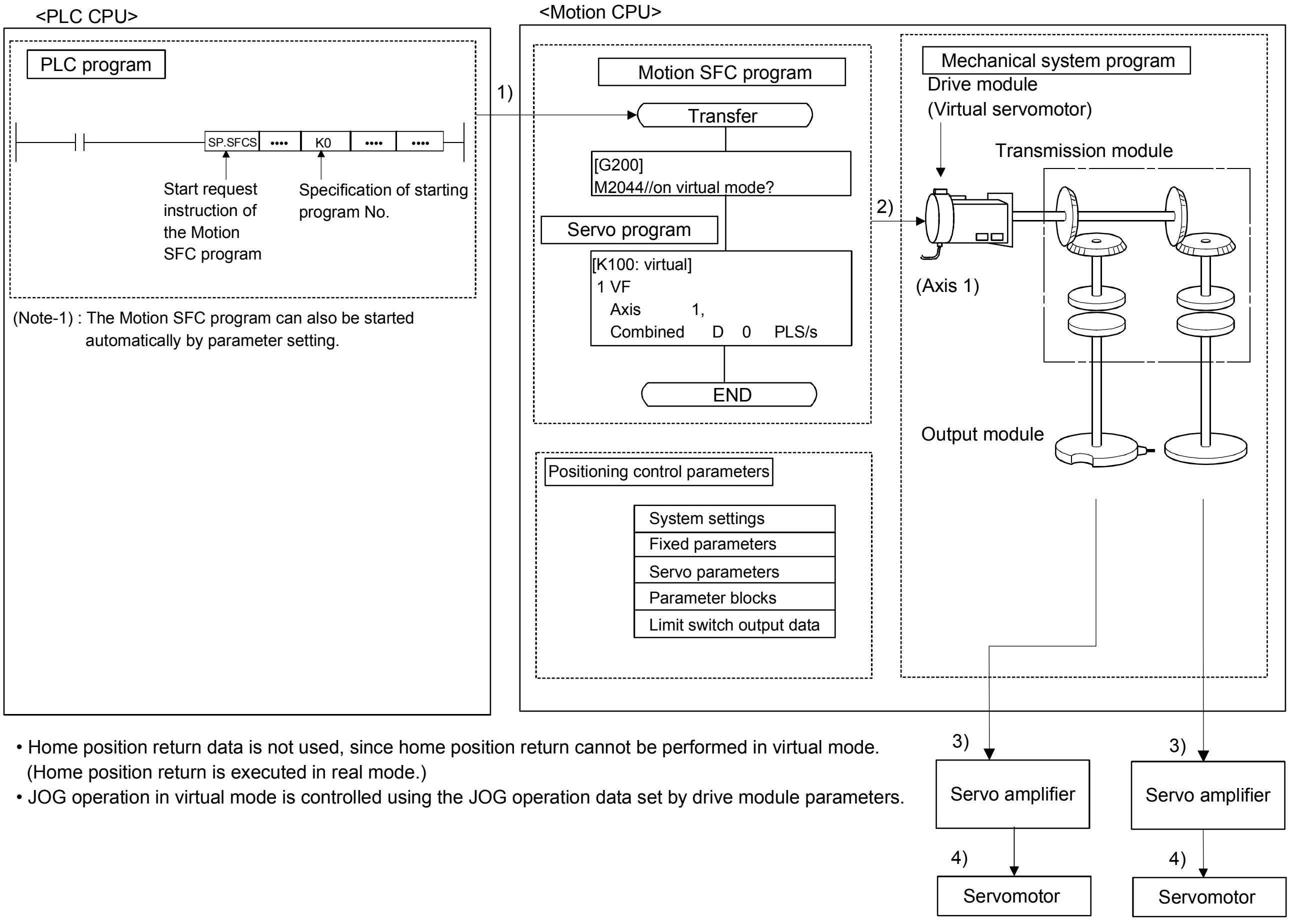
↓

3) Output the operation result obtained through the transmission module to the servo amplifier set as the output module.

↓

4) The servomotor is controlled.

Program structure in SV22 virtual mode



- Home position return data is not used, since home position return cannot be performed in virtual mode. (Home position return is executed in real mode.)
- JOG operation in virtual mode is controlled using the JOG operation data set by drive module parameters.

MEMO

3. MOTION DEDICATED PLC INSTRUCTION

3.1 Motion Dedicated PLC Instruction

(1) The Motion dedicated PLC instruction which can be executed toward the Motion CPU which installed a SV13/SV22 operating system software for the Motion SFC is shown below.

Instruction	Description
S(P).SFCS	Start request of the Motion SFC program (Program No. may be specified.)
S(P).SVST	Start request of the specified servo program
S(P).CHGA	Current value change request of the specified axis
S(P).CHGV	Speed change request of the specified axis
S(P).CHGT	Torque control value change request of the specified axis
S(P).DDWR	Write from the PLC CPU to the Motion CPU
S(P).DDRD	Read from the devices of the Motion CPU
S(P).GINT	Execute request of an event task of Motion SFC program

(Note) : As for the details of each instruction, it explains after the next section.

3.1.1 Restriction item of the Motion dedicated PLC instruction

(1) To self CPU high speed interrupt accept flag from CPUn.
Common precautions of the Motion dedicated PLC instruction as shown below.

(a) To self CPU high speed interrupt accept flag from CPUn is shown in the following table.
To self CPU high speed interrupt accept flag from CPUn is "No operation" even if the instruction is executed when it is cannot be accepted.
When the Motion dedicated PLC instruction is accepted in the Motion CPU, to self CPU high speed interrupt accept flag from CPUn of the self CPU (Motion CPU) shared CPU memory cannot be accepted and processing toward the instruction for requirement.
When processing is completed and it becomes the condition that it has an instruction accepted, to self CPU high speed interrupt accept flag from CPUn can be accepted.

Shared CPU memory address ( ) is decimal address	Description	Example of the reading (When target is the CPU No.2)
30H(48)	The lowest rank bit (30H(48)) toward executing instruction from CPU No.1.	U3E1/G48.0
31H(49)	The lowest rank bit (31H(49)) toward executing instruction from CPU No.2.	U3E1/G49.0
32H(50)	The lowest rank bit (32H(50)) toward executing instruction from CPU No.3.	U3E1/G50.0
33H(51)	The lowest rank bit (33H(51)) toward executing instruction from CPU No.4.	U3E1/G51.0

(b) "To self CPU high speed interrupt accept flag from CPUn" turn ON/OFF at the executing instruction, when the Multiple CPU dedicated instructions are executed to the same CPU from one PLC CPU.
Therefore, when each instruction is executed only once at approval the executing condition, it is necessary to take an interlock by internal relay (M10) and so on besides "To self CPU high speed interrupt accept flag from CPUn".

(2) Execution of the Motion dedicated PLC instruction

(a) Motion dedicated PLC instruction can be executed with fixed cycle execute type PLC and interrupt PLC. However, as for a complete device, the program turned on according to fixed cycle executed type PLC and program type (scan or low speed) executed interrupt PLC is different.

(b) One Motion CPU can be accepted max.32 instructions simultaneously from multiple other CPUs. (Except S(P).GINT instruction.) If 33 instructions or more are executed Motion CPU returns the complete status[4C08] error.
As Motion CPU can be accepted up to 32 instructions, number of acceptable instructions changes according to number of CPUs included Motion CPU. Calculation expression is shown below.

(Number of maximum acceptable instructions per one Motion CPU) =
32 – ((Number of all CPUs) – 2) [Number of instructions]

(c) Local devices and file registers as program are written to device by END processing. Do not use the devices below.

- Each instruction complete device
- D1 of S(P).DDRD instruction (The first device of the self CPU which stored the reading data.)

(d) Use a flag in the shared CPU memory which correspond with each instruction not to execute multiple instructions to the same shaft of the Motion CPU of same CPU No. for the interlock condition.
(Program example 1)

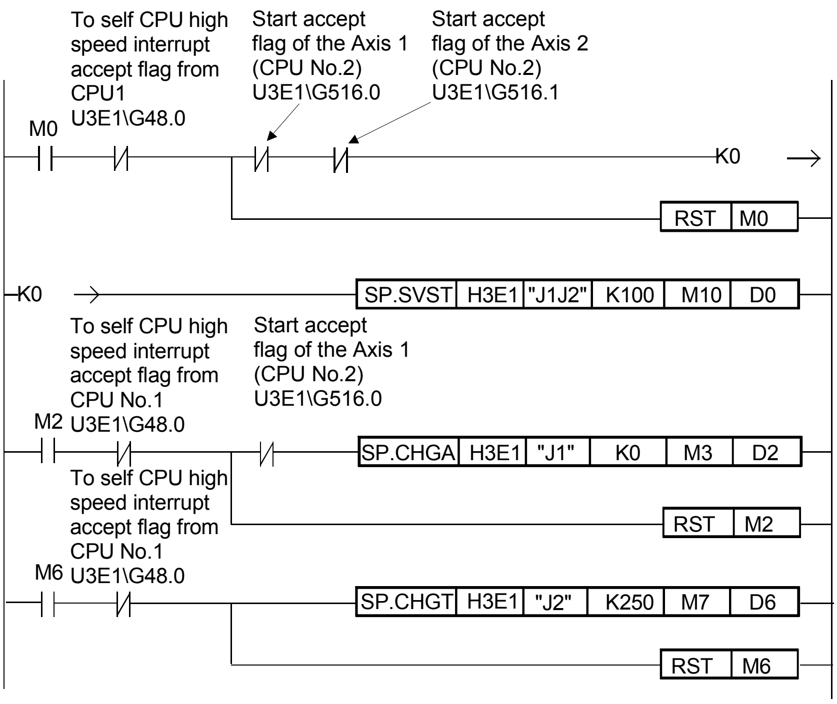
(e) S(P).SFCS/S(P).SVST/S(P).CHGA/S(P).CHGVS(P).CHGT/S(P).DDWR/ S(P).DDRD instructions cannot be executed simultaneously. Therefore, it is necessary to take an interlock by to self CPU high speed interrupt accept flag from CPUn.
One PLC CPU can be executed max.32 Motion dedicated PLC instructions simultaneously using to self CPU high speed interrupt accept flag from CPUn. (Except S(P).GINT instruction.)
If 33 instructions or more are executed, the PLC CPU returns the OPERATION ERROR [4107].

(f) When multiple Motion dedicated PLC instructions are directly executed because one contact-point turns on, an instruction may not be executed.
In this case, create a program with reference to program example.
(Program example 2)

(g) When the Motion dedicated function of the operation control step (Fn/FSn) and Motion control program (Kn) in Motion CPU. Since there is no flag which can be distinguished on instruction execution in the PLC CPU, it is necessary to taken an interlock by user program.
(Program example 3)

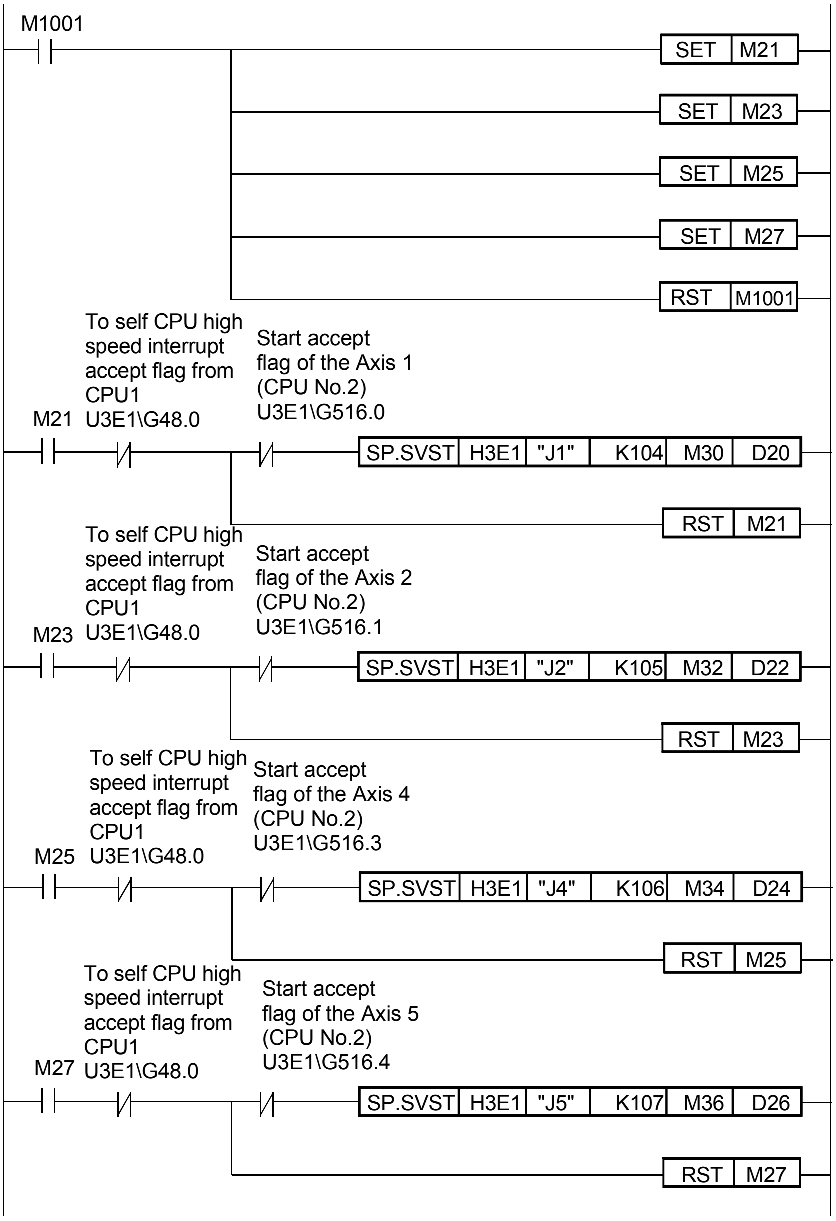
<Program example 1>

Program which executes multiple instructions to the same shaft of the Motion CPU of same CPU No..



<Program example 2>

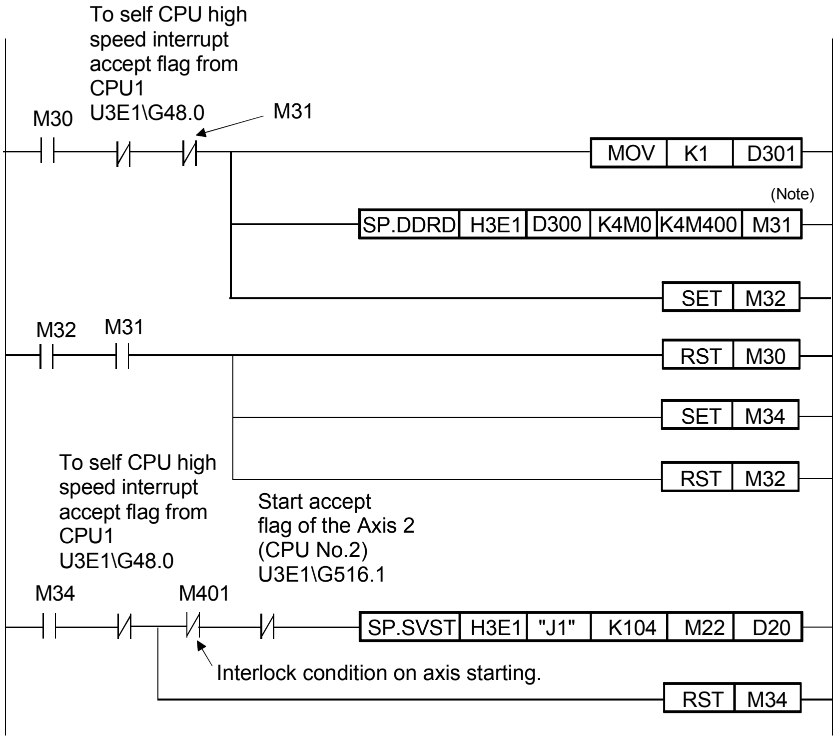
Program which executes directly multiple Motion dedicated PLC instructions because one contact-point turns on.



<Program example 3>

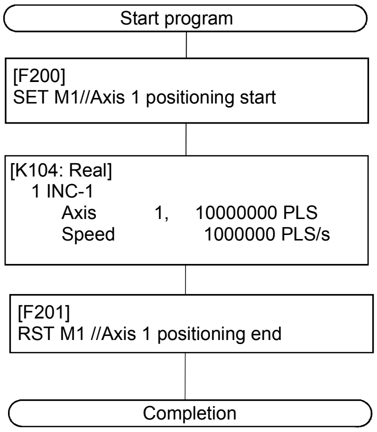
Program which executes the Motion dedicated function of the operation control step (Fn/FSn) and the motion control program (Kn).

- PLC CPU side



(Note) : 4 points worth of the data from "M0" of the CPU No.2 are stored after M400 by S(P).DDRD instruction.

- Motion CPU side



POINT

Access from the PLC CPU is processed before the communication processing of the Motion CPU. Therefore, if the Motion dedicated PLC instruction is frequently performed from the PLC CPU, the scan time of the PLC CPU is not only prolonged, but delay will arise in the communication processing of the Motion CPU.

Perform execution of the Motion dedicated PLC instruction from the PLC CPU by S(P).DDWR/S(P).DDRD/S(P).CHGV instruction etc. only at the time of necessity.

(3) Complete status

The error code is stored in the complete status at abnormal completion of the Multiple CPU dedicated instruction. The error code which is stored is shown below. (The error code marked " * " is dedicated with the Motion CPU.)

Complete status (Error code)(H)	Error factor	Corrective action
0	Normal completion	Confirm a program, and correct it to a correct PLC program.
4C00 *	The specified device cannot be used in the Motion CPU. Or, it is outside the device range.	
4C01 *	The instruction for the Multiple CPU system which did not be correspond with operating system software of the Motion CPU was executed.	
4C02 *	The Motion SFC program No. to start is outside the range 0 to 255.	
4C03 *	The servo program No. to execute is outside the range 0 to 4095.	
4C04 *	Axis No. set by SVST instruction is injustice.	
4C05 *	Axis No. set by CHGA instruction is injustice.	
4C06 *	Axis No. set by CHGV instruction is injustice.	
4C07 *	Axis No. set by CHGT instruction is injustice.	
4C08 *	• When using the S(P).SFCS/S(P).SVST/S(P).CHGA instruction. • There are 33 or more instruction requests to the Motion CPU from the PLC CPU in S(P).SFCS/ S(P).SVST/S(P).CHGA/S(P).GINT sum table simultaneously, and the Motion CPU cannot process them. • When using the S(P).DDRD/S(P).DDWR instruction. There are 33 or more instruction requests to the Motion CPU from the PLC CPU in S(P).DDRD/S(P).DDWR sum table simultaneously, and the Motion CPU cannot process them.	
4C09 *	CPU No. of the instruction cause is injustice.	
4C0A *	Data error (The instruction which cannot be decoded in the Motion CPU was specified.)	
4C80	H/W error of the target CPU	
4C81		
4C83		
4C84		
4C90	Number over of execute instructions of the target CPU. There are 33 or more instruction requests to the Motion CPU from the PLC CPU in S(P).SFCS, S(P).SVST, S(P).CHGA, S(P).SHGV, S(P).CHGT, S(P).DDRD and S(P).DDWD sum table simultaneously, and the Motion CPU cannot process them.	

(4) Self CPU operation data area used by Motion dedicated instruction (30H to 33H)

The complete status of the to self CPU high speed interrupt accept flag from CPUn is stored in the following address.

Shared CPU memory address	Name	Description
30H(48)	To self CPU high speed interrupt accept flag from CPU1	This area is used to check whether to self CPU high speed interrupt accept flag from CPUn can be accepted or not. 0: To self CPU high speed interrupt accept flag from CPUn accept usable. 1: To self CPU high speed interrupt accept flag from CPUn accept disable.
31H(49)	To self CPU high speed interrupt accept flag from CPU2	
32H(50)	To self CPU high speed interrupt accept flag from CPU3	
33H(51)	To self CPU high speed interrupt accept flag from CPU4	

(5) System area used by Motion dedicated instruction (204H to 20DH)
The complete status of the each flag is stored in the following address.

Shared CPU memory address	Name	Description
204H(516)	Start accept flag (Axis1 to 16)	The start accept flag is stored by the 1 to 32 axis, each bit. (As for a bit's actually being set Q173HCPU : J1 to J32/ Q172HCPU : J1 to J8.) OFF : Start accept flag usable ON : Start accept flag disable b15 b1 b0 204H(516) address J16 •••••••• J2 J1 205H(517) address J32 •••••••• J17
205H(517)	Start accept flag (Axis17 to 32)	
206H(518)	Speed changing flag (Axis1 to 16)	The speed changing flag is stored by the 1 to 32 axis, each bit. (As for a bit's actually being set Q173HCPU : J1 to J32/ Q172HCPU : J1 to J8.) OFF : Start accept usable ON : Start accept disable b15 b1 b0 206H(518) address J16 •••••••• J2 J1 207H(519) address J32 •••••••• J17
207H(519)	Speed changing flag (Axis17 to 32)	
208H(520)	Synchronous encoder current value changing flag (Axis1 to 12) (Note-1)	The synchronous encoder current value change flag is stored by the 1 to 16 axis, each bit. (As for a bit's actually being set Q173HCPU : E1 to E12/ Q172HCPU : E1 to E8.) OFF : Start accept usable ON : Start accept disable b15 b1 b0 208H(520) address E16 •••••••• E2 E1
20CH(524)	Cam axis within-one-revolution current value changing flag (Axis1 to 16) (Note-1)	The cam axis within-one-revolution current value changing flag is stored by the 1 to 32 axis, each bit. (As for a bit's actually being set Q173HCPU : C1 to C32/ Q172HCPU : C1 to C8.) OFF : Start accept usable ON : Start accept disable b15 b1 b0 20CH(524) address C16 •••••••• C2 C1 20DH(525) address C32 •••••••• C17
20DH(525)	Cam axis within-one-revolution current value changing flag (Axis17 to 32) (Note-1)	

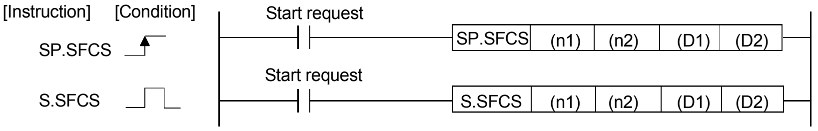
(Note-1) : It can be used in SV22.

3.2 Motion SFC Start Request from The PLC CPU to The Motion CPU: S(P).SFCS (PLC instruction: S(P).SFCS)

Setting data (Note)	Usable devices										
	Internal devices (System, User)		File register	Bit digit specified	Indirectly specified device	MELSECNET/10 direct J□\□		Special function module U□\G□	Index register Z□	Constant K, H	Other
	Bit	Word				Bit	Word				
(n1)		○	○	○	○					○	
(n2)		○	○	○	○					○	
(D1)	○	○	○								
(D2)		○	○		○						

○ : Usable △ : Usable partly

(Note) : Setting data (n1) to (D2) : Index qualification possible



[Setting data]

Setting data	Description	Data type
(n1)	(First I/O No. of the target CPU)/16 Value to specify actually is the following. (Note-1) CPU No.2 : 3E1H, CPU No.3 : 3E2H, CPU No.4 : 3E3H	16-bit binary
(n2)	Motion SFC program No. to start.	16-bit binary
(D1)	Complete devices (D1+0) : Device which make turn on for one scan at accept completion of instruction. (D1+1) : Device which make turn on for one scan at accept abnormal completion of instruction. ("D1+0" also turns on at the abnormal completion.)	Bit
(D2)	Device to store the complete status.	16-bit binary

(Note-1) : Motion CPU cannot used CPU No.1 in the Multiple CPU configuration.

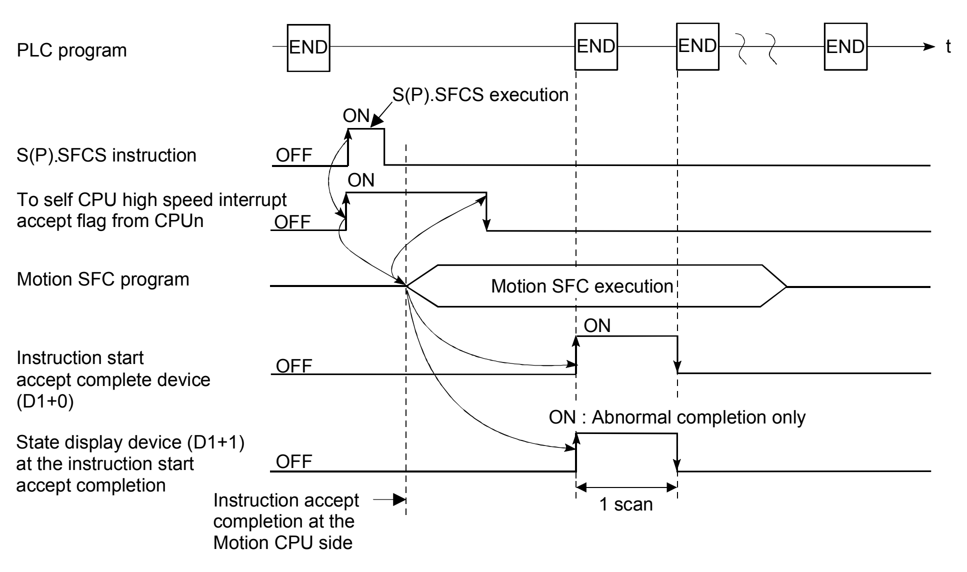
(1) This instruction is dedicated instruction toward the Motion CPU in the Multiple CPU system. Errors occurs when it was executed toward the CPU except the Motion CPU.

(2) Request to start the Motion SFC program of the program No. specified with (n2). The Motion SFC program can start any task setting of the normal task, event task and NMI task.

(3) This instruction is always effective regardless of the state of real mode/virtual mode/mode switching when the operating system software of Motion CPU is SV22.

(4) S(P).SFCS/S(P).SVST/S(P).CHGA/S(P).CHGV/S(P).CHGT/S(P).DDRD/ S(P).DDWR cannot be executed simultaneously toward the CPU executing S(P).SFCS instruction.
When the Motion dedicated PLC instruction is started continuously, it is necessary to execute the next instruction after the complete device of executing instruction turns on.

[Operation]



[Errors]

The abnormal completion in the case shown below, and the error code is stored in the device specified with the complete status storing device (D2).

Complete status (Note) (Error code)(H)	Error factor	Corrective action
4C00	The specified device cannot be used in the Motion CPU. Or, it is outside the device range.	Confirm a program, and correct it to a correct PLC program.
4C01	The instruction for the Multiple CPU system which did not be correspond with operating system software of the Motion CPU was executed.	
4C02	The Motion SFC program No. to start is outside the range 0 to 255.	
4C08	There are 33 or more instruction requests to the Motion CPU from the PLC CPU in S(P).SFCS/S(P).SVST/S(P).CHGA/ S(P).GINT sum table simultaneously, and the Motion CPU cannot process them.	
4C09	CPU No. of the instruction cause is injustice.	

(Note) : 0000H (Normal)

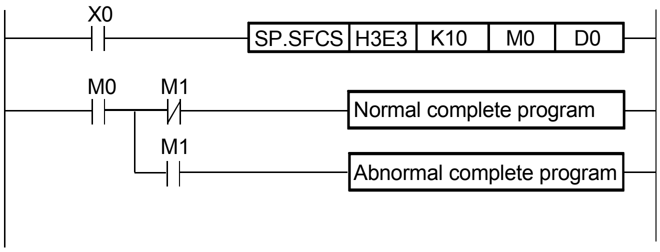
The error flag (SM0) is turned on an operation error in the case shown below, and an error code is stored in SD0.

Error code (Note)	Error factor	Corrective action
2110	The CPU No. to be set by "(First I/O No. of the target CPU)/16" is specified.	Confirm a program, and correct it to a correct PLC program.
2114	The self CPU by "(First I/O No. of the target CPU)/16" is specified.	
2117	The CPU except the Motion CPU by "(First I/O No. of the target CPU)/16" is specified.	
4002	Specified instruction is wrong.	
4004	The instruction is composed of devices except usable devices.	
4100	Since 0 to 3DFH, 3E4H is specified by "(First I/O No. of the target CPU)/16" is specified.	

(Note) : 0000H (Normal)

[Program example]

This program starts the Motion SFC program No.10 of the Motion CPU No.4.

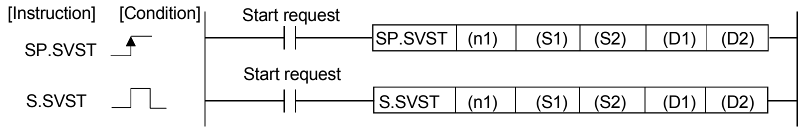


3.3 Servo Program Start Request from The PLC CPU to The Motion CPU: S(P).SVST (PLC instruction: S(P).SVST)

Setting data (Note)	Usable devices										
	Internal devices (System, User)		File register	Bit digit specified	Indirectly specified device	MELSECNET/10 direct J□\□		Special function module U□\G□	Index register Z□	Constant K, H	Other
	Bit	Word				Bit	Word				
(n1)		○	○	○	○					○	
(S1)		○	○		○						○
(S2)		○	○	○	○					○	
(D1)	○	○	○								
(D2)		○	○		○						

○ : Usable △ : Usable partly

(Note) : Setting data except (S1) : Index qualification possible



[Setting data]

Setting data	Description	Data type
(n1)	(First I/O No. of the target CPU)/16 Value to specify actually is the following. (Note-1) CPU No.2 : 3E1H, CPU No.3 : 3E2H, CPU No.4 : 3E3H	16-bit binary
(S1)	Axis No.("Jn") (Note-2) to start. Q173HCPU : J1 to J32/Q172HCPU : J1 to J8	Character sequence
(S2)	Servo program No. to start.	16-bit binary
(D1)	Complete devices (D1+0) : Device which make turn on for one scan at accept completion of instruction. (D1+1) : Device which make turn on for one scan at accept abnormal completion of instruction. ("D1+0" also turns on at the abnormal completion.)	Bit
(D2)	Device to store the complete status.	16-bit binary

(Note-1) : Motion CPU cannot used CPU No.1 in the Multiple CPU configuration.

(Note-2) : "n" shows the numerical value correspond to axis No..
Q173HCPU : Axis No.1 to No.32 (n=1 to 32) / Q172HCPU : Axis No.1 to No.8 (n=1 to 8)

[Controls]

(1) This instruction is dedicated instruction toward the Motion CPU in the Multiple CPU system. Errors occurs when it was executed toward the CPU except the Motion CPU.

(2) Request to start the servo program specified with (S2).

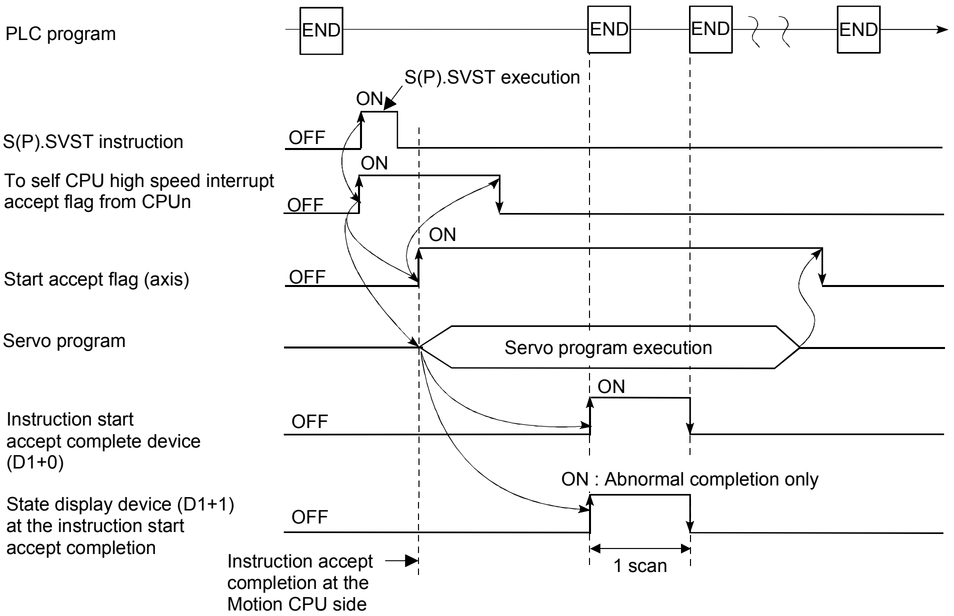
(3) This instruction is always effective regardless of the state of real mode/virtual mode/mode switching when the operating system software of Motion CPU is SV22.

(4) S(P).SFCS/S(P).SVST/S(P).CHGA/S(P).CHGV/S(P).CHGT/S(P).DDRD/ S(P).DDWR cannot be executed simultaneously toward the CPU executing S(P).SFCS instruction.
When the Motion dedicated PLC instruction is started continuously, It is necessary to take an interlock by the to self CPU high speed interrupt accept flag from CPUn.

(5) When the servo program is executed also at the motion control step (Kn) in the Motion CPU, it is necessary to take an interlock by user program, because there is no flag which can distinguish the axis starting in the PLC CPU.
Start accept flag (M2001 to M2032) of the motion devices or positioning start completion flag (M2400+20n) is used as the interlock condition.

(6) It is necessary to take an inter-lock by the start accept flag of the shared CPU memory so that multiple instructions may not be executed toward the same axis of the same Motion CPU No..

[Operation]



(1) The start accept status of each axis can be confirmed with the start accept flag in the shared CPU memory of target CPU.

(2) S(P).SVST instruction accepting and normal/abnormal completion can be confirmed with the complete device (D1) or status display device (D2) at the completion.

(a) Complete device
It is turned on by the END processing of scan which the instruction completed, and turned off by the next END processing.

(b) Status display device at the completion
It is turned on/off according to the status of the instruction completion.

- Normal completion : OFF
- Abnormal completion : It is turned on by the END processing of scan which the instruction completed, and turned off by the next END processing.

[Setting range]

(1) Setting of the starting axis
The starting axis set as (S1) sets J + Axis No. in a character sequence " ".

	(S1) usable range
Q173HCPU	1 to 32
Q172HCPU	1 to 8

Up to 8 axes can be set. If multiple axes are set, it sets without dividing in a space etc,.
The axis No. set in the system setting is used as the axis No. to start.
Refer to the "Q173HCPU/Q172HCPU Motion controller Programming Manual (COMMON)" for system settings.
And, the axis No. to start does not need to be a order.

Example) When multiple axes (Axis1, Axis2, Axis10, Axis11) are set.
"J1J2J10J11"

(2) Setting of the servo program No.

(S2) usable range
0 to 4095

[Start accept flag (System area)]

The complete status of the start accept flag is stored in the address of the start accept flag in the shared CPU memory.

Shared CPU memory address ( ) is decimal address	Description
204H(516) 205H(517)	The start accept flag is stored by the 1 to 32 axis, each bit. (As for a bit's actually being set Q173HCPU : J1 to J32/ Q172HCPU : J1 to J8.) OFF : Start accept flag usable ON : Start accept flag disable b15 ・・・ b1 b0 204H(516) address: J16 ・・・・・・・・・ J2 J1 205H(517) address: J32 ・・・・・・・・・ J17

[Errors]

The abnormal completion in the case shown below, and the error code is stored in the device specified with the complete status storing device (D2).

Complete status (Note) (Error code)(H)	Error factor	Corrective action
4C00	The specified device cannot be used in the Motion CPU. Or, it is outside the device range.	Confirm a program, and correct it to a correct PLC program.
4C01	The instruction for the Multiple CPU system which did not be correspond with operating system software of the Motion CPU was executed.	
4C03	The servo program No. to execute is outside the range 0 to 4095.	
4C04	Axis No. set by SVST instruction is injustice.	
4C08	There are 33 or more instruction requests to the Motion CPU from the PLC CPU in S(P).SFCS/ S(P).SVST/S(P).CHGA/S(P).GINT sum table simultaneously, and the Motion CPU cannot process them.	
4C09	CPU No. of the instruction cause is injustice.	

(Note) : 0000H (Normal)

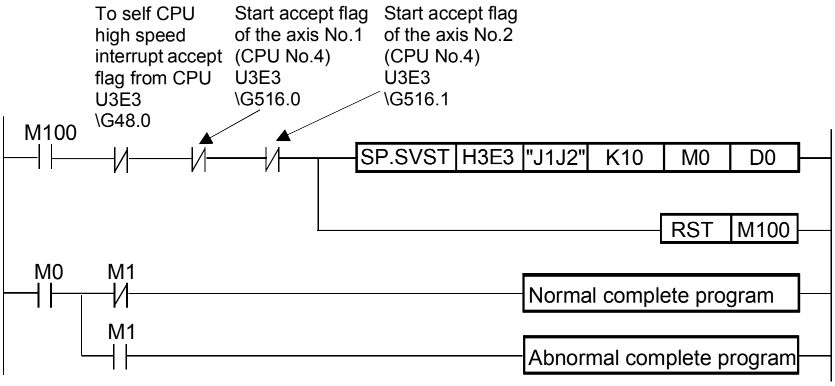
The error flag (SM0) is turned on an operation error in the case shown below, and an error code is stored in SD0.

Error code (Note)	Error factor	Corrective action
2110	The CPU No. to be set by "(First I/O No. of the target CPU)/16" is specified.	Confirm a program, and correct it to a correct PLC program.
2114	The self CPU is by "(First I/O No. of the target CPU)/16" is specified.	
2117	The CPU except the Motion CPU by "(First I/O No. of the target CPU)/16" is specified.	
4004	The instruction be composed of devices except usable devices.	
4100	Since 0 to 3DFH, 3E4H is specified by "(First I/O No. of the target CPU)/16" is specified.	

(Note) : 0000H (Normal)

[Program example]

Program which requests to start the servo program No.10 toward axis No.1 and No.2 of the Motion CPU No.4. from the PLC CPU No.1.

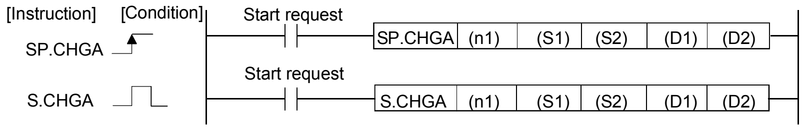


3.4 Current Value Change Instruction from The PLC CPU to The Motion CPU: S(P).CHGA (PLC instruction: S(P).CHGA)

Setting data (Note)	Usable devices										
	Internal devices (System, User)		File register	Bit digit specified	Indirectly specified device	MELSECNET/10 direct J□\□		Special function module U□\G□	Index register Z□	Constant K, H	Other
	Bit	Word				Bit	Word				
(n1)		○	○	○	○					○	
(S1)		○	○		○						○
(S2)		○	○	○	○					○	
(D1)	○	○	○								
(D2)		○	○		○						

○ : Usable △ : Usable partly

(Note) : Setting data except (S1) : Index qualification possible



[Setting data]

Setting data	Description	Data type
(n1)	(First I/O No. of the target CPU)/16 Value to specify actually is the following. (Note-1) CPU No.2 : 3E1H, CPU No.3 : 3E2H, CPU No.4 : 3E3H	16-bit binary
(S1)	Axis No. ("Jn") (Note-2) to execute the current value change. Q173HCPU : J1 to J32/Q172HCPU : J1 to J8	Character sequence
	Synchronous encoder axis No. ("En") to execute the current value change. Q173HCPU : E1 to E12/Q172HCPU : E1 to E8	
	Cam axis No. ("Cn") to execute the within-one-revolution current value change. Q173HCPU : C1 to C32/Q172HCPU : C1 to C8	
(S2)	Setting of the current value to change.	32-bit binary
(D1)	Complete devices (D1+0) : Device which make turn on for one scan at accept completion of instruction. (D1+1) : Device which make turn on for one scan at accept abnormal completion of instruction. ("D1+0" also turns on at the abnormal completion.)	Bit
(D2)	Device to store the complete status.	16-bit binary

(Note-1) : Motion CPU cannot used CPU No.1 in the Multiple CPU configuration.

(Note-2) : "n" shows the numerical value which correspond to axis No..
Q173HCPU : Axis No.1 to No.32 (n=1 to 32) / Q172HCPU : Axis No.1 to No.8 (n=1 to 8)

● When an axis No."Jn" was specified with (S1)

[Controls]

(1) This instruction is dedicated instruction toward the Motion CPU in the Multiple CPU system. Errors occurs when it was executed toward the CPU except the Motion CPU.

(2) The current value change of axis (stopped axis) No. specified with (S1) is changed into the current value specified (S2).

(3) This instruction is always effective regardless of the state of real mode/virtual mode/mode switching when the operating system software of Motion CPU is SV22.

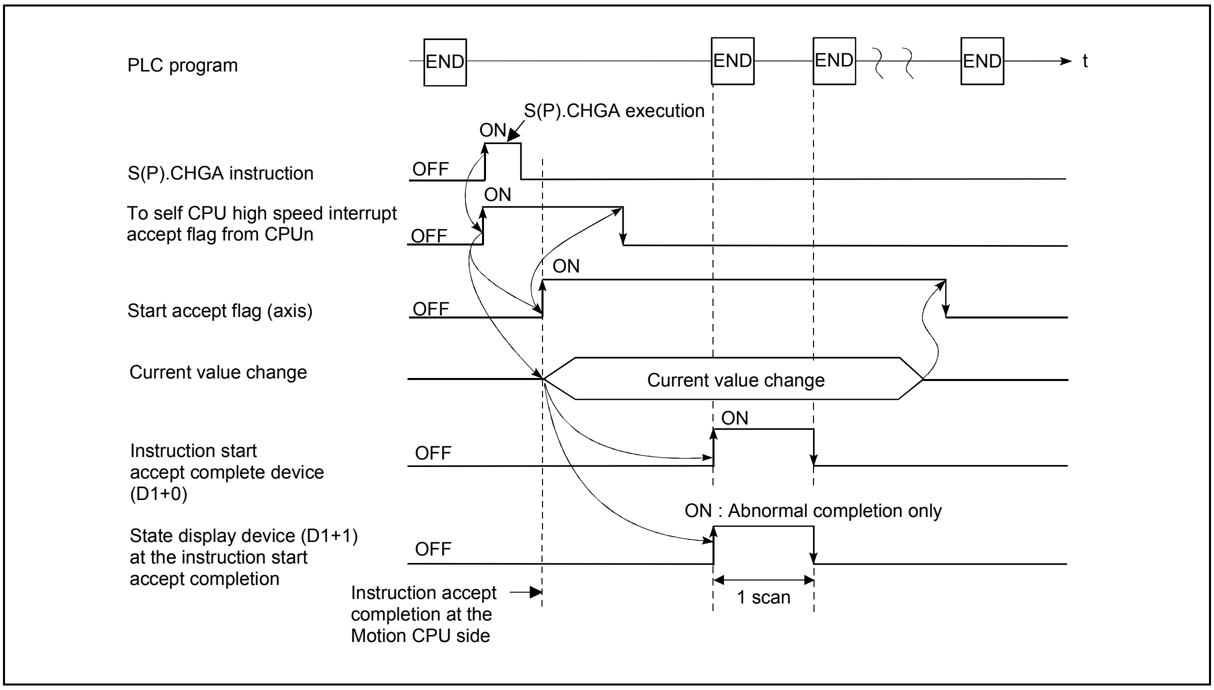
(4) S(P).SFCS/S(P).SVST/S(P).CHGA/S(P).CHGV/S(P).CHGT/S(P).DDRD/ S(P).DDWR cannot be executed simultaneously toward the CPU executing S(P).CHGA instruction.
When the Motion dedicated PLC instruction is started continuously, It is necessary to take an interlock by the to self CPU high speed interrupt accept flag from CPUn.

(5) When the servo program is executed also at the motion control step (Kn) in the Motion CPU, it is necessary to take an interlock by user program, because there is no flag which can distinguish the axis starting in the PLC CPU.
Start accept flag (M2001 to M2032) of the motion devices is used as the interlock condition in the Motion CPU.

(6) It is necessary to take an interlock by the start accept flag of the shared CPU memory so that multiple instructions may not be executed toward the same axis of the same Motion CPU No..

(7) The current change value is also possible when the servo program which execute the CHGA instruction toward an axis is executed in the S(P).SVST instruction.

[Operation]



(1) The start accept status of each axis can be confirmed with the start accept flag in the shared CPU memory of target CPU.

(2) S(P).CHGA instruction accepting and normal/abnormal completion can be confirmed with the complete device (D1) or status display device (D2) at the completion.

(a) Complete device
It is turned on by the END processing of scan which the instruction completed, and turned off by the next END processing.

(b) Status display device at the completion
It is turned on/off according to the status of the instruction completion.

- Normal completion : OFF
- Abnormal completion : It is turned on by the END processing of scan which the instruction completed, and turned off by the next END processing.

[Setting range]

(1) Setting of axis to execute the current value change.
The starting axis set as (S1) sets J + Axis No. in a character sequence " ".

	(S1) usable range
Q173HCPU	1 to 32
Q172HCPU	1 to 8

The number of axes which can set are only 1 axis.
The axis No. set in the system setting is used as the axis No. to start.
Refer to the "Q173HCPU/Q172HCPU Motion controller Programming Manual (COMMON)" for system settings.

(2) Setting of the current value to change.

(S2) usable range
-2147483648 to 2147483647

[Start accept flag (System area)]

The complete status of the start accept flag is stored in the address of the start accept flag in the shared CPU memory.

<table>
<tr><th>Shared CPU memory address
( ) is decimal address</th><th>Description</th></tr>
<tr><td>204H(516)
205H(517)</td><td>The start accept flag is stored by the 1 to 32 axis, each bit.
(As for a bit's actually being set Q173HCPU : J1 to J32/ Q172HCPU : J1 to J8.)
OFF : Start accept flag usable
ON : Start accept flag disable
<table>
<tr><th></th><th>b15</th><th></th><th>b1</th><th>b0</th></tr>
<tr><td>204H(516) address</td><td>J16</td><td>••••••••</td><td>J2</td><td>J1</td></tr>
<tr><td>205H(517) address</td><td>J32</td><td>••••••••</td><td></td><td>J17</td></tr>
</table>
</td></tr>
</table>

[Errors]

The abnormal completion in the case shown below, and the error code is stored in the device specified with the complete status storing device (D2).

Complete status (Note) (Error code)(H)	Error factor	Corrective action
4C00	The specified device cannot be used in the Motion CPU. Or, it is outside the device range.	Confirm a program, and correct it to a correct PLC program.
4C01	The instruction for the Multiple CPU system which did not be correspond with operating system software of the Motion CPU was executed.	
4C05	Axis No. set by CHGA instruction is injustice.	
4C08	There are 33 or more instruction requests to the Motion CPU from the PLC CPU in S(P).SFCS/ S(P).SVST/S(P).CHGA/S(P).GINT sum table simultaneously, and the Motion CPU cannot process them.	
4C09	CPU No. of the instruction cause is injustice.	

(Note) : 0000H (Normal)

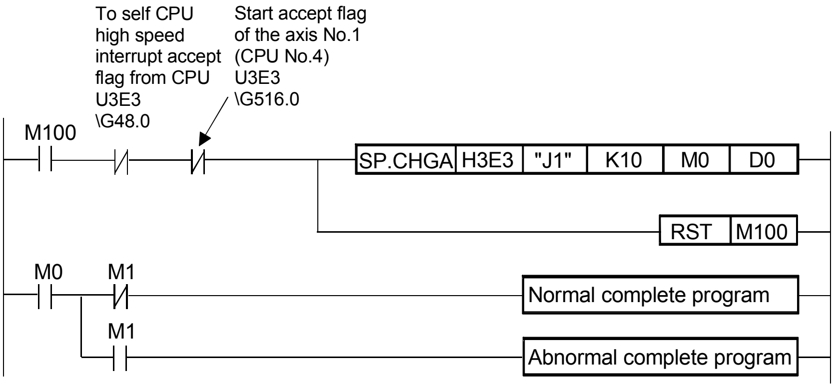
The error flag (SM0) is turned on an operation error in the case shown below, and an error code is stored in SD0.

Error code (Note)	Error factor	Corrective action
2110	The CPU No. to be set by "(First I/O No. of the target CPU)/16" is specified.	Confirm a program, and correct it to a correct PLC program.
2114	The self CPU by "(First I/O No. of the target CPU)/16" is specified.	
2117	The CPU except the Motion CPU by "(First I/O No. of the target CPU)/16" is specified.	
4004	The instruction is composed of devices except usable devices.	
4100	Since 0 to 3DFH, 3E4H by "(First I/O No. of the target CPU)/16" is specified.	

(Note) : 0000H (Normal)

[Program example]

Program which changes the current value of the axis No.1 of the Motion CPU (CPU No.4) from PLC CPU (CPU No.1) to 10.



● When an axis No."En" was specified with (S1)

[Controls]

(1) This instruction is dedicated instruction toward the Motion CPU at the Multiple CPU system. Errors occurs when it was executed toward the CPU except the Motion CPU.

(2) The synchronous encoder axis current value specified with (S1) is changed into the current value specified (S2) at the virtual mode.

(3) This instruction is always effective regardless of the state of real mode/virtual mode/mode switching when the operating system software of Motion CPU is SV22.

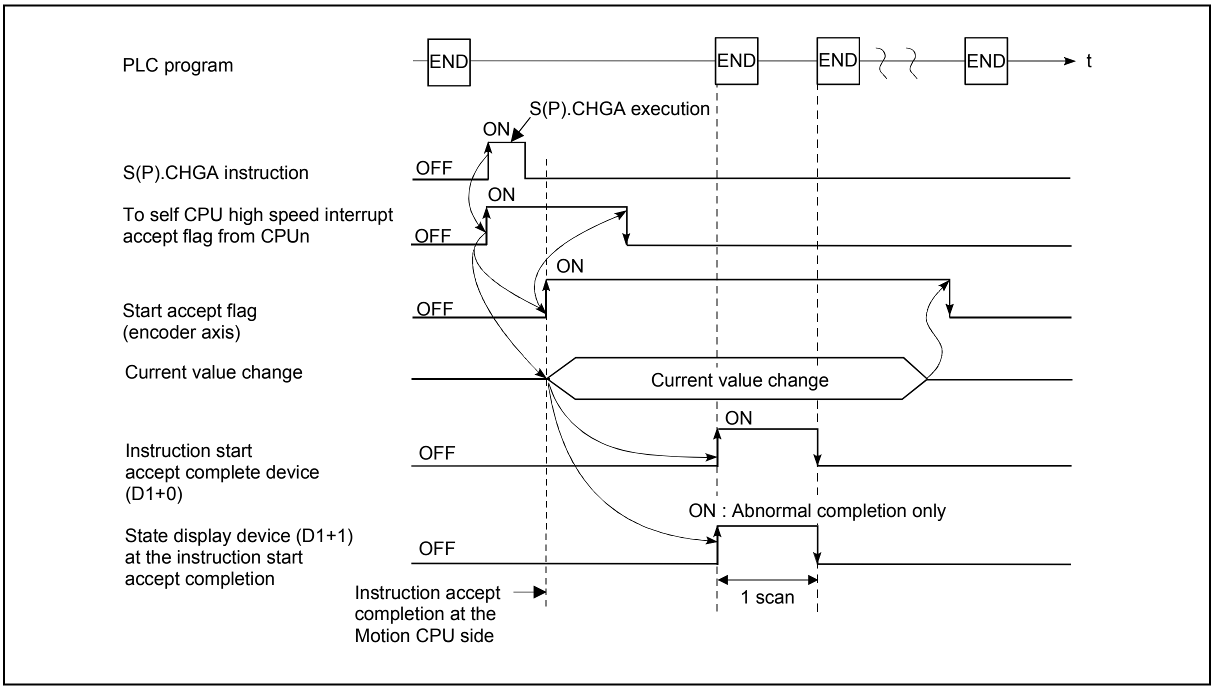
(4) S(P).SFCS/S(P).SVST/S(P).CHGA/S(P).CHGV/S(P).CHGT/S(P).DDRD/ S(P).DDWR cannot be executed simultaneously toward the CPU executing S(P).CHGA instruction.
When the Motion dedicated PLC instruction is started continuously, It is necessary to take an interlock by the to self CPU high speed interrupt accept flag from CPUn.

(5) When the servo program is executed also at the motion control step (Kn) in the Motion CPU, it is necessary to take an interlock by user program, because there is no flag which can distinguish the axis starting in the PLC CPU.
Synchronous encoder current value changing flag (M2101 to M2112) of the motion devices is used as the interlock condition in the Motion CPU.

(6) It is necessary to take an interlock by the current value changing flag of the shared CPU memory so that multiple instructions may not be executed toward the same synchronous encoder axis of the same Motion CPU No..

(7) The current change value is also possible when the servo program which execute the CHGA instruction toward the synchronous encoder axis is executed in the S(P).SVST instruction.

[Operation]



(1) The current value status of the synchronous encoder axis can be confirmed with the current value changing in the shared CPU memory of target CPU.

(2) S(P).CHGA instruction accepting and normal/abnormal completion can be confirmed with the complete device (D1) or status display device (D2) at the completion.

(a) Complete device

It is turned on by the END processing of scan which the instruction completed, and turned off by the next END processing.

(b) Status display device at the completion

It is turned on/off according to the status of the instruction completion.

- Normal completion : OFF
- Abnormal completion : It is turned on by the END processing of scan which the instruction completed, and turned off by the next END processing.

[Setting range]

(1) Setting of the synchronous encoder axis to execute the current value change.
The synchronous encoder axis to execute the current value change set as (S1) sets E + synchronous encoder axis No. in a character sequence " ".

	(S1) usable range
Q173HCPU	1 to 12
Q172HCPU	1 to 8

The number of axes which can set are only 1 axis.
The axis No. set in the system setting is used as the axis No. to start.
Refer to the "Q173HCPU/Q172HCPU Motion controller Programming Manual (COMMON)" for system settings.

(2) Setting of the current value to change.

(S2) usable range
-2147483648 to 2147483647

[Synchronous encoder current value changing flag (System area)]

The complete status of the synchronous encoder current value changing flag is stored in the address of the synchronous encoder current value changing flag in the shared CPU memory.

Shared CPU memory address ( ) is decimal address	Description
208H(520)	The synchronous encoder current value changing flag is stored by the 1 to 16 axis, each bit. (As for a bit's actually being set Q173HCPU : E1 to E12/ Q172HCPU : E1 to E8.) OFF : Start accept usable ON : Start accept disable b15 b1 b0 208H(520) address E16 ••••••••• E2 E1

[Errors]

The abnormal completion in the case shown below, and the error code is stored in the device specified with the complete status storing device (D2).

Complete status (Note) (Error code)(H)	Error factor	Corrective action
4C00	The specified device cannot be used in the Motion CPU. Or, it is outside the device range.	Confirm a program, and correct it to a correct PLC program.
4C01	The instruction for the Multiple CPU system which did not be correspond with operating system software of the Motion CPU was executed.	
4C05	Axis No. set by CHGA instruction is injustice.	
4C08	There are 33 or more instruction requests to the Motion CPU from the PLC CPU in S(P).SFCS/ S(P).SVST/S(P).CHGA/S(P).GINT sum table simultaneously, and the Motion CPU cannot process them.	
4C09	CPU No. of the instruction cause is injustice.	

(Note) : 0000H (Normal)

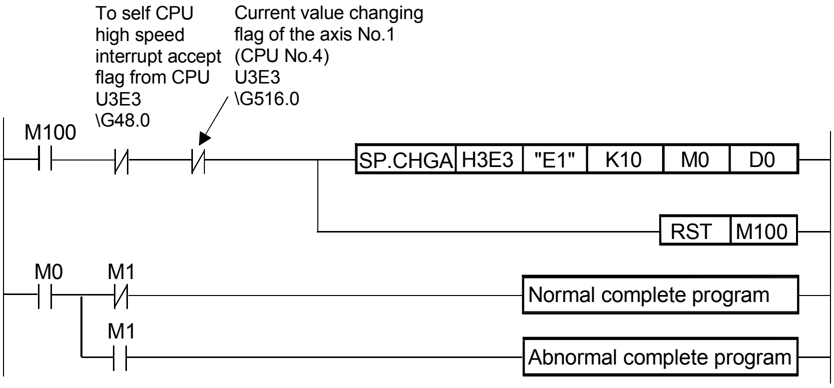
The error flag (SM0) is turned on an operation error in the case shown below, and an error code is stored in SD0.

Error code (Note)	Error factor	Corrective action
2110	The CPU No. to be set by "(First I/O No. of the target CPU)/16" is specified.	Confirm a program, and correct it to a correct PLC program.
2114	The self CPU by "(First I/O No. of the target CPU)/16" is specified.	
2117	The CPU except the Motion CPU by "(First I/O No. of the target CPU)/16" is specified.	
4004	The instruction is composed of devices except usable devices.	
4100	Since 0 to 3DFH, 3E4H by "(First I/O No. of the target CPU)/16" is specified.	

(Note) : 0000H (Normal)

[Program example]

Program which changes the current value of the axis No.1 of the Motion CPU (CPU No.4) from PLC CPU (CPU No.1) to 10.



● When an axis No."Cn" was specified with (S1)

[Controls]

(1) This instruction is dedicated instruction toward the Motion CPU in the Multiple CPU system. Errors occurs when it was executed toward the CPU except the Motion CPU.

(2) The cam axis within-one-revolution current value specified with (S1) is changed into the current value specified (S2) at the virtual mode.

(3) This instruction is always effective regardless of the state of real mode/virtual mode/mode switching when the operating system software of Motion CPU is SV22.

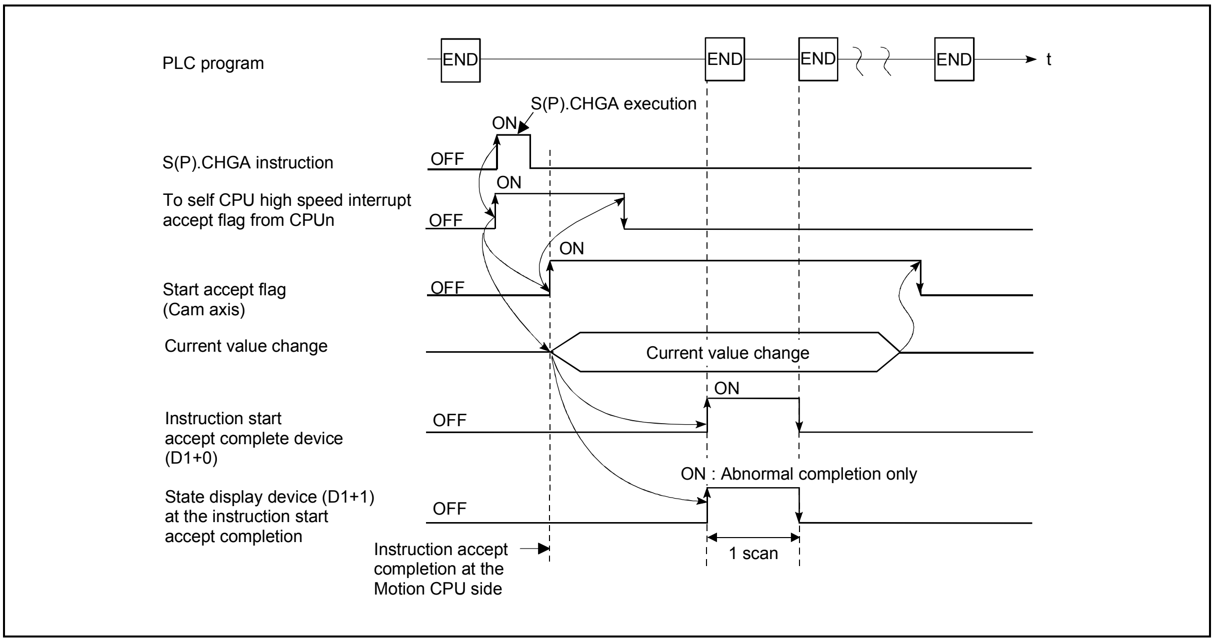
(4) S(P).SFCS/S(P).SVST/S(P).CHGA/S(P).CHGV/S(P).CHGT/S(P).DDRD/ S(P).DDWR cannot be executed simultaneously toward the CPU executing S(P).CHGA instruction.
When the Motion dedicated PLC instruction is started continuously, It is necessary to take an interlock by the to self CPU high speed interrupt accept flag from CPUn.

(5) When the servo program is executed also at the motion control step (Kn) in the Motion CPU, it is necessary to take an interlock by user program, because there is no flag which can distinguish the axis starting in the PLC CPU.

(6) It is necessary to take an interlock by the cam axis within-one-revolution current value changing flag of the shared CPU memory so that multiple instructions may not be executed toward the same cam axis of the same Motion CPU No..

(7) The current change value is also possible when the servo program which execute the CHGA instruction toward the cam axis is executed in the S(P).SVST instruction.

[Operation]



(1) The current value status of the cam axis within-one-revolution current value change can be confirmed with the cam axis within-one-revolution current value changing flag in the shared CPU memory of target CPU.

(2) S(P).CHGA instruction accepting and normal/abnormal completion can be confirmed with the complete device (D1) or status display device (D2) at the completion.

(a) Complete device

It is turned on by the END processing of scan which the instruction completed, and turned off by the next END processing.

(b) Status display device at the completion

It is turned on/off according to the status of the instruction completion.

- Normal completion : OFF
- Abnormal completion : It is turned on by the END processing of scan which the instruction completed, and turned off by the next END processing.

[Setting range]

(1) Setting the cam axis which execute the within-one-revolution current value change.
The cam axis to execute the within-one-revolution current value change set as (S1) sets C + cam axis No. in a character sequence " ".

	(S1) usable range
Q173HCPU	1 to 32
Q172HCPU	1 to 8

The number of axes which can set are only 1 axis.
The axis No. set in the system setting is used as the axis No. to start.
Refer to the "Q173HCPU/Q172HCPU Motion controller Programming Manual (COMMON)" for system settings.

(2) Setting of the current value to change.

(S2) usable range
-2147483648 to 2147483647

[Cam axis within-one-revolution current value changing flag (System area)]

The complete status of the cam axis within-one-revolution current value changing flag is stored in the address of the cam axis within-one-revolution current value changing flag in the shared CPU memory.

Shared CPU memory address ( ) is decimal address	Description
20CH(524) 20DH(525)	The cam axis within-one-revolution current value changing flag is stored by the 1 to 32 axis, each bit. (As for a bit's actually being set Q173HCPU : C1 to C32/ Q172HCPU : C1 to C8.) OFF : Start accept usable ON : Start accept disable b15 ········ b1 b0 20CH(524) address: C16 ········ C2 C1 20DH(525) address: C32 ········ C17

[Errors]

The abnormal completion in the case shown below, and the error code is stored in the device specified with the complete status storing device (D2).

Complete status (Note) (Error code)(H)	Error factor	Corrective action
4C00	The specified device cannot be used in the Motion CPU. Or, it is outside the device range.	Confirm a program, and correct it to a correct PLC program.
4C01	The instruction for the Multiple CPU system which did not be correspond with operating system software of the Motion CPU was executed.	
4C05	Axis No. set by CHGA instruction is injustice.	
4C08	There are 33 or more instruction requests to the Motion CPU from the PLC CPU in S(P).SFCS/ S(P).SVST/S(P).CHGA/S(P).GINT sum table simultaneously, and the Motion CPU cannot process them.	
4C09	CPU No. of the instruction cause is injustice.	

(Note) : 0000H (Normal)

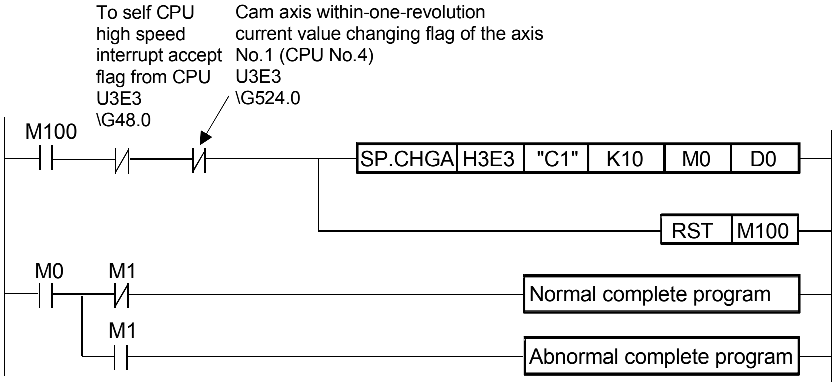
The error flag (SM0) is turned on an operation error in the case shown below, and an error code is stored in SD0.

Error code (Note)	Error factor	Corrective action
2110	The CPU No. to be set by "(First I/O No. of the target CPU)/16" is specified.	Confirm a program, and correct it to a correct PLC program.
2114	The self CPU by "(First I/O No. of the target CPU)/16" is specified.	
2117	The CPU except the Motion CPU by "(First I/O No. of the target CPU)/16" is specified.	
4004	The instruction is composed of devices except usable devices.	
4100	Since 0 to 3DFH, 3E4H by "(First I/O No. of the target CPU)/16" is specified.	

(Note) : 0000H (Normal)

[Program example]

Program which changes the current value of the axis No.1 of the Motion CPU (CPU No.4) from PLC CPU (CPU No.1) to 10.

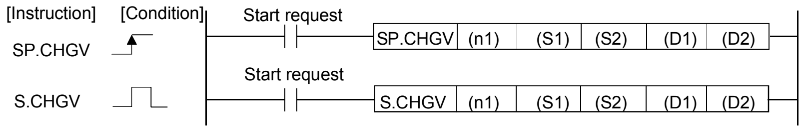


3.5 Speed Change Instruction from The PLC CPU to The Motion CPU: S(P).CHGV (PLC instruction: S(P).CHGV)

Setting data (Note)	Usable devices										
	Internal devices (System, User)		File register	Bit digit specified	Indirectly specified device	MELSECNET/10 direct J□\□		Special function module U□\G□	Index register Z□	Constant K, H	Other
	Bit	Word				Bit	Word				
(n1)		○	○	○	○					○	
(S1)		○	○		○						○
(S2)		○	○	○	○					○	
(D1)	○	○	○								
(D2)		○	○		○						

○ : Usable △ : Usable partly

(Note) :Setting data except (S1) : Index qualification possible



[Setting data]

Setting data	Description	Data type
(n1)	(First I/O No. of the target CPU)/16 Value to specify actually is the following. (Note-1) CPU No.2 : 3E1H, CPU No.3 : 3E2H, CPU No.4 : 3E3H	16-bit binary
(S1)	Axis No. ("Jn") (Note-2) to execute the speed change. Q173HCPU : J1 to J32/Q172HCPU : J1 to J8	Character sequence
(S2)	Setting of the current value to change.	16-bit binary
(D1)	Complete devices (D1+0) : Device which make turn on for one scan at accept completion of instruction. (D1+1) : Device which make turn on for one scan at accept abnormal completion of instruction. ("D1+0" also turns on at the abnormal completion.)	Bit
(D2)	Device to store the complete status.	16-bit binary

(Note-1) : Motion CPU cannot used CPU No.1 in the Multiple CPU configuration.

(Note-2) : "n" shows the numerical value which correspond to axis No..
Q173HCPU : Axis No.1 to No.32 (n=1 to 32) / Q172HCPU : Axis No.1 to No.8 (n=1 to 8)

[Controls]

(1) This instruction is dedicated instruction toward the Motion CPU in the Multiple CPU system. Errors occurs when it was executed toward the CPU except the Motion CPU.

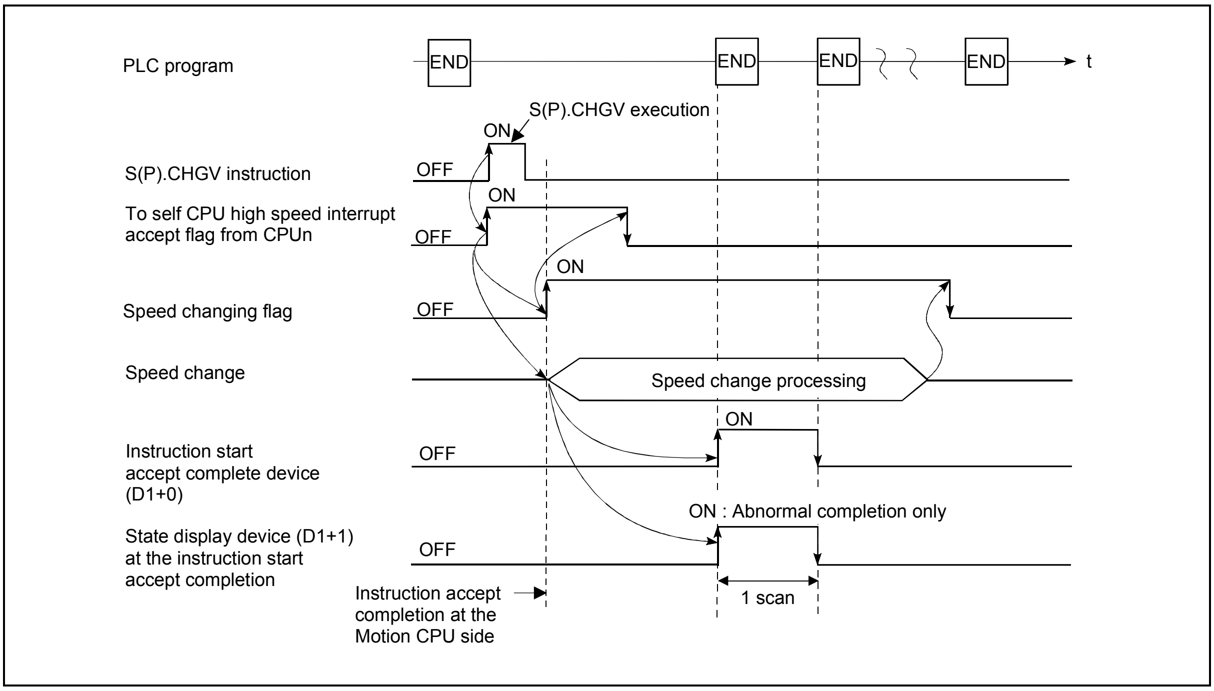
(2) The speed change is executed of the axis specified with (S1) during positioning or JOG operating.

(3) S(P).SFCS/S(P).SVST/S(P).CHGA/S(P).CHGV/S(P).CHGT/S(P).DDRD/ S(P).DDWR cannot be executed simultaneously toward the CPU executing S(P).CHGV instruction.
When the Motion dedicated PLC instruction is started continuously, It is necessary to take an interlock by the to self CPU high speed interrupt accept flag from CPUn.

(4) When the speed change is executed also at the operation control step (Fn/FSn) in the Motion CPU, it is necessary to take an interlock by user program, because there is no flag which can distinguish the speed changing in the PLC CPU.
Speed changing flag (M2061 to M2092) of the motion devices is used as the interlock condition in the Motion CPU.

(5) It is necessary to take an interlock by the speed changing flag of the shared CPU memory so that multiple instructions may not be executed toward the same axis of the same Motion CPU No..

[Operation]



[Setting range]

(1) Setting of axis to execute the speed change.

The axis to execute the speed change set as (S1) sets J + axis No. in a character sequence " ".

	(S1) usable range
Q173HCPU	1 to 32
Q172HCPU	1 to 8

The number of axes which can set are only 1 axis.
The axis No. set in the system setting is used as the axis No. to start.
Refer to the "Q173HCPU/Q172HCPU Motion controller Programming Manual (COMMON)" for system settings.

(2) Setting of the speed to change.

(S2) usable range
-2147483648 to 2147483647

[Speed changing flag (System area)]

The complete status of the start accept flag is stored in the address of the start accept flag in the shared CPU memory.

Shared CPU memory address ( ) is decimal address	Description
206H(518) 207H(519)	The start accept flag is stored by the 1 to 32 axis, each bit. (As for a bit's actually being set Q173HCPU : J1 to J32/ Q172HCPU : J1 to J8.) OFF : Start accept usable ON : Start accept disable b15 b1 b0 206H(518) address J16 • • • • • • • • • J2 J1 207H(519) address J32 • • • • • • • • • J17

[Errors]

The abnormal completion in the case shown below, and the error code is stored in the device specified with the complete status storing device (D2).

Complete status (Note) (Error code)(H)	Error factor	Corrective action
4C00	The specified device cannot be used in the Motion CPU. Or, it is outside the device range.	Confirm a program, and correct it to a correct PLC program.
4C01	The instruction for the Multiple CPU system which did not be correspond with operating system software of the Motion CPU was executed.	
4C06	Axis No. set by CHGV instruction is injustice.	
4C09	CPU No. of the instruction cause is injustice.	

(Note) : 0000H (Normal)

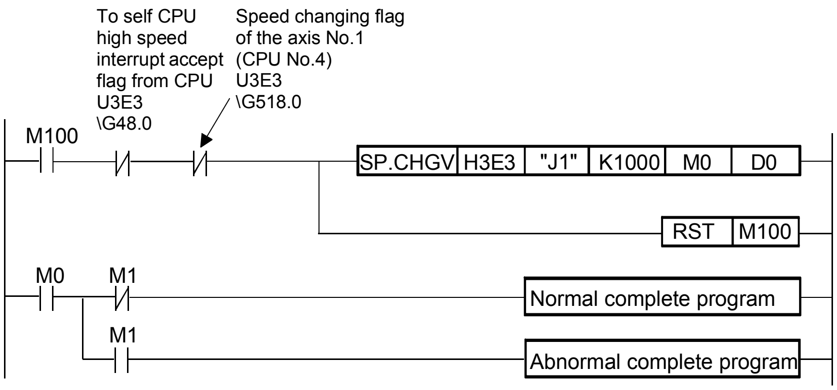
The error flag (SM0) is turned on an operation error in the case shown below, and an error code is stored in SD0.

Error code (Note)	Error factor	Corrective action
2110	The CPU No. to be set by "(First I/O No. of the target CPU)/16" is specified.	Confirm a program, and correct it to a correct PLC program.
2114	The self CPU by "(First I/O No. of the target CPU)/16" is specified.	
2117	The CPU except the Motion CPU by "(First I/O No. of the target CPU)/16" is specified.	
4004	The instruction is composed of devices except usable devices.	
4100	Since 0 to 3DFH, 3E4H by "(First I/O No. of the target CPU)/16" is specified.	

(Note) : 0000H (Normal)

[Program example]

Program which changes the positioning speed of the axis No.1 of the Motion CPU (CPU No.4) from PLC CPU (CPU No.1) to 1000.

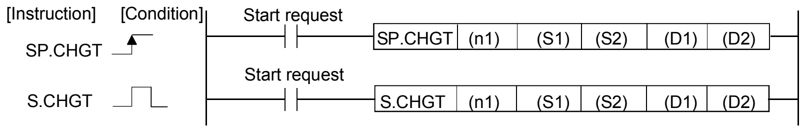


3.6 Torque Limit Value Change Request Instruction from The PLC CPU to The Motion CPU: S(P).CHGT (PLC instruction: S(P).CHGT)

Setting data (Note)	Usable devices										
	Internal devices (System, User)		File register	Bit digit specified	Indirectly specified device	MELSECNET/10 direct J□\□		Special function module U□\G□	Index register Z□	Constant K, H	Other
	Bit	Word				Bit	Word				
(n1)		○	○	○	○					○	
(S1)		○	○		○						○
(S2)		○	○	○	○					○	
(D1)	○	○	○								
(D2)		○	○		○						

○ : Usable △ : Usable partly

(Note) : Setting data except (S1) : Index qualification possible



[Setting data]

Setting data	Description	Data type
(n1)	(First I/O No. of the target CPU)/16 Value to specify actually is the following. (Note-1) CPU No.2 : 3E1H, CPU No.3 : 3E2H, CPU No.4 : 3E3H	16-bit binary
(S1)	Axis No. ("Jn") (Note-2) to execute the torque limit value change. Q173HCPU : J1 to J32/Q172HCPU : J1 to J8	Character sequence
(S2)	Setting of the torque limit value change to change.	16-bit binary
(D1)	Complete devices (D1+0) : Device which make turn on for one scan at accept completion of instruction. (D1+1) : Device which make turn on for one scan at accept abnormal completion of instruction. ("D1+0" also turns on at the abnormal completion.)	Bit
(D2)	Device to store the complete status.	16-bit binary

(Note-1) : Motion CPU cannot used CPU No.1 in the Multiple CPU configuration.

(Note-2) : "n" shows the numerical value which correspond to axis No..
Q173HCPU : Axis No.1 to No.32 (n=1 to 32) / Q172HCPU : Axis No.1 to No.8 (n=1 to 8)

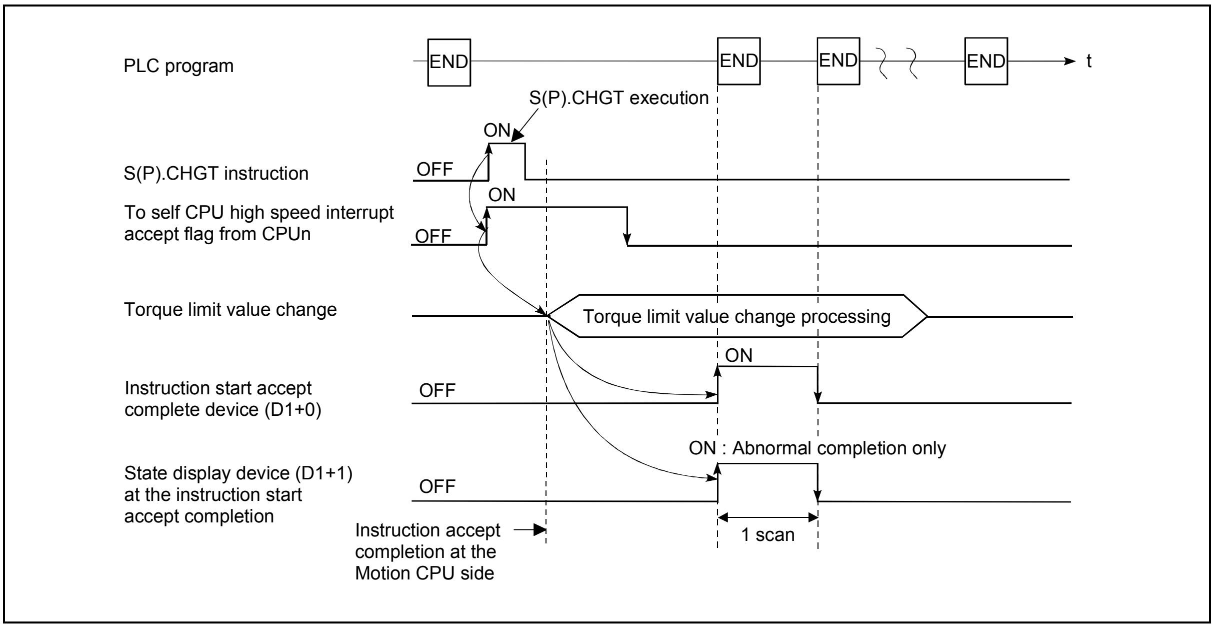
[Controls]

(1) This instruction is dedicated instruction toward the Motion CPU in the Multiple CPU system. Errors occurs when it was executed toward the CPU except the Motion CPU.

(2) The torque limit value of the axis specified with (S1) is changed to the value of (S2) regardless of the state of during operating or stopping at the real mode.

(3) S(P).SFCS/S(P).SVST/S(P).CHGA/S(P).CHGV/S(P).CHGT/S(P).DDRD/ S(P).DDWR cannot be executed simultaneously toward the CPU executing S(P).CHGT instruction.
When the Motion dedicated PLC instruction is started continuously, It is necessary to take an interlock by the to self CPU high speed interrupt accept flag from CPUn.

[Operation]



[Setting range]

(1) Setting of the axis to execute the torque limit value change.
The axis to execute the torque limit change set as (S1) sets J + axis No. in a character sequence " ".

	(S1) usable range
Q173HCPU	1 to 32
Q172HCPU	1 to 8

The number of axes which can set are only 1 axis.
The axis No. set in the system setting is used as the axis No. to start.
Refer to the "Q173HCPU/Q172HCPU Motion controller Programming Manual (COMMON)" for system settings.

(2) Setting of the torque limit value to change.

(S2) usable range
1 to 1000

[Errors]

The abnormal completion in the case shown below, and the error code is stored in the device specified with the complete status storing device (D2).

Complete status (Note) (Error code)(H)	Error factor	Corrective action
4C00	The specified device cannot be used in the Motion CPU. Or, it is outside the device range.	Confirm a program, and correct it to a correct PLC program.
4C01	The instruction for the Multiple CPU system which did not be correspond with operating system software of the Motion CPU was executed.	
4C07	Axis No. set by CHGT instruction is injustice.	
4C09	CPU No. of the instruction cause is injustice.	

(Note) : 0000H (Normal)

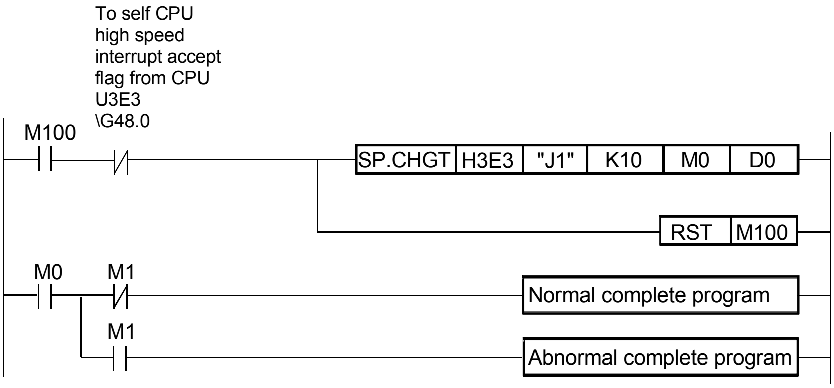
The error flag (SM0) is turned on an operation error in the case shown below, and an error code is stored in SD0.

Error code (Note)	Error factor	Corrective action
2110	The CPU No. to be set by "(First I/O No. of the target CPU)/16" is specified.	Confirm a program, and correct it to a correct PLC program.
2114	The self CPU by "(First I/O No. of the target CPU)/16" is specified.	
2117	The CPU except the Motion CPU by "(First I/O No. of the target CPU)/16" is specified.	
4004	The instruction is composed of devices except usable devices.	
4100	Since 0 to 3DFH, 3E4H by "(First I/O No. of the target CPU)/16" is specified.	

(Note) : 0000H (Normal)

[Program example]

Program which changes the torque limit value of the axis No.1 of the Motion CPU (CPU No.4) from PLC CPU (CPU No.1) to 10[%].

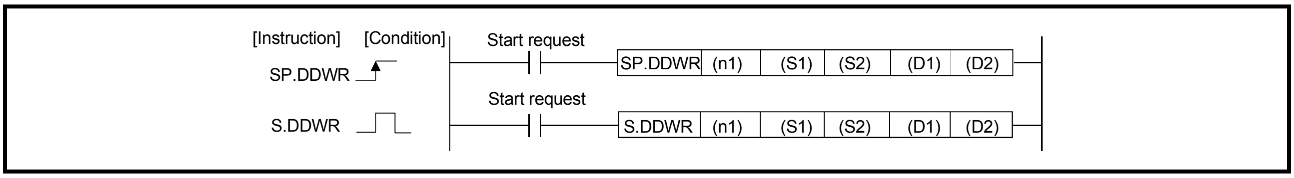


3.7 Write from The PLC CPU to The Motion CPU: S(P).DDWR (PLC instruction: S(P).DDWR)

Setting data (Note)	Usable devices										
	Internal devices (System, User)		File register	Bit digit specified	Indirectly specified device	MELSECNET/10 direct J□\□		Special function module U□\G□	Index register Z□	Constant K, H	Other
	Bit	Word				Bit	Word				
(n1)		○	○	○	○					○	
(S1)		○	○		○						
(S2)		○	○	△	○						
(D1)		○		△	○						
(D2)	○	○	○								

○ : Usable △ : Usable partly

(Note) : Setting data (n1) to (D2) : Index qualification possible



[Setting data]

Setting data	Description	Data type
(n1)	(First I/O No. of the target CPU)/16 Value to specify actually is the following. (Note-1) CPU No.1 : 3E0H, CPU No.2 : 3E1H, CPU No.3 : 3E2H, CPU No.4 : 3E3H	16-bit binary
(S1)	First device of the self CPU in which control data is stored.	
(S2)	First device of the self CPU in which writing data is stored.	
(D1)	First device of the target Motion CPU which stores the writing data.	
(D2)	Bit device which make turn on for one scan at completion of instruction.	Bit

(Note-1) : Motion CPU cannot used CPU No.1 in the Multiple CPU configuration.

[Control data]

Device	Item	Setting data	Setting range	Set by
S1+0	Complete status	The condition result at the completion of the instruction is stored. 0 : No error (Normal completion) Except 0 : Error code	—	System
S1+1	Number of writing data	Set the number of writing data	1 to 16	User

[Controls]

(1) This instruction is dedicated instruction toward the Motion CPU in the Multiple CPU system. Errors occurs when it was executed toward the CPU except the Motion CPU.
A part for the number of writing data of the control data specified with (S1) of data since the device specified with (S2) of the self CPU are stored to since the word device specified with (D1) of the target CPU (n1) in the Multiple CPU system.

(2) Figure specification of the bit device is possible for (S2) and (D1). However, figure specification is 4 figures and a start bit device number is only the multiple of 16. It becomes INSTRCT CODE ERROR [4004] when other values are specified.

(3) If the target CPU is not instruction acceptable condition, even if the S(P).DDWR instruction is executed, it may not be processed. In this case, it is necessary to execute the S(P).DDWR instruction again.
(S(P).SFCS/S(P).SVST/S(P).CHGA/S(P).CHGV/S(P).CHGT/S(P).DDRD/ S(P).DDWR cannot be executed simultaneously toward the CPU executing S(P).DDWR instruction.). It can be confirmed by data in the shared CPU memory of the target CPU (Motion CPU) whether the instruction is acceptable or not.
When the Motion dedicated PLC instruction is started continuously, it is must be design to execute next instruction after executing instruction complete device on.

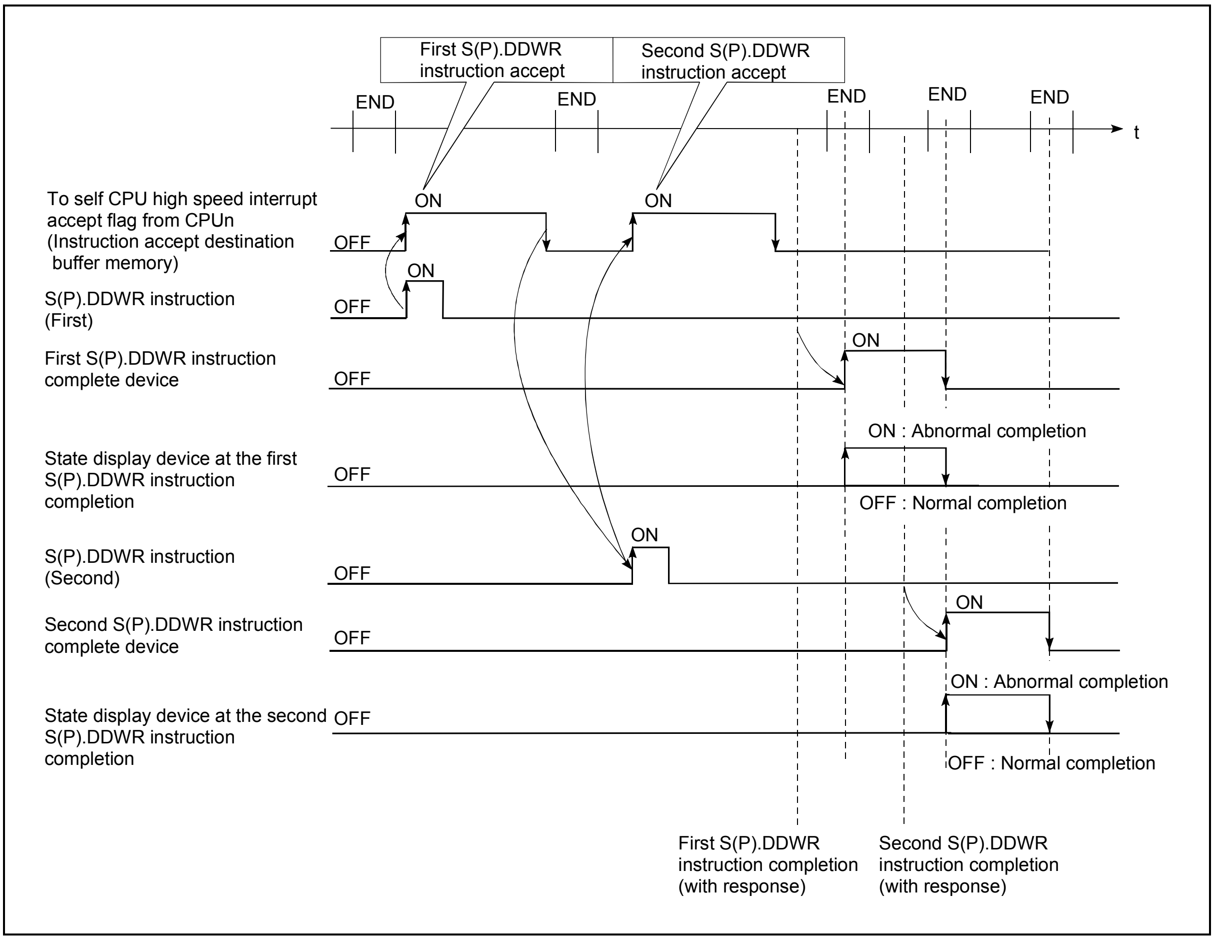
(4) The target CPU device range check is not executed with self CPU at the S(P).DDWR instruction execution, but it checks by the target CPU side, and it becomes abnormal completion at the device range over.

(5) S(P).DDWR instruction accepting and normal/abnormal completion can be confirmed with the complete device (D1) or status display device (D2) at the completion.
   (a) Complete device
   It is turned on by the END processing of scan which the instruction completed, and turned off by the next END processing.

   (b) Status display device at the completion
   It is turned on/off according to the status of the instruction completion.
   - Normal completion : OFF
   - Abnormal completion : It is turned on by the END processing of scan which the instruction completed, and turned off by the next END processing.

(6) SM390 turns on when the target CPU specified with (n1) complete to accept.
SM390 turns off when the target CPU specified with (n1) cannot be write correctly by the reset status or error factor (5000 to 5999).

[Operation]



[Errors]

The abnormal completion in the case shown below, and the error code is stored in the control data (S1+ 0 : Complete status).

Complete status (Note) (Error code)(H)	Error factor	Corrective action
4C00	The specified device cannot be used with the Motion CPU. Or, it is outside the device range.	Confirm a program, and correct it to a correct PLC program.
4C08	There are 33 or more instruction requests to the Motion CPU from the PLC CPU in S(P).DDRD/S(P).DDWR sum table simultaneously, and the Motion CPU cannot process them.	
4C09	CPU No. of the instruction cause is injustice.	

(Note) : 0000H (Normal)

The error flag (SM0) is turned on an operation error in the case shown below, and an error code is stored in SD0.

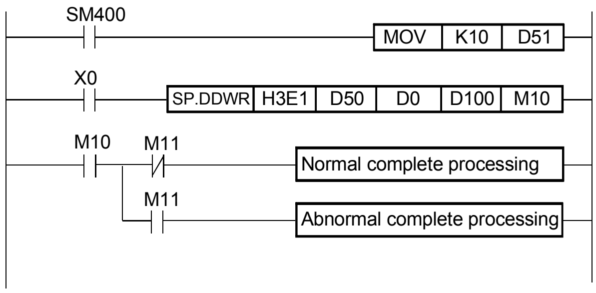
Error code (Note)	Error factor	Corrective action
2110	The CPU No. to be set by "(First I/O No. of the target CPU)/16" is specified.	Confirm a program, and correct it to a correct PLC program.
2114	The self CPU by "(First I/O No. of the target CPU)/16" is specified.	
2117	The CPU except the Motion CPU by "(First I/O No. of the target CPU)/16" is specified.	
4002	Specified instruction is wrong.	
4004	The instruction is composed of devices except usable devices.	
4100	Since 0 to 3DFH, 3E4H is specified by "(First I/O No. of the target CPU)/16" is specified.	
4101	Number of the writing data is except 1 to 16.	
	Number of writing data exceeds range of the storage device of the written data.	

(Note) : 0000H (Normal)

[Program example]

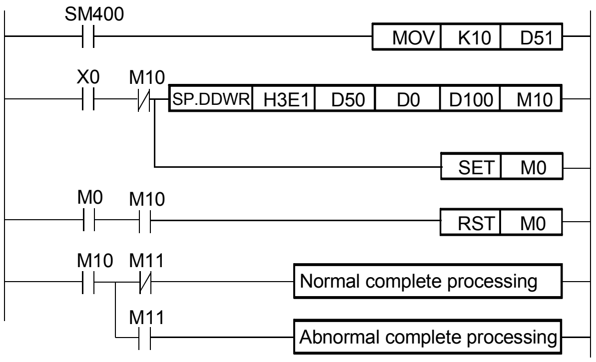
<Example 1>

Program which stores 10 points worth of the data from D0 of the self CPU (CPU No.1) since D100 of CPU No.2., when X0 is turned on.



<Example 2>

Program which stores 10 points worth of the data from D0 of the self CPU (CPU No.1) since D100 of CPU No.2. during turn on X0.

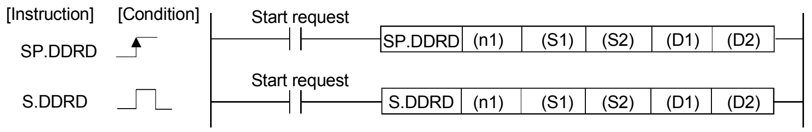


3.8 Read from The Devices of The Motion CPU: S(P).DDRD (PLC instruction: S(P).DDRD)

Setting data (Note)	Usable devices										
	Internal devices (System, User)		File register	Bit digit specified	Indirectly specified device	MELSECNET/10 direct J□\□		Special function module U□\G□	Index register Z□	Constant K, H	Other
	Bit	Word				Bit	Word				
(n1)		○	○	○	○					○	
(S1)		○	○		○						
(S2)		○		△	○						
(D1)		○	○	△	○						
(D2)	○	○	○								

○ : Usable　　△ : Usable partly

(Note) : Setting data (n1) to (D2) : Index qualification possible



[Setting data]

Setting data	Description	Data type
(n1)	(First I/O No. of the target CPU)/16 Value to specify actually is the following. (Note-1) CPU No.1 : 3E0H, CPU No.2 : 3E1H, CPU No.3 : 3E2H, CPU No.4 : 3E3H	16-bit binary
(S1)	First device of the self CPU in which control data is stored.	
(S2)	First device of the target CPU in which reading data is stored.	
(D1)	First device of the self CPU which stores the reading data.	
(D2)	Bit device which make turn on for one scan at completion of instruction.	Bit

(Note-1) : Motion CPU cannot used CPU No.1 in the Multiple CPU configuration.

[Control data]

Device	Item	Setting data	Setting range	Set by
S1+0	Complete status	The condition result at the completion of the instruction is stored. 0 : No error (Normal completion) Except 0 : Error code	—	System
S1+1	Number of reading data	Set the number of reading data.	1 to 16	User

[Controls]

(1) This instruction is dedicated instruction toward the Motion CPU in the Multiple CPU system. Errors occurs when it was executed toward the CPU except the Motion CPU.
A part for the number of reading data of the control data specified with (S1) of data since the device specified with (S2) in the target CPU (n1) is stored to since the word device specified with (D1) of the self CPU in the Multiple CPU system.

(2) Figure specification of the bit device is possible for (S2) and (D1). However, figure specification is 4 figures and a start bit device number is only the multiple of 16. It becomes INSTRCT CODE ERROR [4004] when other values are specified.

(3) If the target CPU is not instruction acceptable condition, even if the S(P).DDWR instruction is executed, it may not be processed. In this case, it is necessary to execute the S(P).DDWR instruction again.
(S(P).SFCS/S(P).SVST/S(P).CHGA/S(P).CHGV/S(P).CHGT/S(P).DDRD/ S(P).DDWR cannot be executed simultaneously toward the CPU executing S(P).DDWR instruction.). It can be confirmed by data in the shared CPU memory of the target CPU (Motion CPU) whether the instruction is acceptable or not.
When the Motion dedicated PLC instruction is started continuously, it is must be design to execute next instruction after executing instruction complete device on.

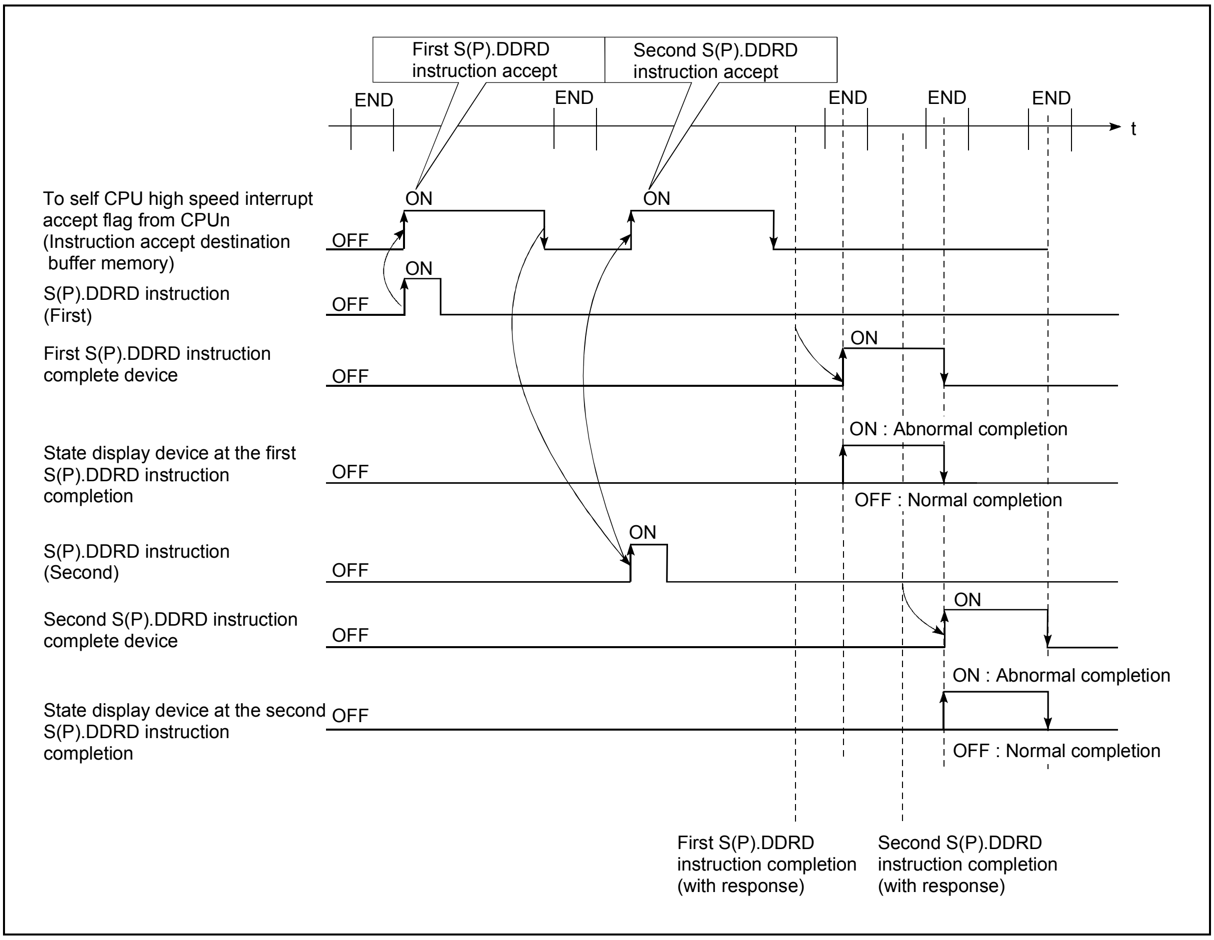
(4) The target CPU device range check is not executed with self CPU at the S(P).DDRD instruction execution, but it checks by the target CPU side, and it becomes abnormal completion at the device range over.

(5) S(P).DDRD instruction accepting and normal/abnormal completion can be confirmed with the complete device (D1) or status display device (D2) at the completion.
   (a) Complete device
   It is turned on by the END processing of scan which the instruction completed, and turned off by the next END processing.

   (b) Status display device at the completion
   It is turned on/off according to the status of the instruction completion.
   - Normal completion : OFF
   - Abnormal completion : It is turned on by the END processing of scan which the instruction completed, and turned off by the next END processing.

(6) SM390 turns on when the target CPU specified with (n1) complete to accept.
SM390 turns off when the target CPU specified with (n1) cannot be write correctly by the reset status or error factor (5000 to 5999).

[Operation]



[Errors]

The abnormal completion in the case shown below, and the error code is stored in the control data (S1+ 0 : Complete status).

Complete status (Note) (Error code)(H)	Error factor	Corrective action
4C00	The specified device cannot be used in the Motion CPU. Or, it is outside the device range.	Confirm a program, and correct it to a correct PLC program.
4C08	There are 33 or more instruction requests to the Motion CPU from the PLC CPU in S(P).DDRD/S(P).DDWR sum table simultaneously, and the Motion CPU cannot process them.	
4C09	CPU No. of the instruction cause is injustice.	

(Note) : 0000H (Normal)

The error flag (SM0) is turned on an operation error in the case shown below, and an error code is stored in SD0.

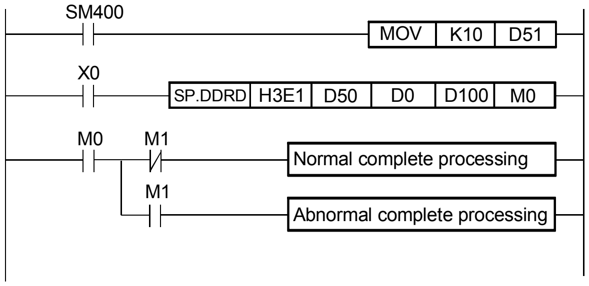
Error code (Note)	Error factor	Corrective action
2110	The CPU No. to be set by "(First I/O No. of the target CPU)/16" is specified.	Confirm a program, and correct it to a correct PLC program.
2114	The self CPU by "(First I/O No. of the target CPU)/16" is specified.	
2117	The CPU except the Motion CPU by "(First I/O No. of the target CPU)/16" is specified.	
4002	Specified instruction is wrong.	
4004	The instruction is composed of devices except usable devices.	
4100	Since 0 to 3DFH, 3E4H is specified by "(First I/O No. of the target CPU)/16" is specified.	
4101	Number of the writing data is except 1 to 16.	
	Number of writing data exceeds range of the storage device of the written data.	

(Note) : 0000H (Normal)

[Program example]

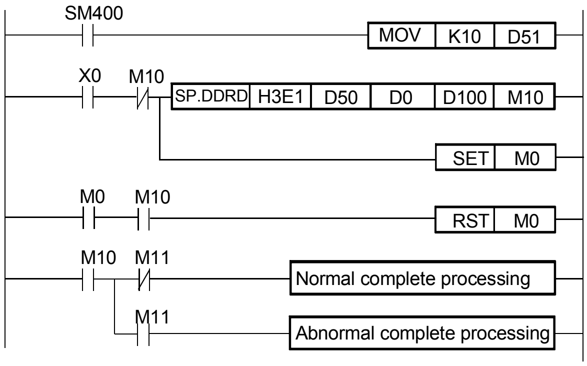
<Example 1>

Program which stores 10 points worth of the data from D0 of the CPU since D100 of self CPU (CPU No.1), when X0 is turned on.



<Example 2>

Program stores 10 points worth of the data from D0 of the CPU No.2 since D100 of self CPU (CPU No.1) during turn on X0..

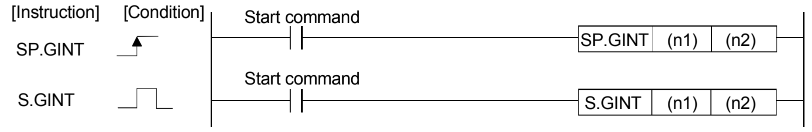


3.9 Interrupt Instruction to The Other CPU: S(P).GINT (PLC instruction: S(P).GINT)

Setting data (Note)	Usable devices										
	Internal devices (System, User)		File register	Bit Digit specified	Indirectly specified device	MELSECNET/10 direct J□\□		Special function module U□\G□	Index register Z□	Constant K, H	Other
	Bit	Word				Bit	Word				
(n1)		○	○	○	○					○	
(n2)		○	○	○	○					○	

○ : Usable △ : Usable partly

(Note) : Setting data (n1) to (D2) : Index qualification possible



[Setting data]

Setting data	Description	Data type
(n1)	(First I/O No. of the target CPU)/16 Value to specify actually is the following. (Note-1) CPU No.1 : 3E0H, CPU No.2 : 3E1H, CPU No.3 : 3E2H, CPU No.4 : 3E3H	16-bit binary
(n2)	Interrupt instruction No. (0 to 15)	16-bit binary

(Note-1) : Motion CPU cannot used CPU No.1 in the Multiple CPU configuration.

[Controls]

This instruction generates the interrupt to the Motion CPU by PLC program when the execution instruction of S(P).GINT is started (OFF → ON).
The Motion CPU executes the active program (operation program status) processing of the Motion SFC program set by "PLC interruption of the event task" at the interrupt generation from the PLC CPU.

(1) This instruction is always effective regardless of the state of real mode/virtual mode/mode switching when the operating system software of Motion CPU is SV22.

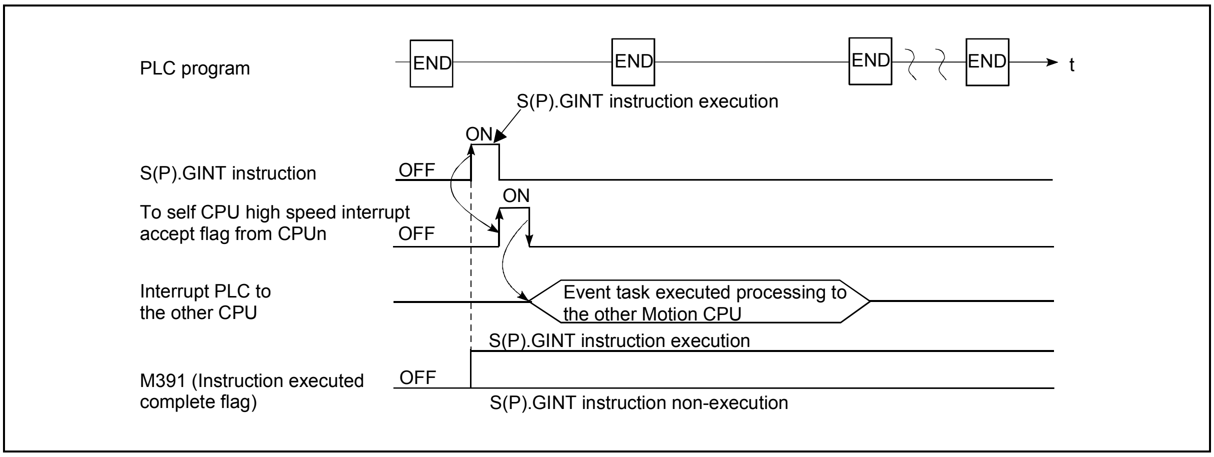
(2) Motion CPU side is during DI (interrupt disable), event processing can make wait even as for the EI (interrupt enable) instruction execution.

(3) SM390 turn on when the transmission of the instruction toward the target CPU was completed. SM391 (S(P).GINT instruction execution completion flag) turned on simultaneously.

(4) SM390 turn off when the transmission of the instruction toward the target CPU was not completed. SM391 (S(P).GINT instruction execution completion flag) turned off when the instruction toward the target CPU cannot be transmitted.

(5) Number of instruction execution does not have restriction, if to self CPU high speed interrupt accept flag from CPUn in the target shared CPU memory of S(P).GINT instruction.

[Operation]



[Errors]

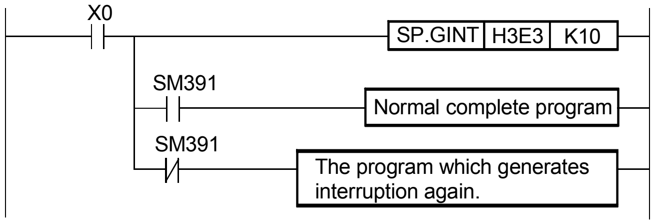
The error flag (SM0) is turned on an operation error in the case shown below, and an error code is stored in SD0.

Complete status (Note) (Error code)(H)	Error factor	Corrective action
2110	The CPU No. to be set by "(First I/O No. of the target CPU)/16" is specified.	Confirm a program, and correct it to a correct PLC program.
2114	The self CPU by "(First I/O No. of the target CPU)/16" is specified.	
2117	The CPU except the Motion CPU by "(First I/O No. of the target CPU)/16" is specified.	
4100	Since 0 to 3DFH, 3E4H is specified by "(First I/O No. of the target CPU)/16" is specified.	
4C08	There are 33 or more instruction requests to the Motion CPU from the PLC CPU in S(P).SFCS/S(P).SVST/ S(P).CHGA/S(P).GINT sum table simultaneously, and the Motion CPU cannot process them.	

(Note) : 0000H (Normal)

[Program example]

Program which generates the interrupt toward the Motion CPU No.4.

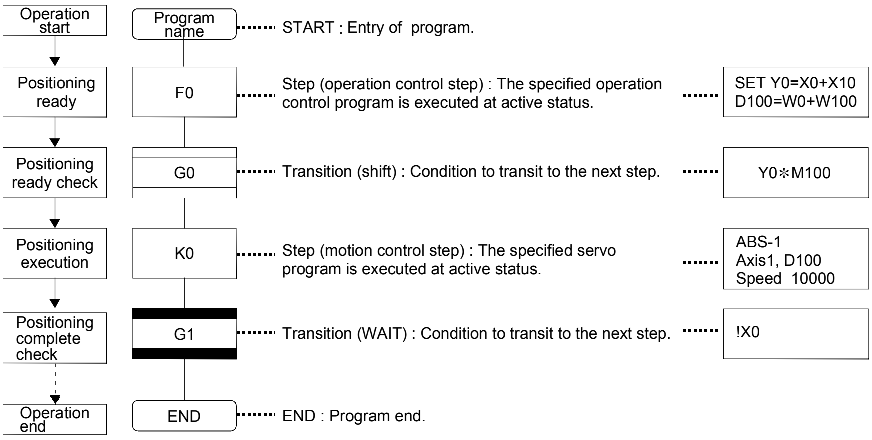


4. MOTION SFC PROGRAMS

Refer to Chapter "11 ERROR CODE LISTS" for details of Motion SFC program error.

4.1 Motion SFC Program Configuration

The Motion SFC Program is constituted by the combination of start, steps, transitions, end and others are shows below.



The above Motion SFC program to be started performs the following operations.

(1) The step (F0) is activated and the operation specified with the step (F0) is executed (positioning ready). A step in such an active state is called an active step.

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

(3) The operating completion of the step (K0) (positioning completion of the servo program K0) is checked, and control transits to the next step at operating completion (completion of condition).

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

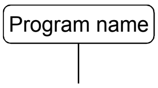
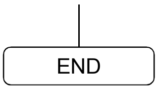
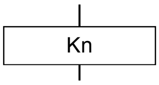
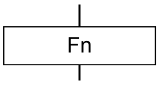
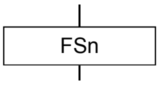
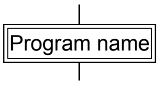
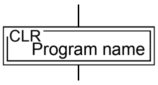
Refer to Section "9.2.2 Task operation" for details of the execution timing of the Motion SFC program such as above.

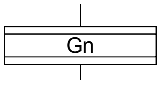
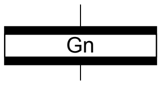
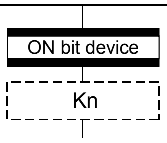
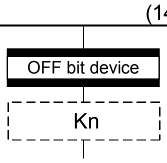
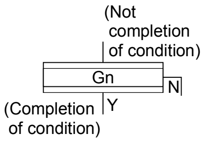
POINT

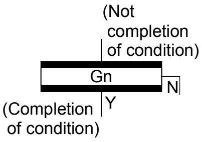
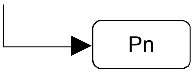
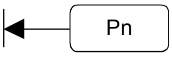
The number of steps which can be active steps simultaneously is up to 256, with those of all Motion SFC programs combined. Excess of 256 will result in the Motion SFC Program error 16120.
Each symbol of the Motion SFC program is as follows.
F/FS : Operation control, K : Positioning control, G : Judgment

4.2 Motion SFC Chart Symbol List

Parts as Motion SFC program components are shown below. The operation sequence or transition control is expressed with connecting these parts by directed lines in the Motion SFC program.

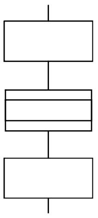
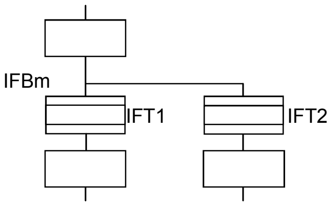
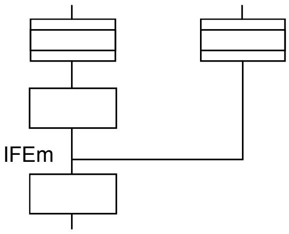
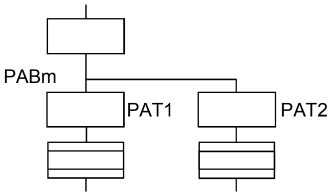
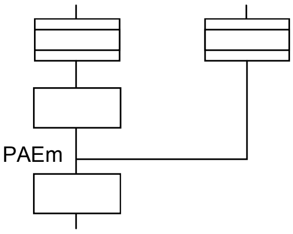
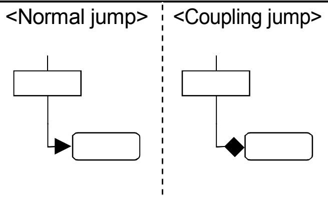
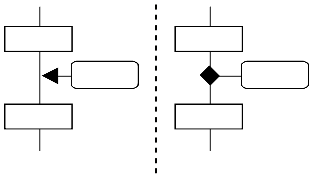
Classification	Name	Symbol (Code size (byte))	List Representation	Function
Program start/end	START	 (0)	Program name	• Indicates an entry of program as a program name. • Specify this program name at a subroutine call. • Only one program name for one program.
	END	 (8)	END	• Indicates an end (exit) of program. • When a subroutine call was carried out, returns to the call source program. • Multiple program names or no symbols for one program.
Step	Motion control step	 (8)	CALL Kn	• Starts a servo program Kn (K0 to K4095).
	Once execution type operation control step	 (8)	CALL Fn	• Execute once the operation control program Fn (F0 to F4095).
	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	• When the next of GSUB is WAIT, performs "subroutine call" and transits control to the specified program. Control returns to the call source program at END execution. • When the next of GSUB is except WAIT, performs "subroutine start", and starts the specified program and transits to the next (lower part). The start source and destination programs are executed simultaneously, and the call destination program ends at END execution.
	Clear step	 (8)	CLR program name	• Stops and ends the specified program running. After an end, it is started from the initial (start step) by restarting the program. • When the specified program is during "subroutine call", the subroutine program is also stopped to execute. • When the specified program is after "subroutine start", the subroutine program is not stopped to execute. • When clearing to the subroutine by which the "subroutine call" was executed, the specified subroutine is stopped to execute, returns to the call source program, and transits to the next.

Classification	Name	Symbol (Code size (byte))	List representation	Function
Transition	Shift (Pre-read transition)	 (8)	SFT Gn	• When just before is the motion control step, transits to the next step by formation of transition condition Gn (G0 to G4095) without waiting for the motion operating completion. • When just before is the operation control step, transits to the next step by the completion of transition condition after operating execution. • When just before is subroutine call or starting step, transits to the next step by formation of transition condition without waiting for the operating completion of subroutine.
	WAIT	 (8)	WAIT Gn	• When just before is the motion control step, waits for the motion operating completion and then transits to the next step by the completion of transition condition Gn (G0 to G4095). • When just before is the operation control step, transits to the next step by formation of transition condition after operating execution. (Same operation as Shift.) • When just before is subroutine call or starting step, waits for the operating completion of subroutine and then transits to the next step by the completion of transition condition.
	WAITON	 (14)	WAITON bit device	• Prepares for starting of the next motion control step, and issues an instruction immediately when the specified bit device turns ON. • Always pair this transition with the motion control step one-for-one.
	WAITOFF	 (14)	WAITOFF bit device	• Prepares for starting of the next motion control step, and issues an instruction immediately when the specified bit device turns OFF. • Always pair this transition with the motion control step one-for-one.
	Shift Y/N		IFBm IFT1 SFT Gn : JMP IFEm IFT2 SFT Gn+? : JMP IFEm IFEm	• When just before is the motion control step, transits to the next step by formation of transition condition Gn (G0 to G4095) without waiting for the motion operating completion. If not formation of transition condition, transits to the right-connected step. • When just before is the operation control step, transits to the next step by the completion of transition condition after operating execution. If not the completion of transition condition, transits to the right-connected step. • When just before is "subroutine call" or "starting step", transits to the next step by the completion of transition condition without waiting for the operating of subroutine completion. If not formation of transition condition, transits to the right-connected step.

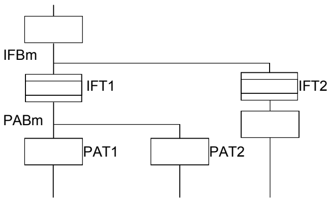
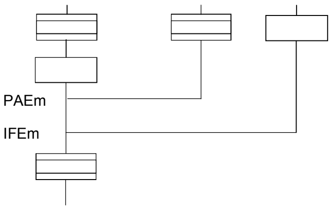
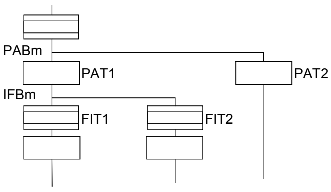
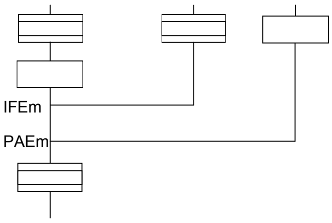
Classification	Name	Symbol (Code size (byte))	List representation	Function
Transition	WAIT Y/N		IFBm IFT1 WAIT Gn : JMP IFEm IFT2 WAIT Gn+? : JMP IFEm IFEm	• When just before is the motion control step, waits for the motion operating completion and then transits to the next step by formation of transition condition Gn (G0 to G4095). If not completion of transition condition, transits to the right-connected step. • When just before is the operation control step, transits to the next step by the completion of transition condition after operating execution. If not the completion of transition condition, transits to the right-connected step. (Same operation as Shift.) • When just before is subroutine call or starting step, waits for the operating completion of subroutine, and then transits to the next step by the completion of transition condition. If not formation of transition condition, transits to the right-connected step.
Jump	Jump	 (14)	JMP Pn	• Jumps to the specified pointer Pn (P0 to P16383) of the self program.
Pointer	Pointer	 (8)	Pn	• 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 one program. The same No. may also be used in other programs.

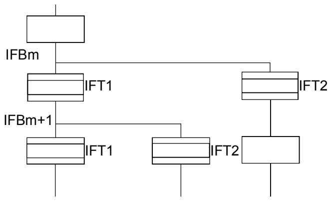
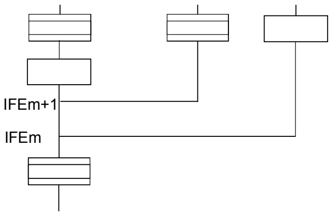
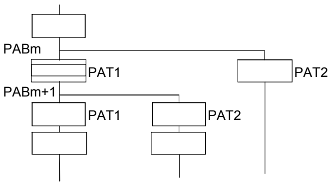
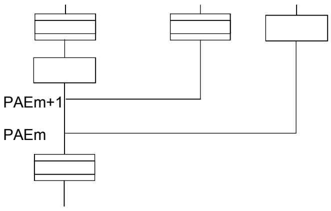
4.3 Branch and Coupling Chart List

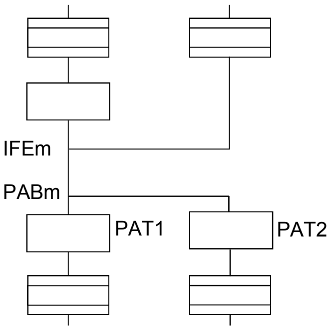
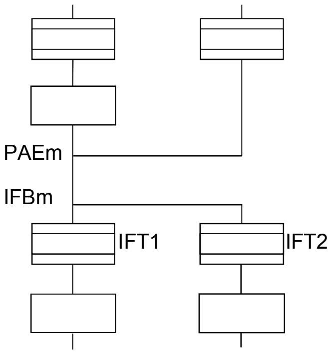
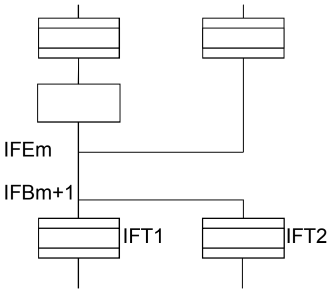
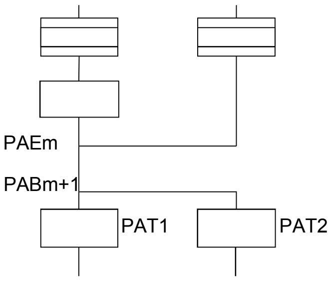
Branch and coupling patterns which specify step and transition sequences in the Motion SFC charts are shown below.

	Name (Code size (byte))	Motion SFC chart symbol	List representation	Function
Basic type	Series transition (Corresponding symbol size)		List representation corresponding to the Motion SFC chart symbols shown in Section 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 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''	• The route which transition condition enables first is executed after executing the step or transition preceding a branch. • 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)		CALL Kn PABm PAT1 CALL Fn SFT Gn' : JMP PAEm PAT2 CALL Fn' SFT 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)		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 completion of transition condition of the jump destination. 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 CALL Kn	

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

	Name	Motion SFC chart symbol	List representation	Function
Appli-cation type	Selective branch | Parallel branch		CALL Kn IFBm IFT1 SFT Gn PABm PAT1 CALL Fn : JMP PAEm PAT2 CALL Fn’ : (JMP PAEm) PAEm JMP IFEm IFT2 SFT Gn’ CALL Fn” : (JMP IFEm) IFEm SFT 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 the Motion SFC chart, this type is displayed in order of a parallel coupling → a selective coupling, as shown on the left. • In this case, a pointer (Pn) cannot be set between the parallel coupling point (PAEm) and the selective coupling point (IFEm).
	Parallel branch | Selective branch		SFT Gn PABm PAT1 CALL Fn IFBm IFT1 SFT Gn’ CALL Fn’ : JMP IFEm IFT2 SFT Gn” CALL Fn” : (JMP IFEm) IFEm JMP PAEm PAT2 CALL Fn’” : CALL Kn (JMP PAEm) PAEm SFT 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 the Motion SFC chart, this type is displayed in order of a selective coupling → a parallel coupling, as shown on the left. • In this case, a pointer (Pn) cannot be set between the selective coupling point (IFEm) and the parallel coupling point (PAEm).

	Name	SFC chart symbol	List representation	Function
Appli- cation type	Selective branch | Selective branch		CALL Kn IFBm IFT1 SFT Gn IFBm+1 IFT1 SFT Gn’ : JMP IFEm+1 IFT2 SFT Gn” : (JMP IFEm+1) IFEm+1 JMP IFEm IFT2 SFT Gn”’ CALL Fn’ : (JMP IFEm) IFEm SFT 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 the Motion SFC chart, this type is displayed in order of a selective coupling → a selective coupling, as shown on the left. • In this case, a pointer (Pn) cannot be set between the selective coupling point (IFEm+1) and the selective coupling point (IFEm).
	Parallel branch | Parallel branch		CALL Kn PABm PAT1 SFT Gn 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 SFT 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 the Motion SFC chart, this type is displayed in order of a parallel coupling → a parallel coupling, as shown on the left. • In this case, a pointer (Pn) cannot be set between the parallel coupling point (PAEm+1) and the parallel coupling point (PAEm).

	Name	SFC chart 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 the Motion SFC chart, this type is displayed in order of a selective coupling → a parallel branch, as shown on the left. • In this case, a pointer (Pn) cannot be set between the selective coupling point (IFEm) and the parallel branch point (PABm).
	Parallel coupling | Selective branch		: JMP PAEm PAEm IFBm IFT1 SFT Gn : JMP IFEm IFT2 SFT Gn' : (JMP IFEm) IFEm :	• The parallel coupling point and selective branch point can be the same. Note that in the Motion 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, a pointer (Pn) cannot be set between the parallel coupling point (PAEm) and the selective branch point (IFBm).
	Selective coupling | Selective branch		: (JMP IFEm) IFEm IFBm+1 IFT1 SFT Gn : JMP IFEm+1 IFT2 SFT Gn' : (JMP IFEm+1) IFEm+1	• The selective coupling point and selective branch point can be the same. Note that in the Motion SFC chart, this type is displayed in order of a selective coupling → a selective branch, as shown on the left. • In this case, a pointer (Pn) cannot be set 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 the Motion 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, a pointer (Pn) cannot be set between the parallel coupling point (PAEm) and the parallel branch point (PABm+1).

4.4 Motion SFC Program Name

Set the "Motion SFC program name" to the Motion SFC program No.0 to No.255 individually. (Make this setting in the "Motion SFC program management window" on the Motion SFC program edit screen.)

Set the Motion SFC program name within 16 characters. Specify this Motion SFC program name for a "subroutine call/start step (GSUB)" and "clear step (CLR)".

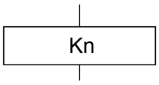
Motion SFC programs correspond to No.0 to No.255 and saved in a one program-for-one file format. The preset "Motion SFC program name" is used as the file name of the Motion SFC Program file for user file management. (Refer to Chapter "10 USER FILES" for details.)

POINT

(1) It is can be set the Motion SFC program to any of No.0 to No.255. There are no specific programs which have special roles.
(2) "$" cannot be used in the first character of the Motion SFC program name.
(3) " / : ; , . * ? " < > |" cannot be used in Motion SFC program name.

4.5 Steps

4.5.1 Motion control step

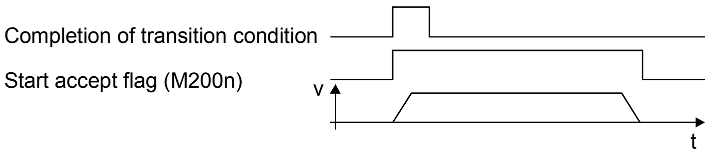
Name	Symbol	Function
Motion control step	 Kn	Starts the servo program Kn. Specified range: K0 to K4095

[Operations]

(1) Turns on the start accept flag of the axis specified with the specified servo program Kn (n = 0 to 4095) runnnig.

(2) Starts the specified servo program Kn (n = 0 to 4095).

Execution timing



[Errors]

(1) When the specified servo program Kn does not exist, the Motion SFC program error [16200] will occur and stops to execute the Motion SFC program at the error detection.

[Instructions]

(1) When the current value change is executed in the Motion SFC program running, 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 with the motion control step, the Motion SFC program continues executing. When the Motion SFC program is stopped 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	Fn/FSn	Executes the operation control program Fn/FSn. Specified range: F0 to F4095/FS0 to FS4095

[Operations]

(1) Once execution type operation control step Fn
In the case of Fn, executes the specified operation control program Fn (n = 0 to 4095) once.

(2) Scan execution type operation control step FSn
In the case of FSn, repeats the specified operation control program FSn (n =0 to 4095) until the next transition condition enables.

[Errors]

(1) When the specified operation control program Fn/FSn does not exist, the Motion SFC program error [16201] will occur and stops to execute the Motion SFC program at the error detection.

[Instructions]

(1) Refer to Chapter "5 OPERATION CONTROL PROGRAMS" for operation expressions that may be described in operation control programs.

(2) If an operation or similar error occurs the operation control program running, the Motion SFC program continues executing.

4.5.3 Subroutine call/start step

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

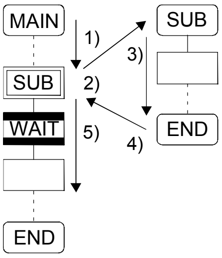
[Operations]

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

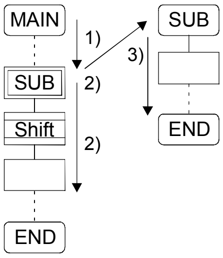
(2) Control varies with the type of the transition coupled next to the subroutine call/start step.

(a) WAIT (Subroutine Call)
When the subroutine call step is executed, control transits to the specified program as shown below, and when END of the called program is executed, control returns to the call source program.

(b) Except WAIT (Subroutine Start)
When the subroutine start step is executed, control starts the specified program and then shifts to the next as shown below. Since, the start source and destination Motion SFC programs are executed in parallel. The started program ends at END execution.



WAIT
(Subroutine call)



Except WAIT
(Subroutine start)

[Errors]

(1) When the specified Motion SFC program does not exist at a subroutine call/start, the Motion SFC program error [16005] will occur and stops to execute the Motion SFC program at the error detection.

(2) When the called/started Motion SFC program is already starting at a subroutine call/start, the Motion SFC program error [16006] will occur and stops to execute the Motion SFC program at the error detection.

(3) When the self program is started at a subroutine call/start, the Motion SFC program error [16110] will occur and stops to execute the Motion SFC program at the error detection.

(4) When the subroutine to be called/started at a subroutine call/start in the Motion SFC program 2 running which was called/started from the Motion SFC program 1 is the Motion SFC program 1 (call source/start program), the Motion SFC program error [16111] will occur and the call/start source Motion 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 Motion SFC program continues processing if the start destination Motion SFC program stops due to an error.

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

4.5.4 Clear step

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

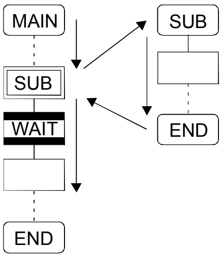
[Operations]

(1) Stops the specified Motion SFC program running.

(2) The clear-specified Motion SFC program will not start automatically after stopped if it has been set to start automatically.

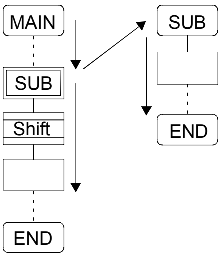
(3) The specified program may be its self program.

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



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

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



If the program has been "subroutine started" as shown on the left
- When the started program (MAIN) is cleared. Even if the subroutine (SUB) is running, the started 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) When the Motion SFC program specified with the clear step does not exist, the Motion SFC program error [16203] will occur.

[Instructions]

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

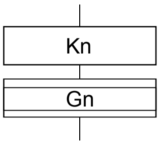
(2) If the Motion SFC program running is stopped by the clear step, the output is held.

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.
Refer to Chapter "6 TRANSITION PROGRAMS" for the conditional/operation expressions that can be described in transition conditions.

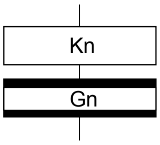
(1) Combinations with motion control steps

(a) Motion control step + Shift



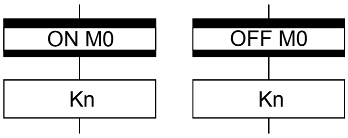
[Operations]
- Transits to the next step by formation of transition condition Gn without waiting for the operating completion of the servo program Kn started at the motion control step.

(b) Motion control step + WAIT



[Operations]
- Waits for the operating completion of the servo program Kn started at the motion control step, and then transits to the next step by formation of transition condition Gn.
- 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



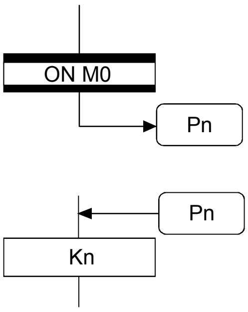
[Operations]
- Prepares for the start of the motion control step next to WAITON/WAITOFF, and makes a start immediately when the specified bit device turns ON/OFF. When the motion control step is executed without being used with WAITON/WAITOFF, preparations for a start are made after the transition condition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

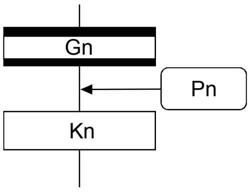
- Specifiable bit devices

Device	Range
X	X0 to X1FFF
Y	Y0 to Y1FFF
M	M0 to M8191
Special relay	M9000 to M9255
L	L0 to L8191
B	B0 to B1FFF
F	F0 to F2047

[Instructions]

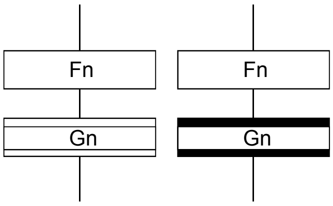
- Always pair a transition with a motion control step one-for-one. If the step following WAITON/WAITOFF is not a motion control step, the Motion SFC program error [16102] will occur and the Motion SFC program running will stop at the error detection.
- An error will not occur if the jump destination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





- If the servo program specified with a motion control step could not be started due to a major/minor error, the Motion SFC program continues running and execution shifts to the next, independently of the WAITON/WAITOFF bit device status. To stop the Motion SFC program at error detection, provide an error detection condition at the next transition (transition condition).
- The following instructions can be used in the motion control step used combining the WAITON/WAITOFF.
  (Linear interpolation control, circular interpolation control, helical interpolation, speed switching control, position follow-up control, constant-speed control and high speed oscillation.)

(2) Combination with operation control step



[Operations]

- At an operation control step, both Shift and WAIT perform the same operation, and after executing of the operation control program Fn, transits to the next step by formation of transition condition Gn.

(3) Combination with subroutine call/start step

Refer to Section "4.5.3 Subroutine call/start step".

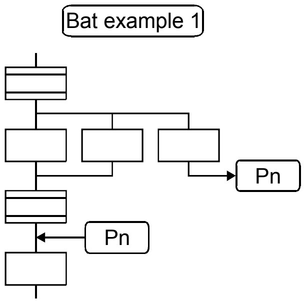
4.7 Jump, Pointer

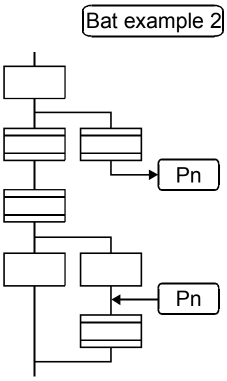
[Operations]

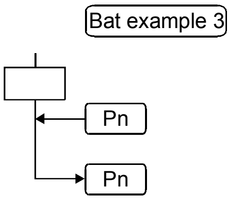
- Setting a jump will cause a jump to the specified pointer Pn of the self program.
- You can set pointers at steps, transitions, branch points and coupling points.
- You can set pointers Pn at P0 to P16383 in one program.

[Instructions]

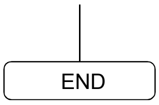
- You cannot make a jump setting which will exit from within parallel branch-parallel coupling. Connect directly. (Bad example 1 given below)
- You cannot make a jump setting from outside parallel branch-parallel coupling to within parallel branch-parallel coupling. (Bad example 2 given below)
- You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







4.8 END



[Operations]

- Ends a program. (In this case of an event task or NMI task, operation changes with end operation setting of the program parameter. Refer to Section "9.5 Program Parameters" for details.)
- Making a subroutine call will return to the call source Motion SFC program.

[Instructions]

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

4.9 Branches, Couplings

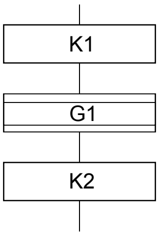
4.9.1 Series transition

Transits 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.



Starts the servo program K1.

Transits to next by the completion of condition set at transition G1 without waiting for operating completion of the servo program K1.

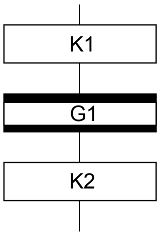
Starts the servo program K2.

POINT

For a subroutine start, self 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.



Starts the servo program

Transits to next when the start axis stops in the servo program K1 (start accept flag turns OFF) and condition is completed set at transition G1.

Starts servo program K2.

POINT

(1) The above start accept 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.

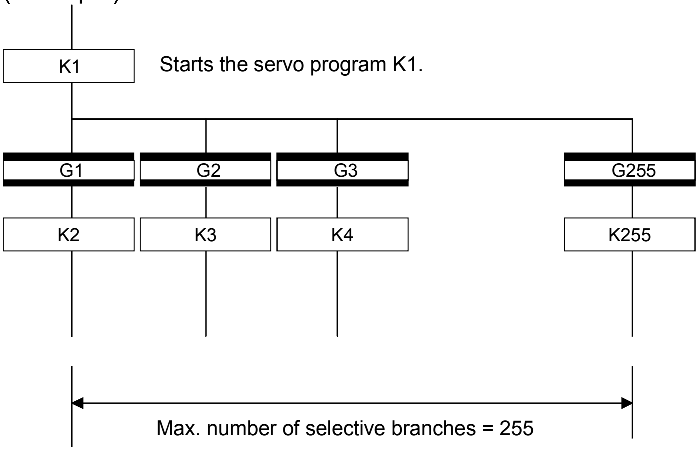
(2) WAIT must be set to proceed to the next step on operation completion.
However, when there are specifically no conditions to be set as interlocks, set "NOP (No Operation)" in the transition program (Gn).

4.9.2 Selective branch, selective coupling

(1) Selective branch

Executes only the route which condition was judged to have enabled first among the conditions of multiple transitions connected in parallel. Transitions must be all Shifts or WAITs.

(Example) WAIT



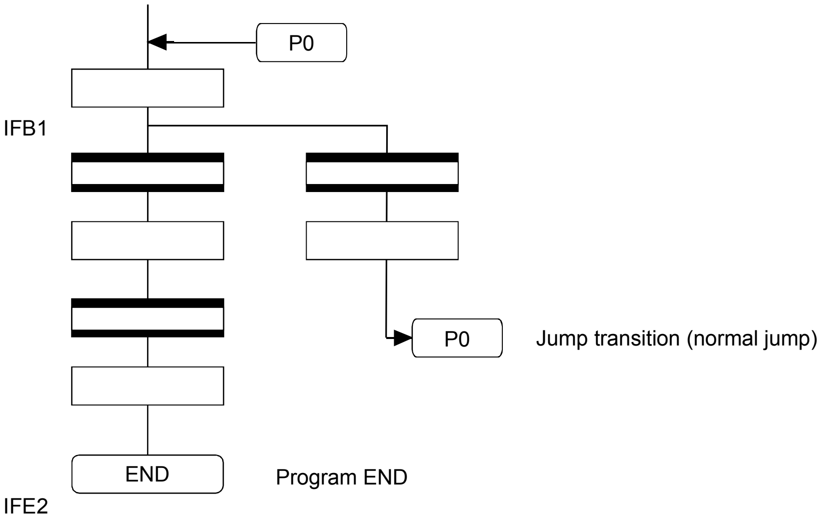
After start axis in the servo program K1 has stopped (start accept flag turns OFF), the conditions of transitions G1 to G255 are judged, and execution transits to route which condition is completed.

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

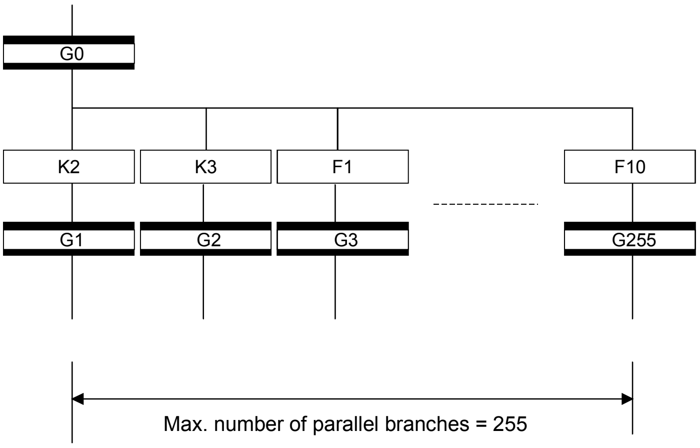
Recoupling of routes into a single route after their processing completions following a selective branch will be a selective coupling. However, you can also make a setting where no coupling will be made as shown below.



4.9.3 Parallel branch, parallel coupling

(1) Parallel branch

Multiple routes connected in parallel are executed simultaneously. Each parallel branch destination may be started by either a step or a transition.



After operation completion of preceding step, steps K2 to F10 connected in parallel are executed when the completion of condition set at transition G0. Thereafter, routes are executed simultaneously up to parallel coupling point.

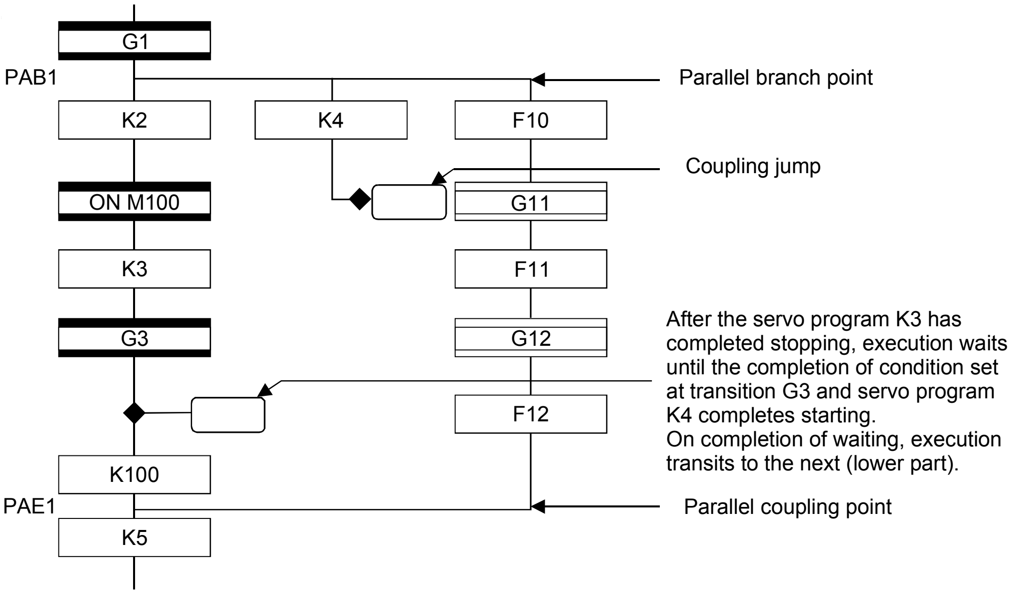
POINT

"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.



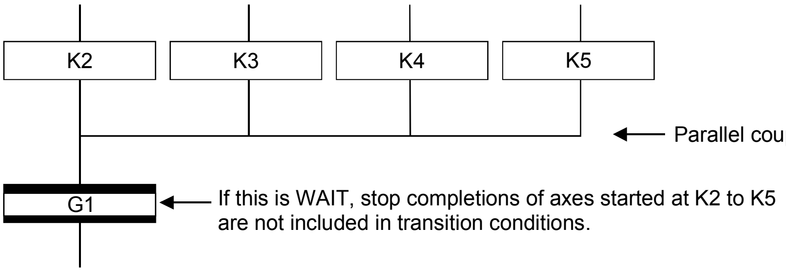
After the servo program K3 has completed stopping, execution waits until the completion of condition set at transition G3 and servo program K4 completes starting.
On completion of waiting, execution transits to the next (lower part).

POINT

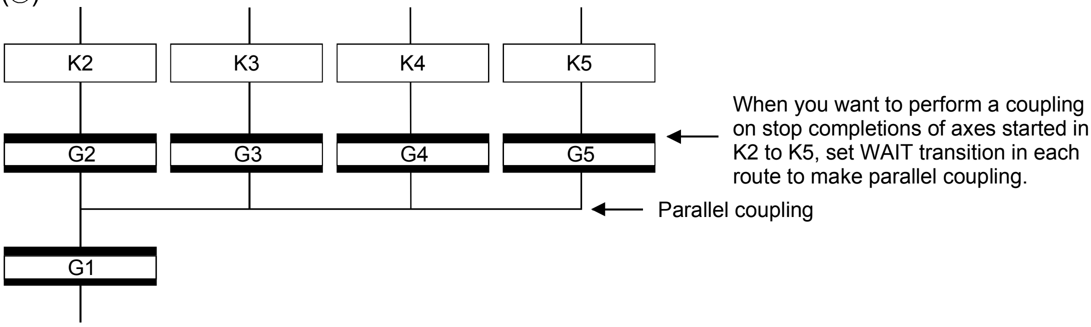
The number of parallel branches need not match that of couplings at a parallel coupling point.
(In the example of the diagram in Section 4.9.3 (2), the number of parallel branches is 3 and that of couplings is 2.)

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.

(×)

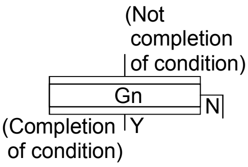
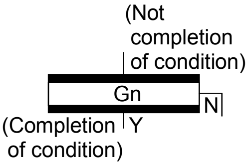


(○)



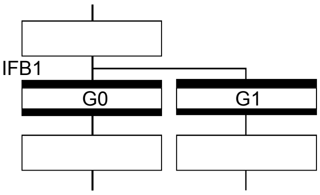
4.10 Y/N Transitions

When routes are branch at a transition condition enables and disable, "Shift Y/N transition" or "WAIT Y/N transition" will be useful.

Name	Symbol	Function
Shift Y/N transition		• When a transition condition set at Gn enables, execution shifts to the lower step. When that condition disables, 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		

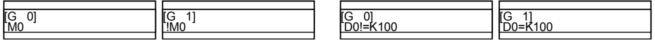
A Y/N transition is designed to describe the following two-route selective branch program easily.

<Y/N transition is not used.>



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

(Example 1) (Example 2)



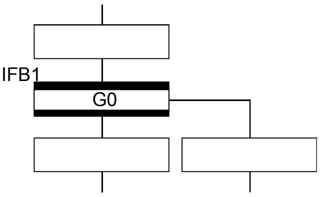
[G 0]
M0

[G 1]
!M0

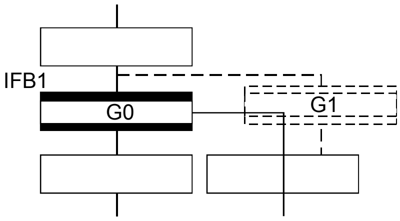
[G 0]
D0!=K100

[G 1]
D0=K100

<Y/N transition is used.>



- Set the G0 program shown in above (Example 1) or (Example 2) as a G0 program.
- The Motion SFC program list codes after conversion are the same as in the conventional description (different only in the Motion SFC chart representation). Therefore, "automatic search for free G number → automatic generation of program which 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 shown below.

<Setting 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 shown below.

<Setting 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

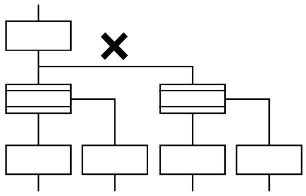
Refer to Section "1.2.3 (2) Table of the operation control/transition instruction" for the instructions usable in the conditional expressions of "Shift Y/N" or "WAIT Y/N" transition programs.

(3) Instructions for the Motion SFC charts

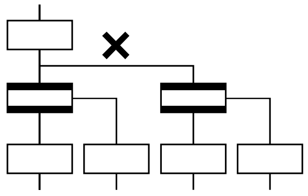
Any Motion SFC chart that will be meaningless to or conflict with the definition of Y/N transitions will result in an error at the time of editing (or Motion SFC chart conversion). Their patterns and instructions will be given below.

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

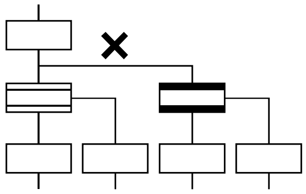
- "Shift Y/N" used as selective branch



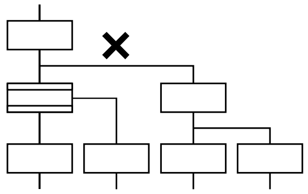
- "WAIT Y/N" used as selective branch



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

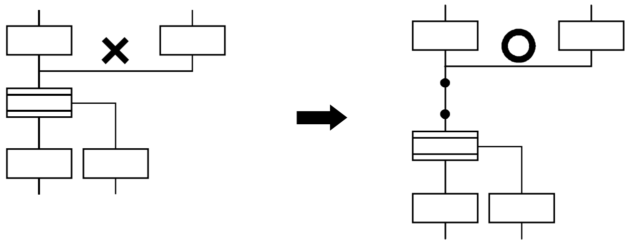


- "Shift (or WAIT) Y/N" used with other step/transition as parallel branch or selective branch



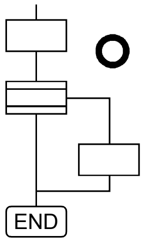
(b) When a coupling precedes "Shift Y/N" or "WAIT Y/N": Provide "coupling-branch continuation" in between.

- Direct coupling with "Shift Y/N" or "WAIT Y/N" is not allowed.
- Provide "coupling-branch continuation" in between.

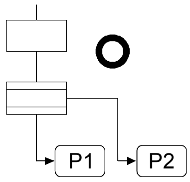


(c) The following patterns may be set.

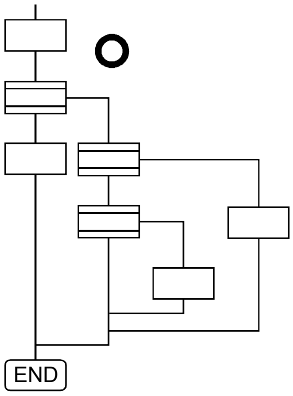
- End (END) from "Shift Y/N" or "WAIT Y/N"



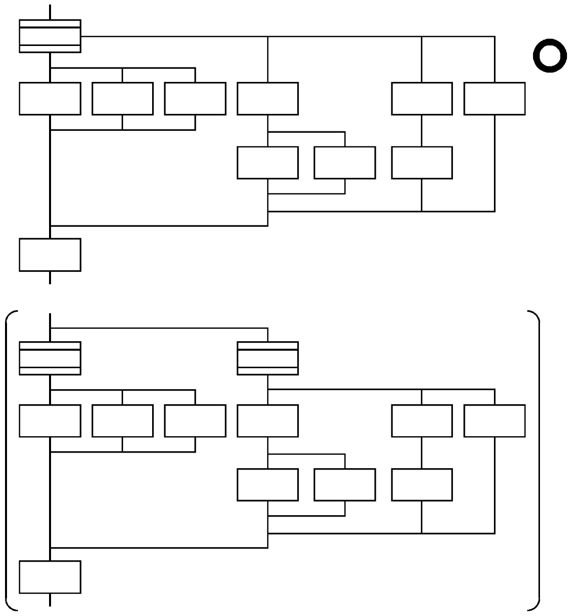
- Jump from "Shift Y/N" or "WAIT Y/N"



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

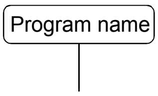
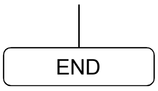
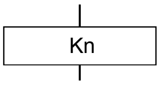
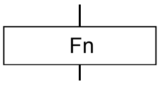
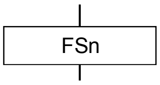
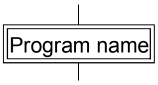
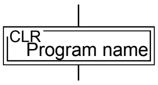
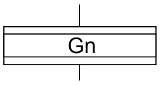
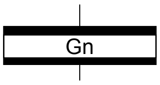
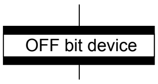
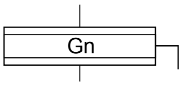
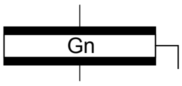
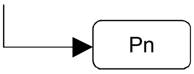
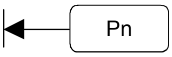


- When there are two or more connection lines from Y/N side of "Shift Y/N" or "WAIT Y/N", selective branch continues to selective branch or parallel branch.



4.11 Motion SFC Comments

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

Classification	Name	Symbol	Comment Setting
Program start/end	START		Comment setting cannot be made.
	END		
Step	Motion control step		Up to 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		Up to 64 characters Displayed in 16 characters × 4 lines
Pointer	Pointer		

POINT

(1) Motion SFC comments are stored into the CPU code area. The CPU code area stores the Motion SFC chart codes, operation control (F/FS) program codes, transition (G) program codes and Motion SFC comments.
Be careful not to set too many comments to avoid code area overflow. (Refer to Section "1.2.2 (1) (b) Motion SFC Performance Specifications" for the code area sizes.)
(2) You cannot use "," in comment statements.

MEMO

5. OPERATION CONTROL PROGRAMS

Refer to Section "11.2 Motion SFC Error Code List" for error codes of the operation error.
(Refer to the "Q173HCPU/Q172HCPU Motion controller (SV13/SV22) Programming Manual (REAL MODE)" and "Q173HCPU/Q172HCPU Motion controller (SV22) Programming Manual (VIRTUAL MODE)" for minor errors of the operation error.)

5.1 Operation Control Programs

(1) Operation control programs

(a) Substitution operation expressions, motion-dedicated functions and bit device control commands can be set in operation control program.

(b) Multiple blocks in one operation control program can be set.

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

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

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

(f) The bit conditional expression that logical data value (true or false) is returned in an operation control program, a comparison conditional expression can be set up only as a source (S) of device set (SET=) or device reset (RST=).

An operation control program example is shown below.

1 block

```
#0=D0+(D1+D2)*#5//Substitution expression (four arithmetic operations)
W0:F=SIN(#10F)//Substitution expression (standard function)
CHGV(K2,K10)//Motion-dedicated function
SET M100=M0+X0//Bit device control (SET=)
RST M10=!X0//Bit device control (RST=)
DIN D0,X0//Bit device control (DIN)
```

1 program

Comment

(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	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 (+)
Low	Substitution (=)

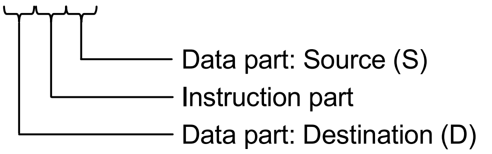
(3) Structure of instruction

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.

"Substitution: =" structure example



(a) Source (S)

1) The source is the data used in an operation.
2) It varies with the device specified in each instruction is shown 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 execution allows changing the data used in that instruction.
   - Constant

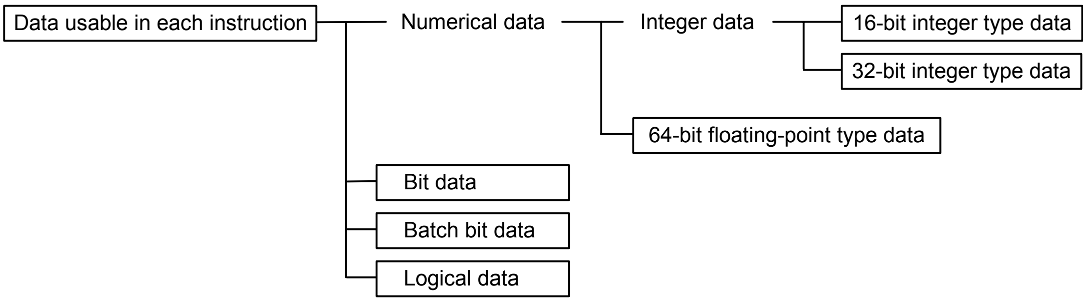
     Specify the numerical value used in an operation.
     As the constant is set during program creation, it cannot be changed during program running.

(b) Destination (D)

1) As the destination data, after-operation data is stored.
2) Destination data is 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 integer value data.
Word devices are used in increments of 1 point.
Data ranges are shown below.

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

(b) 32-bit integer type data

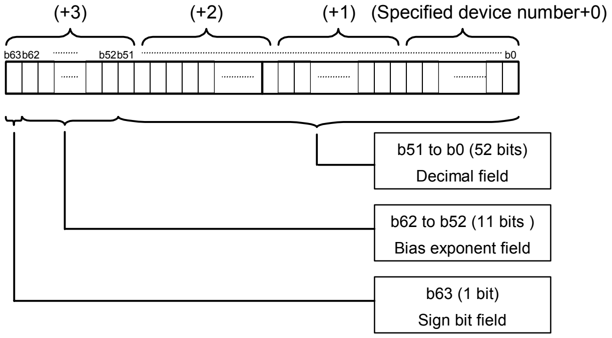
The 32-bit integer type data is 32-bit integer value data.
Word devices are used in increments of 2 points: (specified device No.), (specified device No.+1). Data ranges are shown below.

	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 No.), (specified device No.+1), (specified device No.+2), (specified device No.+3).

1) The internal bit locations are shown below.



2) The represented value is shown below. (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 shown 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 1

```
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 No.) to (specified device No.+15) M0 to M15 in the above program example	(Specified device No.) to (specified device No.+31) M0 to M31 in the above program example

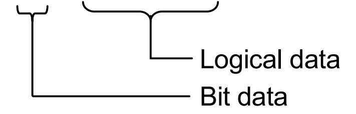
(f) Logical data

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

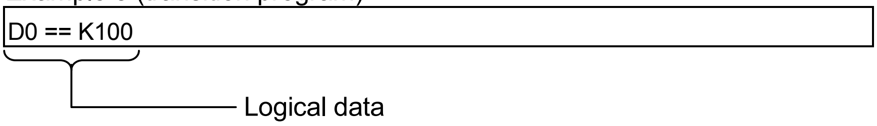
```
SET M0 = X10
```

Example 2



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

Example 3 (transition program)



```
D0 == K100
```

5.2 Device Descriptions

Word and bit device descriptions are shown below.

(1) Word device descriptions

	Device descriptions			Device No. (n) specifying ranges
	16-bit integer type	32-bit integer type ("n" is even No.)	64-bit floating-point type ("n" is even No.)	
Data register	Dn	DnL	DnF	0 to 8191
Link register	Wn	WnL	Wn:F	0 to 1FFF
Special register	Dn	DnL	DnF	9000 to 9255
Motion device	#n	#nL	#nF	0 to 8191 (Motion SFC dedicated devices : 8000 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) For the 32-bit integer type and 64-bit floating-point type, specify the device number with an even number. (It cannot be set as an odd number).

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

(2) Bit device descriptions

	Device description	Device No. (n) specifyied ranges
Input relay	Xn/PXn	0 to 1FFF
Output relay	Yn/PYn	0 to 1FFF
Internal relay	Mn	0 to 8191
Latch relay	Ln	0 to 8191
Link relay	Bn	0 to 1FFF
Annunciator	Fn	0 to 2047
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 specification of device No.

In the above word/bit device descriptions, device No. (n) can be specified indirectly.

(a) Indirect specification of device No. (n) using word device

- The word device which the device No. was specified indirectly cannot be used.
- You can use the 16-bit and 32-bit integer type word devices for indirect specification.
  The 64-bit floating-point type cannot be used.

(Description examples)

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

(b) Indirect specification of device No. (n) using word device using operation expression

- Device No. 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: －

- The word device which the device No. is specified indirectly cannot be used.
- 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 No. 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 shown 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.

F/FS	G
○	○

5.4 Binary Operations

5.4.1 Substitution : =

Format	(D)=(S)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—
(D)	—	○	○	○	—	—	—	—	—	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Word device/constant/calculation expression to be substituted	Data type of (D)
(D)	Word device which will store the operation result	

[Functions]

(1) The data value specified with (S) is substituted to the specified word device at (D).

(2) 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 substituted.
(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.)

[Errors]

(1) An operation error will occur if:
- The data at (S) is outside the data type range of (D); or
- (D) or (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

(1) Program which substitutes the D0 value to #0

```
#0 = D0
```

#0 123 ← D0 123

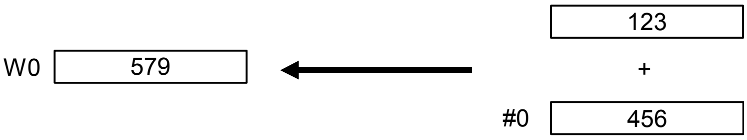
(2) Program which substitutes K123456.789 to D0L

```
D0L = K123456.789
```

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

(3) Program which substitutes the result of adding K123 and #0 to W0

```
W0 = K123 + #0
```



F/FS	G
○	○

5.4.2 Addition : +

Format	(S1)+(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Augend data	Data type of (S1) or (S2) which is greater
(S2)	Addend data	

[Functions]

(1) The data specified with (S2) is added to the data specified with (S1).

(2) 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.

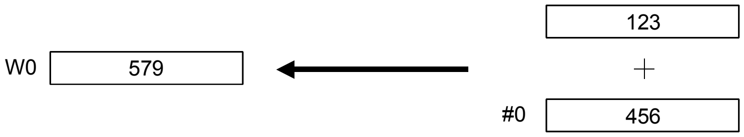
[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

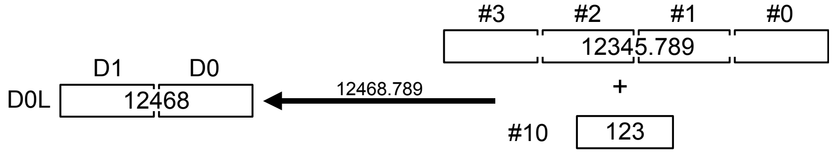
(1) Program which substitutes the result of adding K123 and #0 to W0

```
W0 = K123 + #0
```



(2) Program which substitutes the result of adding #0F and #10 to D0L

```
D0L = #0F + #10
```



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

F/FS	G
○	○

5.4.3 Subtraction : −

Format	(S1)−(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Minuend data	Data type of (S1) or (S2) which is greater
(S2)	Subtracted data	

[Functions]

(1) The data specified with (S2) is subtracted from the data specified with (S1).

(2) 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.

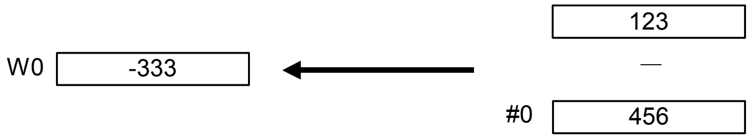
[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

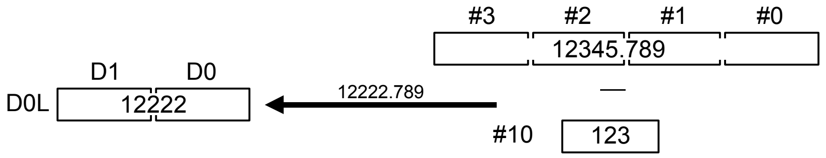
(1) Program which substitutes the result of subtracting #0 from K123 to W0

```
W0 = K123−#0
```



(2) Program which substitutes the result of subtracting #10 from #0F to D0L

```
D0L = #0F−#10
```



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

F/FS	G
○	○

5.4.4 Multiplication : *

Format	(S1)*(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Multiplicand data	Data type of (S1) or (S2) which is greater
(S2)	Multiplier data	

[Functions]

(1) The data specified with (S1) is multiplied by the data specified with (S2).

(2) 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.

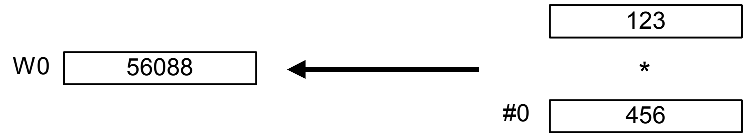
[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

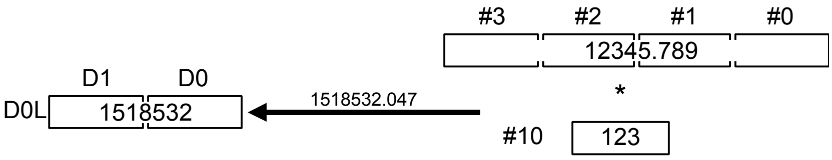
(1) Program which substitutes the result of multiplying K123 by #0 to W0

```
W0 = K123*#0
```



(2) Program which substitutes the result of multiplying #0F by #10 to D0L

```
D0L = #0F*#10
```



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

F/FS	G
○	○

5.4.5 Division : /

Format	(S1)/(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Dividend data	Data type of (S1) or (S2) which is greater
(S2)	Divisor data	

[Functions]

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

(2) 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.

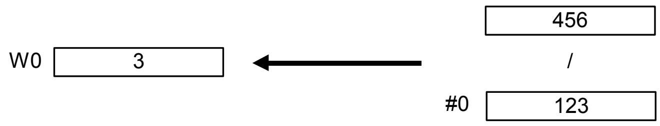
[Errors]

(1) An operation error will occur if:
- (S2) is 0; or
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

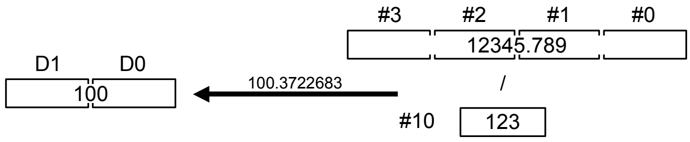
(1) Program which divides K456 by #0 and substitutes a quotient to W0

```
W0 = K456 / #0
```



(2) Program which divides #0F by #10 and substitutes a quotient to D0L

```
D0L = #0F / #10
```



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

F/FS	G
○	○

5.4.6 Remainder : %

Format	(S1)%(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	—	○	○	○	—	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Dividend data	Data type (integer type) of (S1) or (S2) which is greater (Integer type)
(S2)	Divisor data	

[Functions]

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

(2) 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.

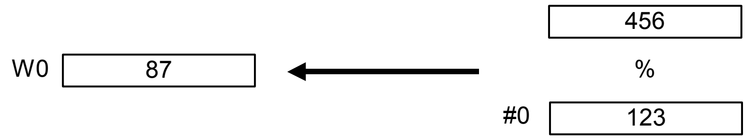
[Errors]

(1) An operation error will occur if:
- (S2) is 0; or
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

(1) Program which divides K456 by #0 and substitutes a remainder to W0

```
W0 = K456 % #0
```



F/FS	G
○	○

5.5 Bit Operations

5.5.1 Bit inversion (Complement) : ˜

Format	˜(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	—	○	○	○	—	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data whose bits will be inverted	Data type of (S) (Integer type)

[Functions]

(1) The bit inverted value of the data specified with (S) is found.

[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

(1) Program which finds the bit inverted value of #0 and substitutes the value to D0

```
D0 = ˜#0
```

F/FS	G
○	○

5.5.2 Bit logical AND : &

Format	(S1)&(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	—	○	○	○	—	○	—	—

○ : Usable

[Setting data]

Setting data	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)		

[Functions]

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

(2) 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.

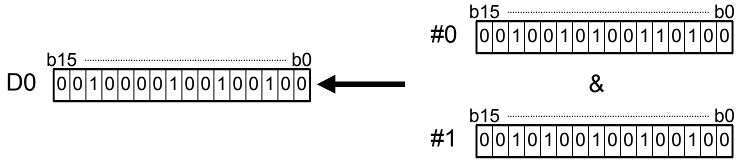
[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

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



F/FS	G
○	○

5.5.3 Bit logical OR : |

Format	(S1) I (S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	—	○	○	○	—	○	—	—

○ : Usable

[Setting data]

Setting data	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)		

[Functions]

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

(2) 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.

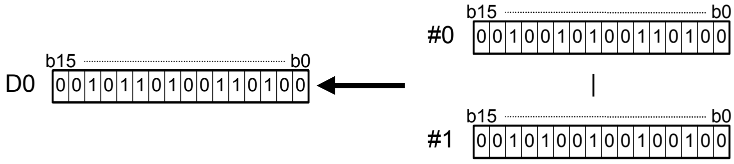
[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
D0 = #0 I #1
```



F/FS	G
○	○

5.5.4 Bit exclusive logical OR : ^

Format	(S1)^(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	—	○	○	○	—	○	—	—

○ : Usable

[Setting data]

Setting data	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)		

[Functions]

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

(2) 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.

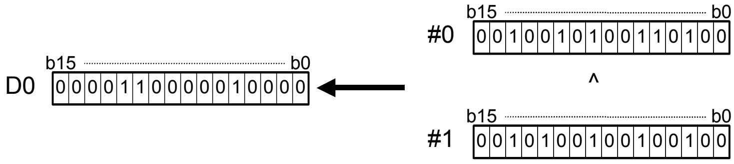
[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

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



F/FS	G
○	○

5.5.5 Bit right shift : >>

Format	(S1) >> (S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	—	○	○	○	—	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Data to be right-shifted	Data type of (S1) (Integer type)
(S2)	Number of right shifts	

[Functions]

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

(2) 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.

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

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

[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

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

F/FS	G
○	○

5.5.6 Bit left shift : <<

Format	(S1) << (S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	—	○	○	○	—	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Data to be left-shifted	Data type of (S1) (Integer type)
(S2)	Number of left shifts	

[Functions]

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

(2) 0 enters the least significant bit of the left shift result.

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

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

[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

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

F/FS	G
○	○

5.5.7 Sign inversion (Complement of 2) : －

Format	－(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data whose sign will be inverted	Data type of (S)

[Functions]

(1) The sign-inverted value of the data specified with (S) is found.

[Errors]

(1) An operation error will occur if:

- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

(1) Program which substitutes the sign-inverted value of #0 to D0

```
D0 =－#0
```

F/FS	G
○	○

5.6 Standard Functions

5.6.1 Sine : SIN

Format	SIN(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Angle data on which SIN (sine) operation will be performed	Floating-point type

[Functions]

(1) SIN (sine) operation is performed on the data specified with (S).

(2) The data specified with (S) is in an angle (degree) unit.

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

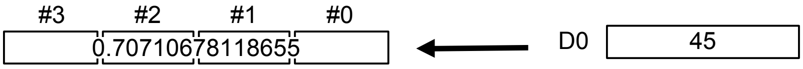
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0F = SIN(D0)
```



F/FS	G
○	○

5.6.2 Cosine : COS

Format	COS(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Angle data on which COS (cosine) operation will be performed	Floating-point type

[Functions]

(1) COS (cosine) operation is performed on the data specified with (S).

(2) The data specified with (S) is in an angle (degree) unit.

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

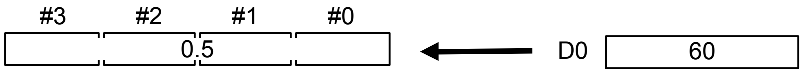
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0F = COS(D0)
```



F/FS	G
○	○

5.6.3 Tangent : TAN

Format	TAN(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Angle data on which TAN (tangent) operation will be performed	Floating-point type

[Functions]

(1) TAN (tangent) operation is performed on the data specified with (S).

(2) The data specified with (S) is in an angle (degree) unit.

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

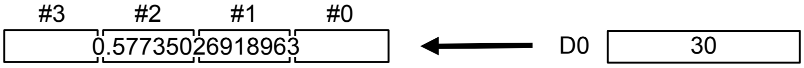
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range; or
- (S) is 90+(180*n). ("n" is an integer)

[Program examples]

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

```
#0F = TAN(D0)
```



F/FS	G
○	○

5.6.4 Arcsine : ASIN

Format	ASIN(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	SIN value data on which SIN $^{-1}$ (arcsine) operation will be performed	Floating-point type

[Functions]

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

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

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

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

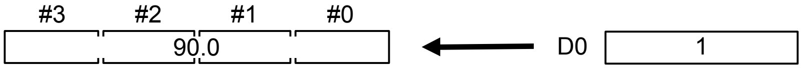
[Errors]

(1) An operation error will occur if:
- (S) is outside the range -1.0 to 1.0; or
- (S) is an indirectly specified device and its device number is outside the range.

[Program examples]

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

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



F/FS	G
○	○

5.6.5 Arccosine : ACOS

Format	ACOS(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

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

[Functions]

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

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

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

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

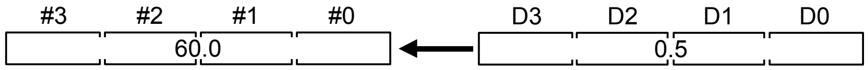
[Errors]

(1) An operation error will occur if:
- (S) is outside the range -1.0 to 1.0; or
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0F = ACOS(D0F)
```



F/FS	G
○	○

5.6.6 Arctangent : ATAN

Format	ATAN(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	TAN value data on which TAN $^{-1}$ (arctangent) operation will be performed	Floating-point type

[Functions]

(1) TAN $^{-1}$ (arctangent) operation is performed on the TAN value data specified with (S) to find an angle.

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

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

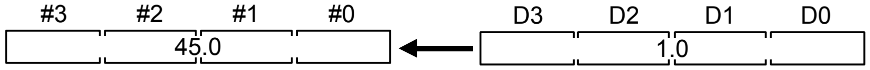
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0F = ATAN(D0F)
```



F/FS	G
○	○

5.6.7 Square root : SQRT

Format	SQRT(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data on which square root operation will be performed	Floating-point type

[Functions]

(1) The square root of the data specified with (S) is found.

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

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

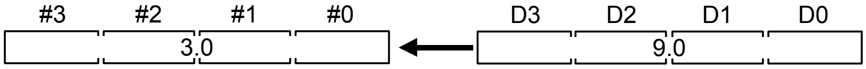
[Errors]

(1) An operation error will occur if:
- (S) is a negative number; or
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0F = SQRT(D0F)
```



F/FS	G
○	○

5.6.8 Natural logarithm : LN

Format	LN(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data on which natural logarithm operation will be performed	Floating-point type

[Functions]

(1) The base e natural logarithm of the data specified with (S) is found.

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

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

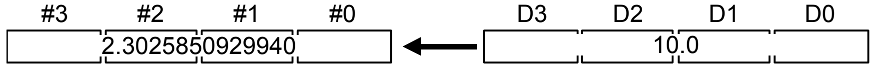
[Errors]

(1) An operation error will occur if:
- (S) is 0 or a negative number; or
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0F = LN(D0F)
```



F/FS	G
○	○

5.6.9 Exponential operation : EXP

Format	EXP(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data on which exponential operation will be performed	Floating-point type

[Functions]

(1) Exponential operation is performed on the base e data specified with (S).

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

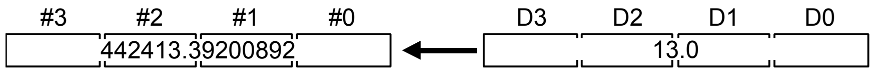
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0F = EXP(D0F)
```



F/FS	G
○	○

5.6.10 Absolute value : ABS

Format	ABS(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data on which absolute value conversion will be performed	Data type of (S)

[Functions]

(1) The absolute value of the data specified with (S) is found.

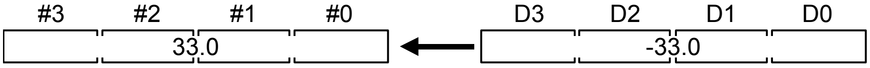
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0F = ABS(D0F)
```



F/FS	G
○	○

5.6.11 Round-off : RND

Format	RND(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data whose fractional portion will be rounded off	Data type of (S)

[Functions]

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

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

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

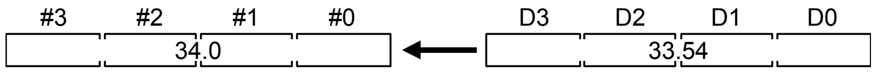
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

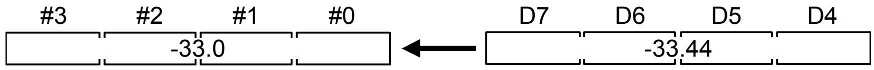
(1) Program which finds the rounded-off fractional portion value of D0F and substitutes the result to #0F

```
#0F = RND(D0F)
```



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

```
#0F = RND(D4F)
```



F/FS	G
○	○

5.6.12 Round-down : FIX

Format	FIX(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data whose fractional portion will be rounded down	Data type of (S)

[Functions]

(1) The largest integer not greater than the data specified with (S) is found.

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

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

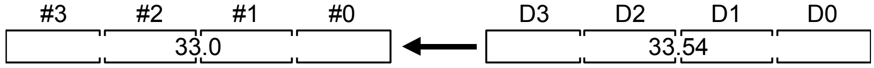
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

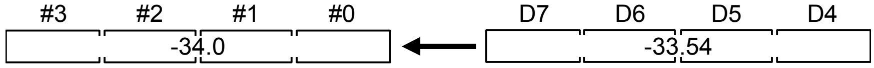
(1) Program which finds the rounded-down fractional portion value of D0F and substitutes the result to #0F

```
#0F = FIX(D0F)
```



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

```
#0F = FIX(D4F)
```



F/FS	G
○	○

5.6.13 Round-up : FUP

Format	FUP(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data whose fractional portion will be rounded up	Data type of (S)

[Functions]

(1) The smallest integer not less than the data specified with (S) is found.

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

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

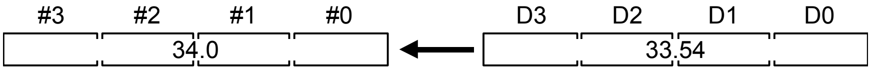
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

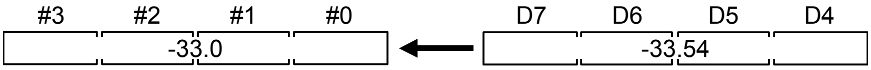
(1) Program which finds the rounded-up fractional portion value of D0F and substitutes the result to #0F

```
#0F = FUP(D0F)
```



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

```
#0F = FUP(D4F)
```



F/FS	G
○	○

5.6.14 BCD → BIN conversion : BIN

Format	BIN(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	—	○	○	○	—	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	BCD data which will be converted into BIN data	Data type of (S) (Integer type)

[Functions]

(1) The BCD data specified with (S) is converted into BIN data.

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

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

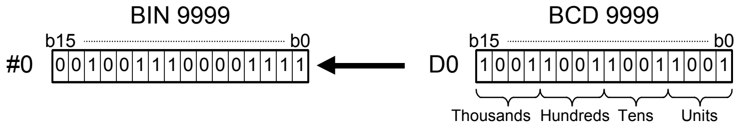
[Errors]

(1) An operation error will occur if:
- A value other than 0 to 9 is in any digit of (S); or
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

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



F/FS	G
○	○

5.6.15 BIN → BCD conversion : BCD

Format	BCD(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	—	○	○	○	—	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	BIN data which will be converted into BCD data	Data type of (S) (Integer type)

[Functions]

(1) The BIN data specified with (S) is converted into BCD data.

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

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

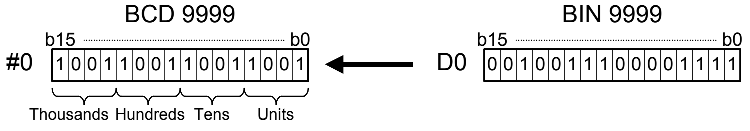
[Errors]

(1) An operation error will occur if:
- The data is other than 0 to 9999 when (S) is a 16-bit integer type;
- The data is other than 0 to 99999999 when (S) is a 32-bit integer type; or
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

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



F/FS	G
○	○

5.7 Type Conversions

5.7.1 Signed 16-bit integer value conversion : SHORT

Format	SHORT(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data which will be converted into signed 16-bit integer value	16-bit integer type

[Functions]

(1) The data specified with (S) is converted into a signed 16-bit integer value.

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

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

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

[Errors]

(1) An operation error will occur if:
- The (S) data is outside the range -32768 to 32767; or
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0 = SHORT(D0L)
```

F/FS	G
○	○

5.7.2 Unsigned 16-bit integer value conversion : USHORT

Format	USHORT(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data which will be converted into unsigned 16-bit integer value	16-bit integer type

[Functions]

(1) The data specified with (S) is converted into an unsigned 16-bit integer value.

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

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

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

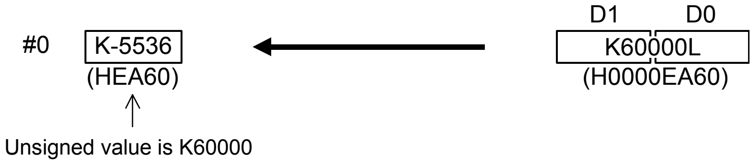
[Errors]

(1) An operation error will occur if:
- The (S) data is outside the range 0 to 65535; or
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

(1) Program which converts the data of D0L into an unsigned 16-bit integer value and substitutes the result to #0

```
#0 = USHORT(D0L)
```



F/FS	G
○	○

5.7.3 Signed 32-bit integer value conversion : LONG

Format	LONG(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data which will be converted into signed 32-bit integer value	32-bit integer type

[Functions]

(1) The data specified with (S) is converted into a signed 32-bit integer value.

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

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

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

[Errors]

(1) An operation error will occur if:
- The (S) data is outside the range -2147483648 to 2147483647; or
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0L = LONG(D0)
```

F/FS	G
○	○

5.7.4 Unsigned 32-bit integer value conversion : ULONG

Format	ULONG(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data which will be converted into unsigned 32-bit integer value	32-bit integer type

[Functions]

(1) The data specified with (S) is converted into an unsigned 32-bit integer value.

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

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

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

[Errors]

(1) An operation error will occur if:
- The (S) data is outside the range 0 to 4294967295; or
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

(1) Program which converts the data of D0 into an unsigned 32-bit integer value and substitutes the result to #0L

```
#0L = ULONG(D0)
```

F/FS	G
○	○

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

Format	FLOAT(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data which will be converted into signed 64-bit floating-point value	64-bit floating-point type

[Functions]

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

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

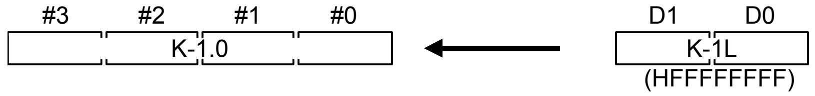
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0F = FLOAT(D0L)
```



F/FS	G
○	○

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

Format	UFLOAT(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data which will be converted into unsigned 64-bit floating-point value	64-bit floating-point type

[Functions]

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

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

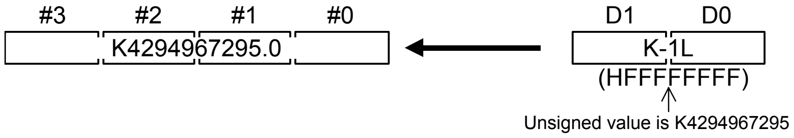
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0F = UFLOAT(D0L)
```



F/FS	G
○	○

5.8 Bit Device Statuses

5.8.1 ON (Normally open contact) : (None)

Format	(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○	—	—	—	—	—	—	—	—	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Bit device used in bit conditional expression	Logical type (true/false)

[Functions]

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

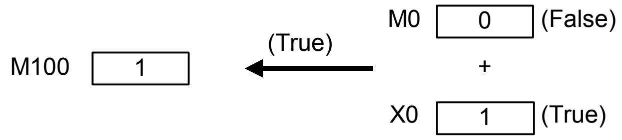
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
SET M100 = M0 + X0
```



F/FS	G
○	○

5.8.2 OFF (Normally closed contact) : !

Format	!(S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○	—	—	—	—	—	—	—	—	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Bit device used in bit conditional expression	Logical type (true/false)

[Functions]

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

[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
RST M100 = !M0
```

M100 0 ← !M0 0 (True)

F/FS	G
○	○

5.9 Bit Device Controls

5.9.1 Device set : SET

Format	SET(D)=(S)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○	—	—	—	—	—	—	—	—	○	○

○ : Usable

(Note-1) : PX is write-disabled and cannot be used at (D).
(Note-2) : M2001 to M2032 cannot be used at (D).

[Setting data]

Setting data	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	

[Functions]

(1) If the data specified with (S) is true, the bit data specified with (D) is set.

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

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

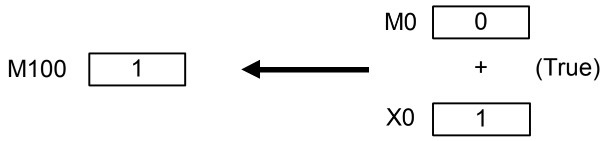
[Errors]

(1) An operation error will occur if:
- (D) or (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

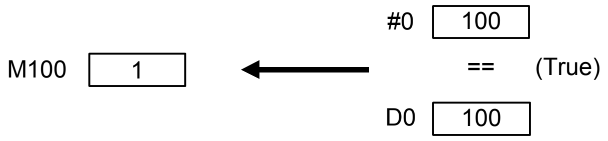
(1) Program which sets M100 when either of M0 and X0 is 1

```
SET M100 = M0 + X0
```



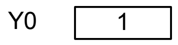
(2) Program which sets M100 when #0 is equal to D0

```
SET M100 = #0 = = D0
```



(3) Program which sets Y0 unconditionally

```
SET Y0
```



F/FS	G
○	○

5.9.2 Device reset : RST

Format	RST(D)=(S)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○	—	—	—	—	—	—	—	—	○	○

○ : Usable

(Note-1) : PX is write-disabled and cannot be used at (D).
(Note-2) : M2001 to M2032 cannot be used at (D).

[Setting data]

Setting data	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	

[Functions]

(1) If the data specified with (S) is true, the bit data specified with (D) is reset.

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

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

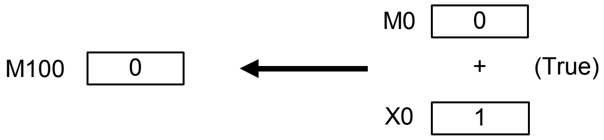
[Errors]

(1) An operation error will occur if:
- (D) or (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

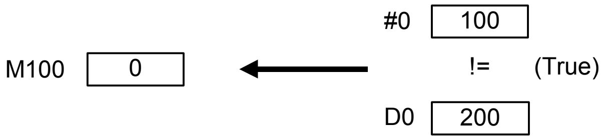
(1) Program which resets M100 when either of M0 and X0 is 1

```
RST M100 = M0 + X0
```



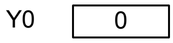
(2) Program which resets M100 when #0 is equal to D0

```
RST M100 = #0 != D0
```



(3) Program which resets Y0 unconditionally

```
RST Y0
```



F/FS	G
○	○

5.9.3 Device output : DOUT

Format	DOUT(D), (S)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	—	○	○	○	—	○	—	—

○ : Usable

(Note-1) : PX and special relay cannot be used at (D).
(Note-2) : Range including M2000 to M2127 cannot be used at (D).

[Setting data]

Setting data	Description	Data type of result
(D)	Output destination bit data	Batch bit
(S)	Output source data	

[Functions]

(1) The data specified with (S) is output to the bit data specified with (D).

(2) Specify a multiple of 16 as the device No. of the bit data specified with (D).

(3) 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 with (D).

(4) 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 with (D).

[Errors]

(1) An operation error will occur if:
- (D) or (S) is an indirectly specified device and its device No. is outside the range.
- (D) is an indirectly specified device and its device No. is not a multiple of 16.

[Program examples]

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

```
DOUT Y0, D0
```

YF ………… Y0			b15 ………… b0
0 0 1 0 0 1 1 1 0 0 0 0 1 1 1 1	←	D0	0 0 1 0 0 1 1 1 0 0 0 0 1 1 1 1

F/FS	G
○	○

5.9.4 Device input : DIN

Format	DIN(D), (S)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○	—	—	—	—	—	—	—	—	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(D)	Input destination data	Data type of (D) (Integer type)
(S)	Input source bit data	

[Functions]

(1) The bit data specified with (S) is input to the data specified with (D).

(2) Specify a multiple of 16 as the device No. of the bit data specified with (S).

(3) 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 with (S).

(4) 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 with (S).

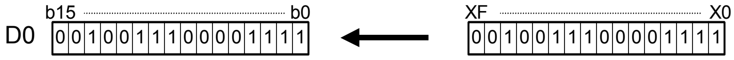
[Errors]

(1) An operation error will occur if:
- (D) or (S) is an indirectly specified device and its device No. is outside the range.
- (S) is an indirectly specified device and its device No. is not a multiple of 16.

[Program examples]

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

```
DIN D0, X0
```



	b15															b0
D0	0	0	1	0	0	1	1	1	0	0	0	0	1	1	1	1

XF															X0
0	0	1	0	0	1	1	1	0	0	0	0	1	1	1	1

F/FS	G
○	○

5.9.5 Bit device output : OUT

Format	OUT(D)=(S)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○	—	—	—	—	—	—	—	—	○	○

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(D)	Bit device for device output	Bit logical type (true/false)
(S)	Condition data which determines device output	

[Functions]

(1) If the data specified with (S) is true, the bit data specified with (D) is set, and if the data specified with (S) is false, the bit data specified with (D) is reset.

(2) When this instruction is set as a transition condition in the last block of a transient program, whether the data specified with (S) is true or false is returned as logical type data.

(3) In this case, (S) cannot be omitted.

[Errors]

(1) An operation error will occur if:
- (D) or (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

(1) Program which sets M100 when M0 is ON (1) and program which resets M100 when M0 is OFF (0)

```
OUT M100 = M0
```

(2) Program which sets M100 when M0 and M1 are both on and resets M100 except it

```
OUT M100 = M0 * M1
```

(3) Program which sets M100 when D0 is equal to D2000 and resets M100 when D is not equal to D2000

```
OUT M100 = (D0 == D2000)
```

F/FS	G
○	○

5.10 Logical Operations

5.10.1 Logical acknowledgement : (None)

Format	(S)

Number of basic steps	—

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○	—	—	—	—	—	—	—	—	○	○

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data which will be logically acknowledged	Logical type (true/false)

[Functions]

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

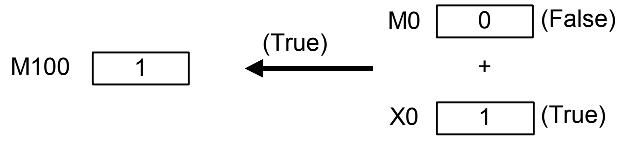
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
SET M100 = M0 + X0
```



F/FS	G
○	○

5.10.2 Logical negation : !

Format	! (S)

Number of basic steps	2

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○	—	—	—	—	—	—	—	—	○	○

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Data which will be logically negated	Logical type (true/false)

[Functions]

(1) The data specified with (S) is logically negated.

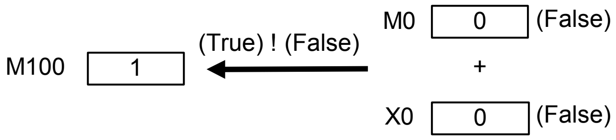
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
SET M100 = ! (M0 + X0)
```



F/FS	G
○	○

5.10.3 Logical AND : ∗

Format	(S1)∗(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○	—	—	—	—	—	—	—	—	○	○

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Data which will be ANDed	Logical type (true/false)
(S2)		

[Functions]

(1) The data specified with (S1) and the data specified with (S2) are ANDed.

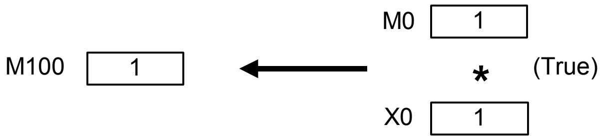
[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
SET M100 = M0∗X0
```



F/FS	G
○	○

5.10.4 Logical OR : +

Format	(S1)+(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○	—	—	—	—	—	—	—	—	○	○

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Data which will be ORed	Logical type (true/false)
(S2)		

[Functions]

(1) The data specified with (S1) and the data specified with (S2) are ORed.

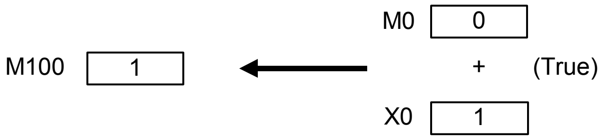
[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
SET M100 = M0 + X0
```



F/FS	G
○	○

5.11 Comparison Operations

5.11.1 Equal to : ==

Format	(S1)==(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Data which will be compared	Logical type (true/false)
(S2)		

[Functions]

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

(2) 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.

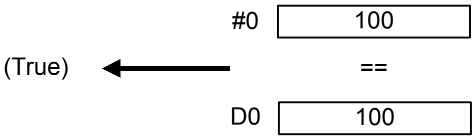
[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0 = = D0
```



F/FS	G
○	○

5.11.2 Not equal to : !=

Format	(S1)!=(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Data which will be compared	Logical type (true/false)
(S2)		

[Functions]

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

(2) 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.

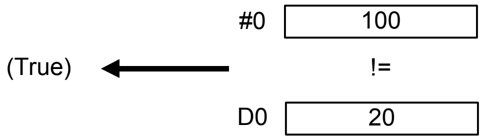
[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0 != D0
```



F/FS	G
○	○

5.11.3 Less than : <

Format	(S1)<(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Data which will be compared	Logical type (true/false)
(S2)		

[Functions]

(1) The result is true if the data specified with (S1) is less than the data specified with (S2).

(2) 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.

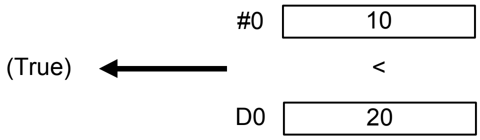
[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0 < D0
```



F/FS	G
○	○

5.11.4 Less than or equal to: <=

Format	(S1)<=(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Data which will be compared	Logical type (true/false)
(S2)		

[Functions]

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

(2) 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.

[Errors]

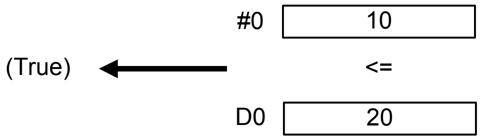
(1) An operation error will occur if:

- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0 <= D0
```



F/FS	G
○	○

5.11.5 More than : >

Format	(S1)>(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Data which will be compared	Logical type (true/false)
(S2)		

[Functions]

(1) The result is true if the data specified with (S1) is greater than the data specified with (S2).

(2) 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.

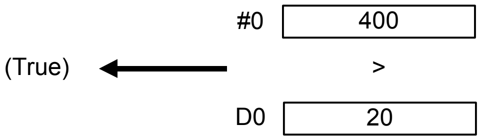
[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0 > D0
```



F/FS	G
○	○

5.11.6 More than or equal to: >=

Format	(S1)>=(S2)

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	○	○	○	○	○	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Data which will be compared	Logical type (true/false)
(S2)		

[Functions]

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

(2) 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.

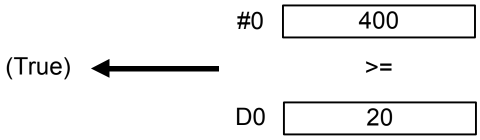
[Errors]

(1) An operation error will occur if:
- (S1) or (S2) is an indirectly specified device and its device No. is outside the range.

[Program examples]

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

```
#0 >= D0
```



F/FS	G
○	○

5.12 Motion-Dedicated Functions (CHGV, CHGT)

5.12.1 Speed change request : CHGV

Format	CHGV((S1), (S2))

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	—	—	○	○	—	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Axis No. to which speed change request will be given	—
(S2)	Specified speed	

[Functions]

(1) A speed change is made in the following procedure.
  (a) The speed changing flag (M2061 to M2092) correspond to the axis specified with (S1) is turned ON.
  (b) The speed of the axis specified with (S1) is changed to the speed specified with (S2).
  (c) The speed changing flag is turned OFF.

(2) The axis No. that may be set at (S1) is within the following range.

Q172HCPU	Q173HCPU
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	Speed change is made so that the combined speed becomes the speed specified with (S2).
Longest axis designation	Speed change is made so that the longest axis speed becomes the speed specified with (S2).
Reference axis speed designation	Speed change is made so that the reference axis speed becomes the speed specified with (S2).

(3) 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

(4) The specified speed that may be set at (S2) is within the following range.

(a) Real mode

	mm		inch		degree		PLS	
	Setting range	Unit	Setting range	Unit	Setting range	Unit	Setting range	Unit
Speed change request	0 to 600000000	× 10^{-2} mm/min	0 to 600000000	× 10^{-3} inch/min	0 to 2147483647	× 10^{-3} degree/min (Note-1)	0 to 10000000	PLS/s
Return request	-1 to -600000000	× 10^{-2} mm/min	-1 to -600000000	× 10^{-3} inch/min	-1 to -2147483647	× 10^{-3} degree/min (Note-1)	-1 to -10000000	PLS/s

(Note-1) : When the "speed control 10 × multiplier setting for degree axis" is set to "valid" in the fixed parameter, the unit is " × 10^{-2} [degree/min] ".

(b) Virtual mode

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

(5) The speed changed by CHGV instruction is effective only on the servo program during starting.

(6) The speed change does not executed for the axis specified with (S1) during deceleration stop.

(7) By specifying a negative speed and making a speed change request during the start, allows the axis to start deceleration at that point and return in the opposite direction upon completion of deceleration.
The following operations by the servo instruction are shown 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 instruction 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] (Note) will occur and the axis will be controlled at the speed limit value.
Position follow-up control	PFSTART	
Speed control with fixed position stop	PVF PVR	
Speed switching control	VSTART	
JOG operation		
High-speed oscillation	OSC	A speed change cannot be made. Minor error [310] (Note) will occur.
Home position return	ZERO	A speed change cannot be made. Minor error [301] (Note) will occur.

(Note) : Minor error [301] : A speed change was made during home position return.
Minor error [305] : The setting speed is outside the range of 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 is executed with the control mode during the start as indicated in the above table.

(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 (n : Axis No., m : Axis No. –1)

- Start accept (M2000+n) ON (unchanged from before execution of CHGV instruction)
- Positioning start completion (M2400+20m) ON (unchanged from before execution of CHGV instruction)
- Positioning completion (M2401+20m) OFF
- In-position (M2402+20m) ON
- Command in-position (M2403+20m) OFF
- Speed change "0" accepting flag (M2240+m) ON

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

(d) While the axis is reversion in the speed control mode
1) Make a speed change to a positive speed to change the travel 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.

[Errors]

(1) An operation error will occur and a speed change will not be made if:
- The specified axis No. of (S1) is outside the range.
- (S2) is an indirectly specified device and its device No. is outside the range.

(2) A minor error will occur and a speed change will not be made if:
- The axis specified with (S1) is home position return. (Minor error: 301).

POINT

If the speed change is executed for the axis specified with (S1) during deceleration, the speed change is ignored. An error will not occur in this case

(3) A minor error will occur and the axis to be controlled at the speed limit value if:
- The absolute value of the speed specified with (S2) is greater than the speed limit value. (Minor error: 305)

POINT

If the absolute value of a negative new speed is higher than the speed specified with the servo program during constant-speed control, 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.

[Program examples]

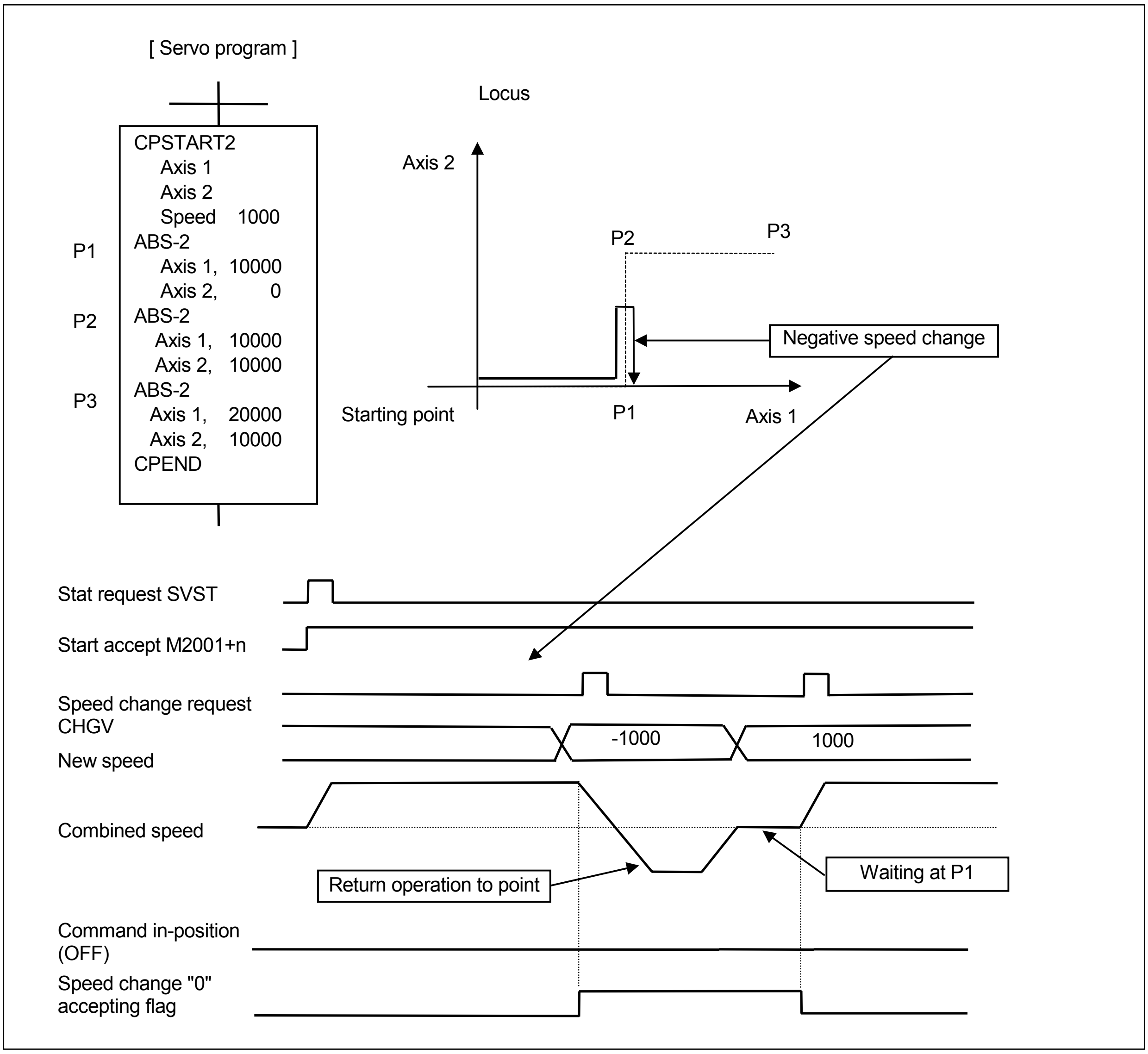
(1) Program which changes the positioning speed of axis 2

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

(2) 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.

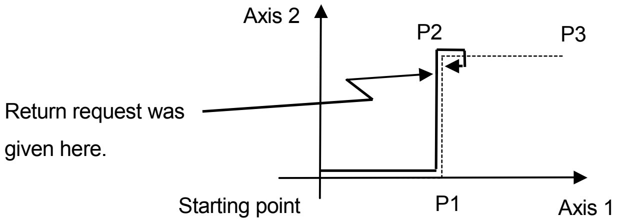
POINT

(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 signal 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) 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.



F/FS	G
○	○

5.12.2 Torque limit value change request : CHGT

Format	CHGT((S1), (S2))

Number of basic steps	4

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	—	—	○	○	—	○	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S1)	Axis No. to which torque limit value change request will be given	—
(S2)	Specified torque limit value	

[Functions]

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

(2) 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.

(3) The axis No. that may be set at (S1) is within the following range.

Q172HCPU	Q173HCPU
1 to 8	1 to 32

(4) The torque limit value that may be set at (S2) is within the range 1 to 1000[%].

(5) 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

(a) 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 switching control
  - Torque limit value at the point of switching to position control in speed/position changing control
  - Torque limit value in speed control

(b) The CHGT instruction accepts a torque limit value which is higher than the torque limit value set in the servo program or parameter block.

(6) The torque limit value changed by CHGT instruction is effective only during power supply is on.

[Errors]

(1) An operation error will occur and a torque limit value change will not be made if:
- The specified axis No. at (S1) is outside the range; or
- (S2) is an indirectly specified device and its device No. is outside the range.

(2) A minor error will occur and a torque limit value change will not be made if:
- The torque limit value specified with (S2) is outside the range 1 to 1000[%] (Minor error: 311); or
- The CHGT instruction is executed for any axis that has not yet been started (Minor error: 312).

[Program examples]

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

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

POINT

(1) CHGT instruction is invalid (ignored) during the virtual mode. 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 mechanical system 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.

F/FS	G
○	○

5.13 Other Instructions

5.13.1 Event task enable : EI

Format	EI

Number of basic steps	1

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)			
—	—	—	—	—	—	—	—	—	—	—	—

○ : Usable

[Setting data]

There are no setting data.

[Functions]

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

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

[Errors]

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

[Program examples]

(1) Enables the execution of an event task.

```
EI
```

F/FS	G
○	○

5.13.2 Event task disable : DI

Format	DI

Number of basic steps	1

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)			
—	—	—	—	—	—	—	—	—	—	—	—

○ : Usable

[Setting data]

There are no setting data.

[Functions]

(1) The execution of an event task is disabled.

(2) 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.)

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

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

(5) The DI status is established at power-on or when a reset is made with the RESET/L.CLR switch.

[Errors]

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

[Program examples]

(1) Program which disables the execution of an event task.

```
DI
```

F/FS	G
○	○

5.13.3 No operation : NOP

Format	NOP

Number of basic steps	1

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)			
—	—	—	—	—	—	—	—	—	—	—	—

○ : Usable

[Setting data]

There are no setting data.

[Functions]

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

[Errors]

(1) There are no operation errors for no operation: NOP.

F/FS	G
○	○

5.13.4 Block transfer : BMOV

Format	BMOV(D), (S), (n)

Number of basic steps	6

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○ (Note-1)	○	—	—	—	—	○	—	—	—	—
(S)	○ (Note-1)	○	—	—	—	—	○	—	—	—	—
(n)	—	○	—	—	—	○	—	—	—	—	—

○ : Usable

(Note-1) : Refer to the Section "1.2.5" for the correspondence version of the Motion CPU and the software.

[Setting data]

Setting data	Description	Data type of result
(D)	Transfer destination device starting No.	—
(S)	Transfer source device starting No.	
(n)	Number of words to be transferred	

[Functions]

(1) The contents for n words from device specified with (S) are batch-transferred to the n words from device specified with (D).

(2) Data can be transferred if the devices of the transfer source and destination overlap.
Data are transferred from 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.

(3) Specifying Nn (cam No.) at (D) or (S) enables batch-transfer of cam data.
In the Motion controller, the cam data of same cam No. must already have been registered.
The number of transferred words specified with (n) should match the resolution of the specified cam No..

At cam data write

The cam data storage area is rewritten.

- 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 No..

At cam data read

The cam data storage area is rewritten.

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

(4) The word devices that may be set at (D), (S) and (n) are shown below.

Setting data	Word devices (Note-2)			Bit devices (Note-2), (Note-3)					Cam No. specification
	Dn	Wn	#n	Mn	Bn	Fn	Xn	Yn	Nn (Note-1)
(D)	○	○	○	○ (Note-5)	○	○	○ (Note-4)	○ (Note-4)	○
(S)	○	○	○	○	○	○	○ (Note-4)	○ (Note-4)	○
(n)	○	○	○	—	—	—	—	—	—

(Note-1) : "Nn" indicates the cam No..

(Note-2) : The device No. cannot be specified indirectly.

(Note-3) : Specify a multiple of 16 as the device number of bit data.

(Note-4) : PX/PY cannot be set.

(Note-5) : Special relays (M9000 to M9255) and dedicated devices (M2000 to M2399) cannot be set.
(Note: DOUT cannot output the PX, special relays (M2000 to M9255) and dedicted devices (M2000 to M2127).)

(5) The cam No. that may be set as "Nn" is within the following range.

Q173HPU/Q172HCPU
1 to 64
101 to 164
201 to 264
301 to 364

[Errors]

(1) An operation error will occur if:

- The cam data of cam No. specified with (D) or (S) are not yet registered to the Motion controller
- The resolution of cam No. specified with (D) or (S) differs from the number of transferred words specified with (n)
- (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
- PX/PY is set in (S) to (S)+(n-1)
- PX/PY is set in (D) to (D)+(n-1)

when (n) specified is a word device

(2) When conversion is made in program editing of the SW6RN-GSV□P, an error will occur if:

- (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
- PX/PY is set in (S) to (S) + (n-1)
- PX/PY is set in (D) to (D) + (n-1)

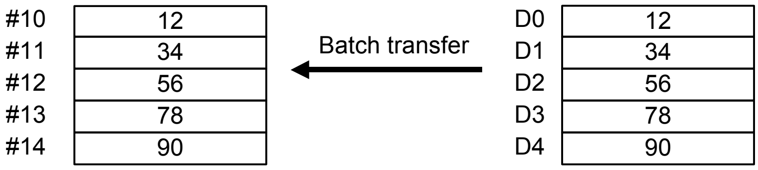
when (n) specified is a constant

- (S) is a bit device and the device number is not a multiple of 16
- (D) is a bit device and the device number is not a multiple of 16

[Program examples]

(1) Program which batch-transfers a contents for 5 words from D0 to all data for 5 words from #10

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

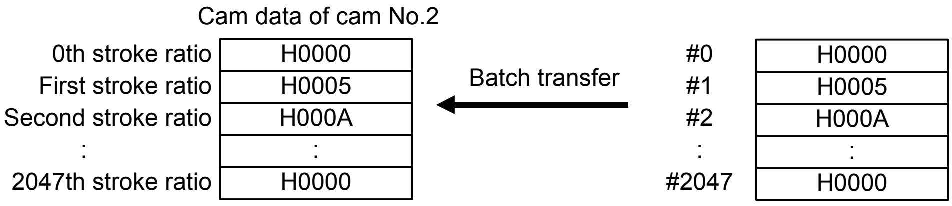


#10	12
#11	34
#12	56
#13	78
#14	90

D0	12
D1	34
D2	56
D3	78
D4	90

(2) Program which batch-transfers a contents for 2048 words from #0 to the data area of cam No.2 (resolution 2048)

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



Cam data of cam No.2

0th stroke ratio	H0000
First stroke ratio	H0005
Second stroke ratio	H000A
:	:
2047th stroke ratio	H0000

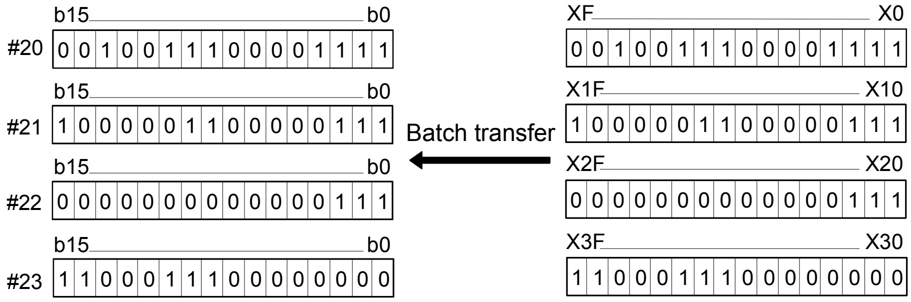
#0	H0000
#1	H0005
#2	H000A
:	:
#2047	H0000

POINT

Cam stroke ratio is set within 0 to 7FFFH.

(3) Program which batch-transfers a contents for 4 words from X0 to all data for 4 words from #20

```
BMOV #20, X0, K4
```



	b15															b0
#20	0	0	1	0	0	1	1	1	0	0	0	0	1	1	1	1
#21	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	1
#22	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
#23	1	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0

XF–X0	0	0	1	0	0	1	1	1	0	0	0	0	1	1	1	1
X1F–X10	1	0	0	0	0	0	1	1	0	0	0	0	0	1	1	1
X2F–X20	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
X3F–X30	1	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0

F/FS	G
○	○

5.13.5 Same data block transfer : FMOV

Format	FMOV(D), (S), (n)

Number of basic steps	6

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○ (Note-1)	○	—	—	—	—	○	—	—	—	—
(S)	○ (Note-1)	○	—	—	—	○	—	—	—	—	—
(n)	—	○	—	—	—	○	—	—	—	—	—

○ : Usable

(Note-1) : Refer to the Section "1.2.5" for the correspondence version of the Motion CPU and the software.

[Setting data]

Setting data	Description	Data type of result
(D)	Transfer destination device starting No.	—
(S)	Device No. which transfer data or data to be transferred are stored.	
(n)	Number of words to be transferred	

[Functions]

(1) The data specified with (S) or contents of device are transferred a part for (n)words of data to the device specified with (D).

(2) The devices that may be set at (D), (S) and (n) are shown below.

Setting data	Word devices (Note-1)			Bit devices (Note-1), (Note-2)				
	Dn	Wn	#n	Mn	Bn	Fn	Xn	Yn
(D)	○	○	○	○ (Note-4)	○	○	○ (Note-3)	○ (Note-3)
(S)	○	○	○	○	○	○	○ (Note-3)	○ (Note-3)
(n)	○	○	○	—	—	—	—	—

(Note-1) : The device No. cannot be specified indirectly.

(Note-2) : Specify a multiple of 16 as the device number of bit data.

(Note-3) : PX/PY cannot be set.

(Note-4) : Special relays (M9000 to M9255) and dedicated devices (M2000 to M2399) cannot be set.

[Errors]

(1) An operation error will occur if:

- (D) to (D)+(n-1) is outside the device range
- (n) is 0 or a negative number
- PX/PY is set in (D) to (D)+(n-1)

When (n) specified is a word device

(2) When conversion is made in program editing of the SW6RN-GSV☐P, an error will occur if:

- (D) to (D)+(n-1) is outside the device range
- (S) is outside the device range
- (n) is 0 or a negative number
- PX/PY is set in (S)
- PX/PY is set in (D) to (D) + (n-1)

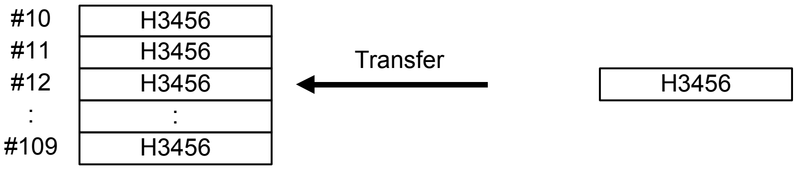
When (n) specified is a constant

- (S) is a bit device and the device number is not a multiple of 16
- (D) is a bit device and the device number is not a multiple of 16

[Program examples]

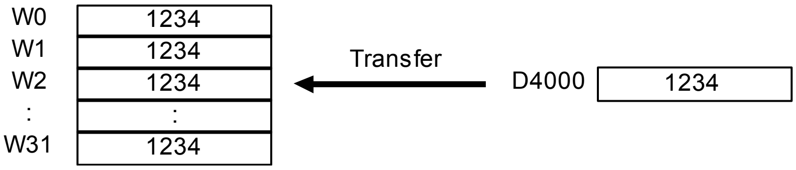
(1) Program which sets 3456H to all data for 100 words from #10

```
FMOV #10,H3456,K100
```



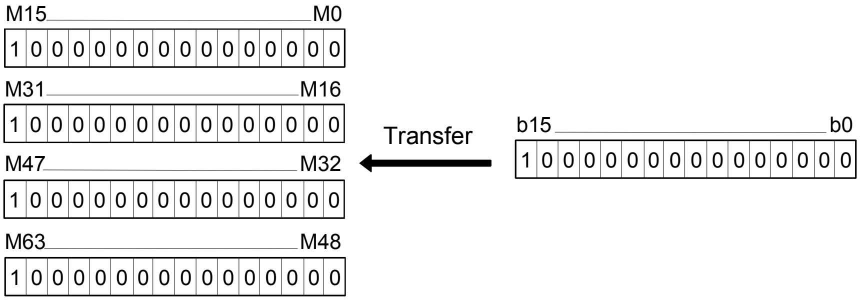
(2) Program which sets a content of D4000 to all data for 50 words from W0

```
FMOV W0,D4000,K50
```



(3) Program which sets 8000H to all data for 4 words from M0

```
FMOV M0, H8000, K4
```



F/FS	G
○	○

5.13.6 Write device data to shared CPU memory of the self CPU: MULTW

Format	MULTW(D), (S), (n), (D1)

Number of basic steps	8

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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-1)	○	—	—	—	—	—	—	—	—	—
(n)	—	○	—	—	—	○	—	—	—	—	—
(D1)	○	—	—	—	—	—	—	—	—	—	—

○ : Usable

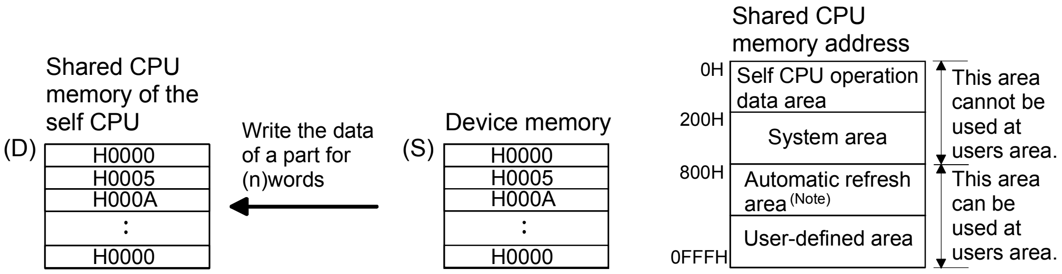
(Note-1) : Refer to the Section "1.2.5" for the correspondence version of the Motion CPU and the software.

[Setting data]

Setting data	Description	Data type of result
(D)	The shared CPU memory address of self CPU of the writing destination device. (800H to FFFH)	—
(S)	First device No. which writing data are stored.	
(n)	Number of words to be written (1 to 256)	
(D1)	Self CPU device is made to turn on by the writing completion.	

[Functions]

(1) A part for (n)words of data since the device specified with (S) of the self CPU module are written to since the shared CPU memory address specified with (D) of the self CPU module. After writing completion of the device data, the complete bit device specified with (D1) turns on.



(Note) : When automatic refresh is not set, it can be used as a user defined area. And, when automatic refresh is set up, since the automatic refresh transmitting range becomes a user defined area.

(2) Do resetting of the complete bit device by the user program.

(3) Another MULTW instruction cannot be processed until MULTW instruction is executed and a complete bit device is turned on. When MULTW instruction was executed again before MULTW instruction is executed and complete bit device is turned on, the MULTW instruction executed later becomes an error.

(4) The word devices that may be set at (D), (S) (n) and (D1) are shown below.

Setting data	Word devices (Note-1)			Bit devices (Note-1), (Note-2)				
	Dn	Wn	#n	Mn	Bn	Fn	Xn	Yn
(D)	○	○	○	—	—	—	—	—
(S)	○	○	○	○	○	○	○ (Note-3)	○ (Note-3)
(n)	○	○	○	—	—	—	—	—
(D1)	—	—	—	○	○	○	○ (Note-4)	○ (Note-4)

(Note-1) : The device No. cannot be specified indirectly.
(Note-2) : Specify a multiple of 16 as the device number of bit data.
(Note-3) : PX and PY cannot be set.
(Note-4) : PX can be set. PY cannot be set.

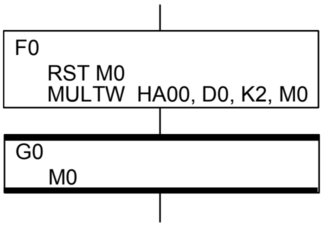
(5) Adjust an executive task, the number of transfer word referring to the operation processing time so that this instruction may not obstruct the execution of the motion operation because processing time becomes long in argument to the number of words (n) to be written.

[Errors]

(1) An operation error will occur if:
- Number of words (n) to be written is outside the range of 1 to 256.
- The shared CPU memory address (D) of self CPU of the writing destination device is outside the range (800H to FFFH) of the shared CPU memory address.
- The shared CPU memory address (D) of self CPU of the writing destination device + number of words (n) to be written is outside the range (800H to FFFH) of the shared CPU memory address.
- First device No. (S) which writing data are stored + number of words (n) to be written is outside the device range.
- MULTW instruction was executed again before MULTW instruction is executed and complete bit device is turned on.
- (D1) is a write-disabled device.
- (S) is a bit device and device number is not a multiple of 16.
- PX/PY is set in (S) to (S)+(n-1).

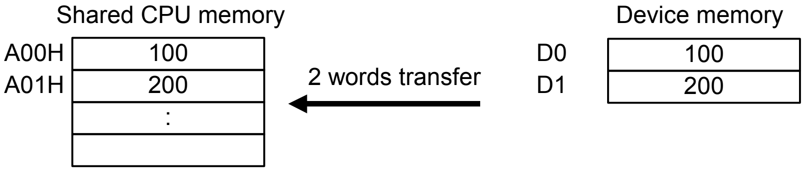
[Program examples]

(1) 2 words from D0 is written in the shared CPU memory to since A00H, and transits to next step after confirmation of writing completion.



```
F0
    RST M0
    MULTW  HA00, D0, K2, M0
```

```
G0
    M0
```



F/FS	G
○	○

5.13.7 Read device data from shared CPU memory of the other CPU: MULTR

Format	MULTR(D), (S1), (S2), (n)

Number of basic steps	7

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○ (Note-1)	○	—	—	—	—	—	—	—	—	—
(S1)	—	○	—	—	—	○	—	—	—	—	—
(S2)	—	○	—	—	—	○	—	—	—	—	—
(n)	—	○	—	—	—	○	—	—	—	—	—

○ : Usable

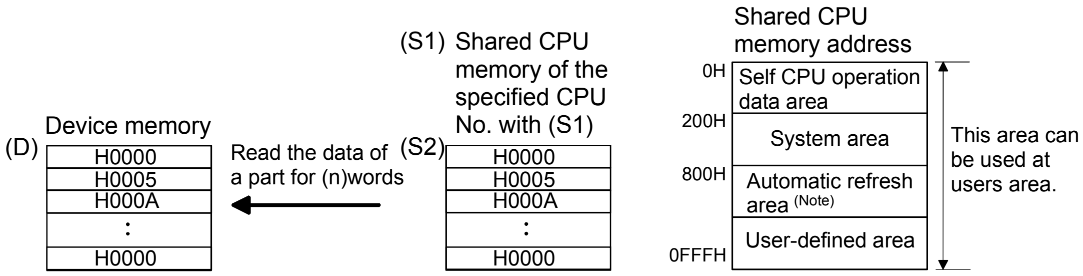
(Note-1) : Refer to the Section "1.2.5" for the correspondence version of the Motion CPU and the software.

[Setting data]

Setting data	Description	Data type of result
(D)	First device No. which stores the reading data.	—
(S1)	First I/O No. of the PLC CPU/Motion CPU which it will be read. (CPU No.1 : 3E0H, CPU No.2 : 3E1H, CPU No.3 : 3E2H, CPU No.4 : 3E3H)	
(S2)	The shared CPU memory first address of the data which it will be read. (000H to FFFH)	
(n)	Number of words to be read (1 to 256)	

[Functions]

(1) A part for (n)words of data of the other CPU specified with (S1) are read from the address specified with (S2) of the shared CPU memory, and are stored since the device specified with (S2).



(Note) : When automatic refresh is not set, it can be used as a user defined area. And, when automatic refresh is set up, since the automatic refresh transmitting range becomes a user defined area.

(2) The word devices that may be set at (D), (S), (n) and (D1) are shown below.

Setting data	Word devices (Note-1)			Bit devices (Note-1), (Note-2)				
	Dn	Wn	#n	Mn	Bn	Fn	Xn	Yn
(D)	○	○	○	○ (Note-3)	○	○	○ (Note-4)	○ (Note-4)
(S)	○	○	○	—	—	—	—	—
(n)	○	○	○	—	—	—	—	—
(D1)	○	○	○	—	—	—	—	—

(Note-1) : The device No. cannot be specified indirectly.
(Note-2) : Specify a multiple of 16 as the device number of bit data.
(Note-3) : Special relays (M9000 to M9255) and dedicated devices (M2000 to M2399) cannot be set.
(Note-4) : PX and PY cannot be set.

(3) When data are read normally from the target CPU specified with (S1), the reading complete flag M9216 to M9219 (CPU No.1 : M9216, CPU No.2 : M9217, CPU No.3 : M9218, CPU No.4 : M9219) corresponding to the target CPU turns on. If data cannot be read normally, the reading complete flag of the target CPU does not turn on.

(4) Adjust an executive task, the number of transfer word referring to the operation processing time so that this instruction may not obstruct the execution of the motion operation because processing time becomes long in argument to the Number of words (n) to be read.

(5) When multiple MULTR instructions are executed to the same CPU simultaneously, the reading complete flag M9216 to M9219 turns on/as a result of MULTR that it is executed at the end.

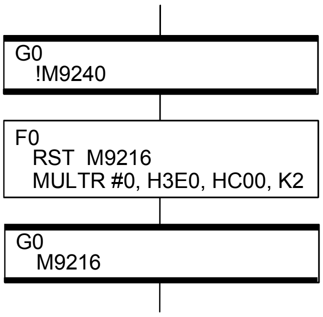
(6) Reset the reading complete flag (M9126 to M9219) using the user program.

[Errors]

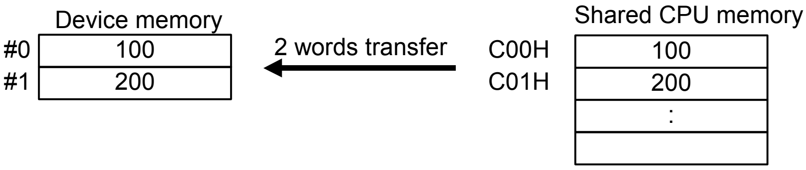
(1) An operation error will occur if:
- Number of words (n) to be read is outside the range of 1 to 256.
- The shared CPU memory first address (S2) of the data which it will be read is outside the range (000H to FFFH) of the shared CPU memory address.
- The shared CPU memory first address (S2) of the data which it will be read + number of words (n) to be read is outside the range (000H to FFFH) of the shared CPU memory address.
- First device No. (D) which stores the reading data + number of words (n) to be read is outside the device range.
- Except 3E0H/3E1H/3E2H/3E3H is set at (S1).
- The self CPU is specified with (S1).
- The CPU which reads is resetting.
- The errors are detected in the CPU which read.
- (D) is a bit device and device number is not a multiple of 16.
- PX/PY is set in (D) to (D)+(n-1).

[Program examples]

(1) It checks that a CPU No.1 is not resetting, 2 words is read to since #0 from the shared CPU memory C00H of CPU No.1, and transits to next step after reading completion.



```
G0
  !M9240

F0
  RST  M9216
  MULTR #0, H3E0, HC00, K2

G0
  M9216
```



F/FS	G
○	○

5.13.8 Write device data to intelligent function module/special function module : TO

Format	TO(D1), (D2), (S), (n)

Number of basic steps	7

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)			
(D1)	—	○	—	—	—	○	—	—	—	—	—
(D2)	—	○	—	—	—	○	—	—	—	—	—
(S)	○ (Note-1)	○	—	—	—	—	—	—	—	—	—
(n)	—	○	—	—	—	○	—	—	—	—	—

○ : Usable

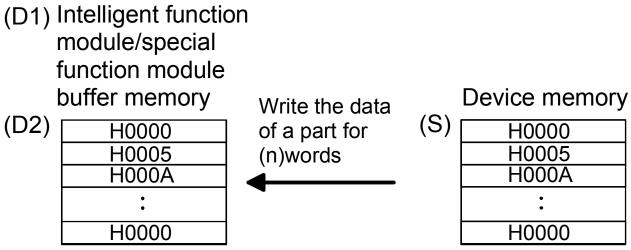
(Note-1) : Refer to the Section "1.2.5" for the correspondence version of the Motion CPU and the software.

[Setting data]

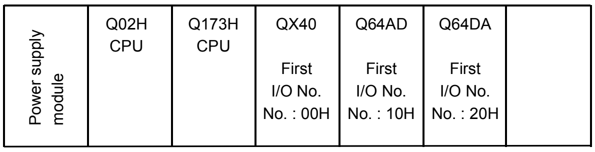
Setting data	Description	Data type of result
(D1)	First I/O No. of the intelligent function module / special function module(000H to FF0H)	—
(D2)	First address of the buffer memory which writes data.	
(S)	First device No. which writing data are stored.	
(n)	Number of words to be written (1 to 256)	

[Functions]

(1) A part for (n)words of data from device specified with (S) are written to since address specified with (D2) of the buffer memory in the intelligent function module/special function module controlled by the self CPU specified with (D1).



(2) First I/O No. of the module set by system setting is specified by (D1).



(D1) sets 20H by the system setting when a TO instruction is executed in the D/A conversion module (Q64DA).

(3) The word devices that may be set at (D1), (D2), (S) and (n) are shown below.

Setting data	Word devices (Note-1)			Bit devices (Note-1), (Note-2)				
	Dn	Wn	#n	Mn	Bn	Fn	Xn	Yn
(D1)	○	○	○	—	—	—	—	—
(D2)	○	○	○	—	—	—	—	—
(S)	○	○	○	○	○	○	○ (Note-3)	○ (Note-3)
(n)	○	○	○	—	—	—	—	—

(Note-1) : The device No. cannot be specified indirectly.
(Note-2) : Specify a multiple of 16 as the device number of bit data.
(Note-3) : PX and PY cannot be set.

(4) Adjust an executive task, the number of transfer word referring to the operation processing time so that this instruction may not obstruct the execution of the motion operation because processing time becomes long in argument to the number of words (n) to be written.

(5) The following analogue modules can be used as the control module of Motion CPU.

- Q62DA
- Q64DA
- Q68DAV
- Q68DAI
- Q64AD
- Q68ADV
- Q68ADI

[Errors]

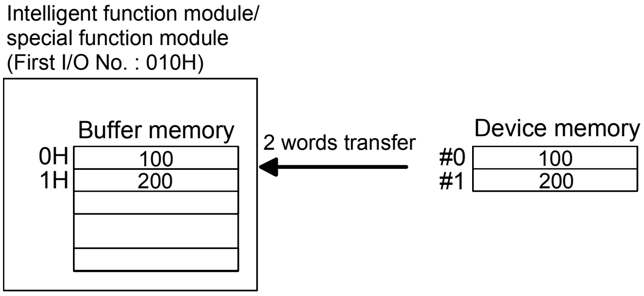
(1) An operation error will occur if:

- Number of words (n) to be written is outside the range of 1 to 256.
- Motion CPU cannot communicate with intelligent function module/special function module at the instruction execution.
- Abnormalities of the intelligent function module/special function module were detected at the instruction execution.
- I/O No.s specified with (D1) differ from the intelligent function module/special function module controlled by the self CPU.
- The address specified with (D2) is outside the buffer memory range.
- First device No. (S) which writing data are stored + number of words (n) to be written is outside the device range.
- (S) is a bit device and device number is not a multiple of 16.
- PX/PY is set in (S) to (S)+(n-1).

[Program examples]

(1) 2 words from #0 is written to since buffer memory address of the Intelligent function module/special function module (First I/O No. : 010H).

```
TO H010, H0, #0, K2
```



F/FS	G
○	○

5.13.9 Read device data from intelligent function module/special function module : FROM

Format	FROM(D), (S1), (S2), (n)

Number of basic steps	7

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	○ (Note-1)	○	—	—	—	—	—	—	—	—	—
(S1)	—	○	—	—	—	○	—	—	—	—	—
(S2)	—	○	—	—	—	○	—	—	—	—	—
(n)	—	○	—	—	—	○	—	—	—	—	—

○ : Usable

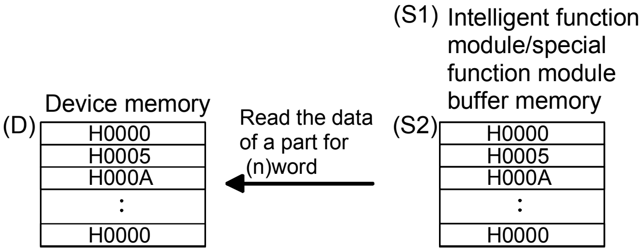
(Note-1) : Refer to the Section "1.2.5" for the correspondence version of the Motion CPU and the software.

[Setting data]

Setting data	Description	Data type of result
(D)	First device No. which stores the reading data.	—
(S1)	First I/O No. of the intelligent function module / special function module (000H to FF0H)	
(S2)	First address of the buffer memory which it will be read.	
(n)	Number of words to be read (1 to 256)	

[Functions]

(1) A part for (n)words of data are read from the address specified with (S2) of the buffer memory in the intelligent function module/special function module controlled by the self CPU specified with (S1), and are stored since the device specified with (S2).



(2) First I/O No. of the module set by system setting is specified by (D1).

Power supply module	Q02H CPU	Q173H CPU	QX40 First device No. No. : 00H	Q64AD First device No. No. : 10H	Q64DA First device No. No. : 20H	

(D1) sets 20H by the system setting when a TO instruction is executed in the D/A conversion module (Q64DA).

(3) The word devices that may be set at (D), (S1), (S2) and (n) are shown below.

Setting data	Word devices (Note-1)			Bit devices (Note-1), (Note-2)				
	Dn	Wn	#n	Mn	Bn	Fn	Xn	Yn
(D)	○	○	○	○ (Note-3)	○	○	○ (Note-4)	○ (Note-4)
(S1)	○	○	○	—	—	—	—	—
(S2)	○	○	○	—	—	—	—	—
(n)	○	○	○	—	—	—	—	—

(Note-1) : The device No. cannot be specified indirectly.

(Note-2) : Specify a multiple of 16 as the device number of bit data.

(Note-3) : Special relays (M9000 to M9255) and dedicated devices (M2000 to M2399) cannot be set.

(Note-4) : PX and PY cannot be set.

(4) Adjust an executive task, the number of transfer word referring to the operation processing time so that this instruction may not obstruct the execution of the motion operation because processing time becomes long in argument to the Number of words (n) to be read.

(5) The following analogue modules can be used as the control module of Motion CPU.

- Q62DA
- Q64DA
- Q68DAV
- Q68DAI
- Q64AD
- Q68ADV
- Q68ADI

[Errors]

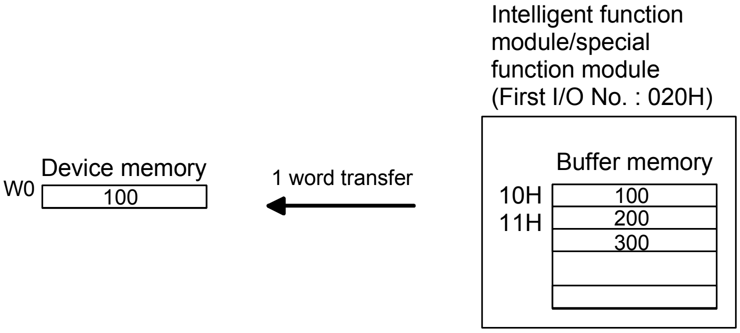
(1) An operation error will occur if:

- Number of words (n) to be read is outside the range of 1 to 256.
- Motion CPU cannot communicate with intelligent function module/special function module at the instruction execution.
- Abnormalities of the intelligent function module/special function module were detected at the instruction execution.
- I/O No.s specified with (S1) differ from the intelligent function module/special function module controlled by the self CPU.
- The address specified with (S2) is outside the buffer memory range.
- First device No. (D) which stores the reading data + number of words (n) to be read is outside the device range.
- (D) is a bit device and device number is not a multiple of 16.
- PX/PY is set in (D) to (D)+(n-1).

[Program examples]

(1) 1 word is read from the buffer memory address 10H of the intelligent function module/special function module (First I/O No. : 020H), and is stored in W0.

```
FROM W0, H020, H10, K1
```



F/FS	G
—	○

5.13.10 Time to wait : TIME

Format	TIME(S)

Number of basic steps	7

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)	—	○	○	—	—	○	○	—	—	—	—

○ : Usable

[Setting data]

Setting data	Description	Data type of result
(S)	Waiting time (0 to 2147483647)[ms]	Logical type (true/false)

[Functions]

(1) A wait state continues for the time specified with (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.

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

[Errors]

(1) An operation error will occur if:
- (S) is an indirectly specified device and its device No. is outside the range. ; or
- The data (device data at indirect specification) specified with (S) is outside the range of 0 to 2147483647.

[Program examples]

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

```
TIME K60000
```

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

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

(3) 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 Motion 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.)

(4) Another transition program (Gn) can executed a time of instruction by multiple Motion SFC program simultaneously. (Multi active step less than 256.)

(5) While time by TIME instruction waits, the wait time can not be stopped.

(6) When using the TIME instruction, a verification error may occur, even when the Motion SFC program of SW6RN-GSV☐P is equal to the Motion CPU, if a verification of Motion SFC program is executed.

F/FS	G
○	○

5.14 Comment Statement : //

Format	//

Number of basic steps	—

[Usable data]

Setting data	Usable Data										
	Bit device	Word device				Constant			Calculation expression	Bit conditional expression	Comparison 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)			
—	—	—	—	—	—	—	—	—	—	—	—

○ : Usable

[Setting data]

There are no setting data.

[Functions]

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

[Errors]

(1) There are no operation errors for comment: //.

[Program examples]

(1) Example which has commented a substitution program.

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

6. TRANSITION PROGRAMS

6.1 Transition Programs

(1) Transition programs

(a) Substitution operation expressions, motion-dedicated functions, bit device control commands and transition conditions can be set in transition programs.

(b) Multiple blocks can be set in one transition program.

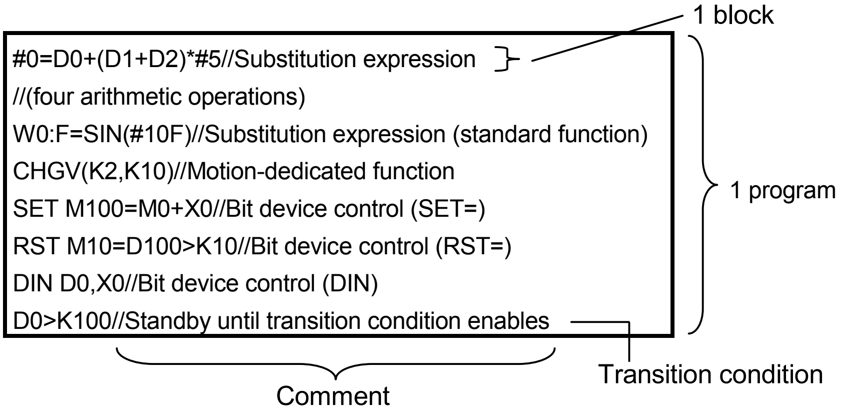
(c) There are no restrictions on the number of blocks that may be set in a single transition program.
Note that one program is within 64k bytes.

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

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

(f) As a special transition program, a program which only no operation (NOP) is set in one block can be created.
This program is used 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. Refer to Section "4.9 Branches, Couplings" for details.

A transition program example is shown below.



```
#0=D0+(D1+D2)*#5//Substitution expression
//(four arithmetic operations)
W0:F=SIN(#10F)//Substitution expression (standard function)
CHGV(K2,K10)//Motion-dedicated function
SET M100=M0+X0//Bit device control (SET=)
RST M10=D100>K10//Bit device control (RST=)
DIN D0,X0//Bit device control (DIN)
D0>K100//Standby until transition condition enables
```

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 executed 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, the bit or comparison conditional expression specified with (S) is not omissible.

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

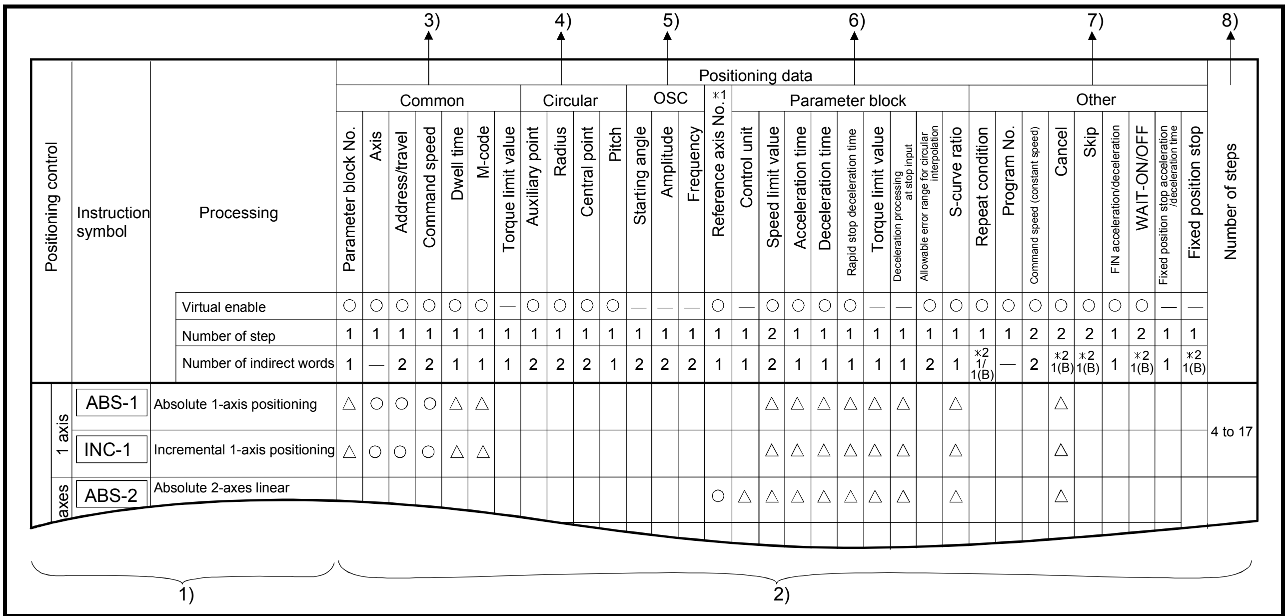
7. MOTION CONTROL PROGRAMS

7.1 Servo Instruction List

Table 7.1 lists servo instructions used in servo programs.
Refer to Section 7.2 to 7.4 for details of the current value change control (CHGA, CHGA-E, CHGA-C).
Refer to the "Q173HCPU/Q172HCPU Motion Controller (SV13/SV22) Programming Manual (REAL MODE)" for other servo instructions.

(1) Guide to servo instruction list

Table 7.1 Guide to Servo Instruction List



Positioning control	Instruction symbol	Processing	Positioning data																																	Number of steps
			Common							Circular				OSC			*1	Parameter block									Other									
			Parameter block No.	Axis	Address/travel	Command speed	Dwell time	M-code	Torque limit value	Auxiliary point	Radius	Central point	Pitch	Starting angle	Amplitude	Frequency	Reference axis No.	Control unit	Speed limit value	Acceleration time	Deceleration time	Rapid stop deceleration time	Torque limit value	Deceleration processing at stop input	Allowable error range for circular interpolation	S-curve ratio	Repeat condition	Program No.	Command speed (constant speed)	Cancel	Skip	FIN acceleration/deceleration	WAIT-ON/OFF	Fixed position stop acceleration /deceleration time	Fixed position stop	
		Virtual enable	○	○	○	○	○	○	—	○	○	○	○	—	—	—	○	—	○	○	○	○	—	—	○	○	○	○	○	○	○	○	○	—	—	
		Number of step	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	1	2	2	2	1	2	1	1	
		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	*2 1/ 1(B)	—	2	*2 1(B)	*2 1(B)	1	*2 1(B)	1	*2 1(B)	
1 axis	ABS-1	Absolute 1-axis positioning	△	○	○	○	△	△											△	△	△	△	△	△		△				△						4 to 17
	INC-1	Incremental 1-axis positioning	△	○	○	○	△	△											△	△	△	△	△	△		△				△						
axes	ABS-2	Absolute 2-axes linear															○	△	△	△	△	△	△	△		△				△						

Number	Description	
1)	Instruction symbol	Gives the servo instructions usable in servo programs.
	Processing	Gives the processing outlines of the servo instructions.
2)	(a) Indicates positioning data which can be set in servo instructions. 1) ○: Item which must be set (Data which cannot execute the servo instruction unless it sets.) 2) △: Item which is set when required (Data which will be controlled by the default value unless it sets.)	
	(b) Allows direct or indirect designation (except axis No.) 1) Direct designation : Set with numerical value. 2) Indirect designation : Set with word device (D, W, #). • Servo program execution 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 No..	
	(c) 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 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							Circular			
				Parameter block No.	Axis	Address/travel value	Command speed	Dwell time	M -code	Torque limit value	Auxiliary point	Radius	Central point	Pitch
			Virtual enable	○	○	○	○	○	○	—	○	○	○	○
			Number of steps	1	1	1	1	1	1	1	1	1	1	1
			Number of indirect words	1	—	2	2	1	1	1	2	2	2	1
Linear interpolation control	1 axis	ABS-1	Absolute 1-axis positioning	△	○	○	○	△	△					
		INC-1	Incremental 1-axis positioning	△	○	○	○	△	△					
	2 axes	ABS-2	Absolute 2-axes linear interpolation	△	○	○	○	△	△					
		INC-2	Incremental 2-sxes linear interpolation	△	○	○	○	△	△					
	3 axes	ABS-3	Absolute 3-axes linear interpolation	△	○	○	○	△	△					
		INC-3	Incremental 3-axes linear interpolation	△	○	○	○	△	△					
	4 axes	ABS-4	Absolute 4-axes linear interpolation	△	○	○	○	△	△					
		INC-4	Incremental 4-axes linear interpolation	△	○	○	○	△	△					
Circular interpolation control	Auxiliary point-specified	ABS⌒	Absolute auxiliary point-specified circular interpolation	△	○	○	○	△	△		○			
		INC ⌒	Incremental auxiliary point-specified circular interpolation	△	○	○	○	△	△		○			
	Radius-specified	ABS↷	Absolute radius-specified circular interpolation less than CW 180°	△	○	○	○	△	△			○		
		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	△	○	○	○	△	△			○		

Positioning data																						Number of steps
OSC			*1	Parameter block									Others									
Starting angle	Amplitude	Frequency	Reference axis No.	Control unit	Speed limit value	Acceleration time	Deceleration time	Rapid stop deceleration time	Torque limit value	Deceleration processing at stop input	Allowable error range for circular interpolation	S-curve ratio	Repeat condition	Program No.	Command speed (constant speed)	Cancel	Skip	FIN acceleration/deceleration	WAIT-ON/OFF	Fixed position stop acceleration /deceleration time	Fixed position stop	
—	—	—	○	—	○	○	○	○	—	—	○	○	○	○	○	○	○	○	○	—	—	
1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	1	2	1	1	
2	2	2	1	1	2	1	1	1	1	1	2	1	*2 1/ 1(B)	—	2	*2 1(B)	*2 1(B)	1	*2 1(B)	1	*2 1(B)	
					△	△	△	△	△	△		△				△						4 to 17
					△	△	△	△	△	△		△				△						
			○	△	△	△	△	△	△	△		△				△						5 to 20
			○	△	△	△	△	△	△	△		△				△						
			○	△	△	△	△	△	△	△		△				△						7 to 21
			○	△	△	△	△	△	△	△		△				△						
			○	△	△	△	△	△	△	△		△				△						8 to 22
			○	△	△	△	△	△	△	△		△				△						
				△	△	△	△	△	△	△	△	△				△						7 to 22
				△	△	△	△	△	△	△	△	△				△						
				△	△	△	△	△	△	△	△	△				△						6 to 21
				△	△	△	△	△	△	△	△	△				△						
				△	△	△	△	△	△	△	△	△				△						
				△	△	△	△	△	△	△	△	△				△						
				△	△	△	△	△	△	△	△	△				△						
				△	△	△	△	△	△	△	△	△				△						
				△	△	△	△	△	△	△	△	△				△						
				△	△	△	△	△	△	△	△	△				△						

○ : Must be set. △ : Set if required.

*1 : Only reference axis speed specification.

*2 : (B) indicates a bit device.

Table 7.2 Servo Instruction List (continued)

Positioning control		Instruction symbol	Processing	Positioning data										
				Common							Circular			
				Parameter block No.	Axis	Address/travel value	Command speed	Dwell time	M-code	Torque limit value	Auxiliary point	Radius	Central point	Pitch
			Virtual enable	○	○	○	○	○	○	—	○	○	○	○
			Number of steps	1	1	1	1	1	1	1	1	1	1	1
			Number of indirect words	1	—	2	2	1	1	1	2	2	2	1
Circular interpolation control	Central point-specified	ABS ↷	Absolute central point-specified circular interpolation CW	△	○	○	○	△	△				○	
		ABS ↶	Absolute central point-specified circular interpolation CCW	△	○	○	○	△	△				○	
		INC ↷	Incremental central point-specified circular interpolation CW	△	○	○	○	△	△				○	
		INC ↶	Incremental central point-specified circular interpolation CCW	△	○	○	○	△	△				○	
Helical interpolation control	Auxiliary point-specified	ABH ⌒	Absolute auxiliary point- specified helical interpolation	△	○	○	○	△	△		○			○
		INH ⌒	Incremental auxiliary point- specified helical interpolation	△	○	○	○	△	△		○			○
	Radius-specified	ABH ↷	Absolute radius-specified helical interpolation less than CW 180°	△	○	○	○	△	△			○		○
		ABH ⟳	Absolute radius-specified helical interpolation CW 180° or more	△	○	○	○	△	△			○		○
		ABH ↶	Absolute radius-specified helical interpolation less than CCW 180°	△	○	○	○	△	△			○		○
		ABH ⟲	Absolute radius-specified helical interpolation CCW 180° or more	△	○	○	○	△	△			○		○
		INH ↷	Incremental radius-specified helical interpolation less than CW 180°	△	○	○	○	△	△			○		○
		INH ⟳	Incremental radius-specified helical interpolation CW 180° or more	△	○	○	○	△	△			○		○
		INH ↶	Incremental radius-specified helical interpolation less than CCW 180°	△	○	○	○	△	△			○		○
		INH ⟲	Incremental radius-specified helical interpolation CCW 180° or more	△	○	○	○	△	△			○		○
	Central point-specified	ABH ↷	Absolute central point-specified helical interpolation CW	△	○	○	○	△	△				○	○
		ABH ↶	Absolute central point-specified helical interpolation CCW	△	○	○	○	△	△				○	○
		INH ↷	Incremental central point-specified helical interpolation CW	△	○	○	○	△	△				○	○
		INH ↶	Incremental central point-specified helical interpolation CCW	△	○	○	○	△	△				○	○

Positioning data																						Number of steps
OSC			*1	Parameter block									Others									
Starting angle	Amplitude	Frequency	Reference axis No.	Control unit	Speed limit value	Acceleration time	Deceleration time	Rapid stop deceleration time	Torque limit value	Deceleration processing at stop input	Allowable error range for circular interpolation	S-curve ratio	Repeat condition	Program No.	Command speed (constant speed)	Cancel	Skip	FIN acceleration/deceleration	WAIT-ON/OFF	Fixed position stop acceleration /deceleration time	Fixed position stop	
—	—	—	○	—	○	○	○	○	—	—	○	○	○	○	○	○	○	○	○	—	—	
1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	1	2	1	1	
2	2	2	1	1	2	1	1	1	1	1	2	1	*2 1/ 1(B)	—	2	*2 1(B)	*2 1(B)	1	*2 1(B)	1	*2 1(B)	
				△	△	△	△	△	△	△	△	△				△						7 to 22
				△	△	△	△	△	△	△	△	△				△						
				△	△	△	△	△	△	△	△	△				△						
				△	△	△	△	△	△	△	△	△				△						
				△	△	△	△	△	△	△		△				△						10 to 27
				△	△	△	△	△	△	△		△				△						
				△	△	△	△	△	△	△		△				△						9 to 26
				△	△	△	△	△	△	△		△				△						
				△	△	△	△	△	△	△		△				△						
				△	△	△	△	△	△	△		△				△						
				△	△	△	△	△	△	△		△				△						
				△	△	△	△	△	△	△		△				△						
				△	△	△	△	△	△	△		△				△						
				△	△	△	△	△	△	△		△				△						
				△	△	△	△	△	△	△		△				△						10 to 27
				△	△	△	△	△	△	△		△				△						
				△	△	△	△	△	△	△		△				△						
				△	△	△	△	△	△	△		△				△						

○ : Must be set. △ : Set if required.
*1 : Only reference axis speed specification.
*2 : (B) indicates a bit device.

Table 7.2 Servo Instruction List (continued)

Positioning control		Instruction symbol	Processing		Positioning data										
					Common							Circular			
					Parameter block No.	Axis	Address/travel value	Command speed	Dwell time	M-code	Torque limit value	Auxiliary point	Radius	Central point	Pitch
				Virtual enable	○	○	○	○	○	○	—	○	○	○	○
				Number of steps	1	1	1	1	1	1	1	1	1	1	1
				Number of indirect words	1	—	2	2	1	1	1	2	2	2	1
Fixed-pitch feed	1 axis	FEED-1	1-axis fixed-pitch feed start		△	○	○	○	△	△					
	2 axes	FEED-2	2-axes linear interpolation fixed-pitch feed start		△	○	○	○	△	△					
	3 axes	FEED-3	3-axes linear interpolation fixed-pitch feed start		△	○	○	○	△	△					
Speed control (Ⅰ)	Forward rotation	VF	Speed control (Ⅰ) forward rotation start		△	○		○		△					
	Reverse rotation	VR	Speed control (Ⅰ) reverse rotation start		△	○		○		△					
Speed control (Ⅱ)	Forward rotation	VVF	Speed control (Ⅱ) forward rotation start		△	○		○		△	△				
	Reverse rotation	VVR	Speed control (Ⅱ) reverse rotation start		△	○		○		△	△				
Speed-position control	Forward rotation	VPF	Speed-position control forward rotation start		△	○	○	○	△	△	△				
	Reverse rotation	VPR	Speed-position control reverse rotation start		△	○	○	○	△	△	△				
	Restart	VPSTART	Speed-position control restart			○									
Speed-switching control		VSTART	Speed-switching control start		△										
		VEND	Speed-switching control end												
		ABS-1	Speed-switching control end point address			○	○	○	△	△	△				
		ABS-2				○	○	○	△	△	△				
		ABS-3				○	○	○	△	△	△				
		INC-1	Travel value up to speed-switching control end point			○	○	○	△	△	△				
		INC-2				○	○	○	△	△	△				
		INC-3				○	○	○	△	△	△				
		VABS	Speed-switching point absolute specification				○	○		△	△				
		VINC	Speed-switching point incremental specification				○	○		△	△				

Positioning data																						Number of steps
OSC			*1	Parameter block									Others									
Starting angle	Amplitude	Frequency	Reference axis No.	Control unit	Speed limit value	Acceleration time	Deceleration time	Rapid stop deceleration time	Torque limit value	Deceleration processing at stop input	Allowable error range for circular interpolation	S-curve ratio	Repeat condition	Program No.	Command speed (constant speed)	Cancel	Skip	FIN acceleration/deceleration	WAIT-ON/OFF	Fixed position stop acceleration /deceleration time	Fixed position stop	
—	—	—	○	—	○	○	○	○	—	—	○	○	○	○	○	○	○	○	○	—	—	
1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	1	2	1	1	
2	2	2	1	1	2	1	1	1	1	1	2	1	*2 1/ 1(B)	—	2	*2 1(B)	*2 1(B)	1	*2 1(B)	1	*2 1(B)	
					△	△	△	△	△	△		△				△						4 to 17
				△	△	△	△	△	△	△		△				△						5 to 19
				△	△	△	△	△	△	△		△				△						7 to 21
					△	△	△	△	△	△		△				△						3 to 15
					△	△	△	△	△	△		△				△						
					△	△	△	△	△	△		△				△						3 to 16
					△	△	△	△	△	△		△				△						
					△	△	△	△	△	△		△				△						4 to 18
					△	△	△	△	△	△		△				△						
																△						2 to 4
				△	△	△	△	△	△	△		△				△						1 to 13
																						1
																△						4 to 9
																△						5 to 10
																△						7 to 12
																△						4 to 9
																△						5 to 10
																△						7 to 12
																						4 to 6

○ : Must be set. △ : Set if required.
*1 : Only reference axis speed specification.
*2 : (B) indicates a bit device.

Table 7.2 Servo Instruction List (continued)

Positioning control		Instruction symbol	Processing	Positioning data										
				Common							Circular			
				Parameter block No.	Axis	Address/travel value	Command speed	Dwell time	M-code	Torque limit value	Auxiliary point	Radius	Central point	Pitch
			Virtual enable	○	○	○	○	○	○	—	○	○	○	○
			Number of steps	1	1	1	1	1	1	1	1	1	1	1
			Number of indirect words	1	—	2	2	1	1	1	2	2	2	1
Speed control with fixed position stop	Forward rotation	PVF	Speed control with fixed position stop absolute specification	△	○	○	○	△	△					
	Reverse rotation	PVR		△	○	○	○	△	△					
Position follow-up control		PFSTART	Position follow-up control start	△	○	○	○		△					
Constant-speed control		CPSTART1	1-axis constant-speed control start	△	○		○							
		CPSTART2	2-axes constant-speed control start	△	○		○							
		CPSTART3	3-axes constant-speed control start	△	○		○							
		CPSTART4	4-axes constant-speed control start	△	○		○							
		ABS-1	Constant-speed control passing point absolute specification		○	○			△	△				
		ABS-2			○	○			△	△				
		ABS-3			○	○			△	△				
		ABS-4			○	○			△	△				
		ABS⌒			○	○			△	△	○			
		ABS↷			○	○			△	△		○		
		ABS↻			○	○			△	△		○		
		ABS↶			○	○			△	△		○		
		ABS↺			○	○			△	△		○		
		ABS⤻			○	○			△	△			○	
		ABS⤺			○	○			△	△			○	
		ABH⌒	Constant-speed control passing point helical absolute specification		○	○			△	△	○			○
		ABH↷			○	○			△	△		○		○
		ABH↻			○	○			△	△		○		○
		ABH↶			○	○			△	△		○		○
		ABH↺			○	○			△	△		○		○
		ABH⤻			○	○			△	△			○	○
		ABH⤺			○	○			△	△			○	○

Positioning data																						Number of steps
OSC			*1	Parameter block									Others									
Starting angle	Amplitude	Frequency	Reference axis No.	Control unit	Speed limit value	Acceleration time	Deceleration time	Rapid stop deceleration time	Torque limit value	Deceleration processing at stop input	Allowable error range for circular interpolation	S-curve ratio	Repeat condition	Program No.	Command speed (constant speed)	Cancel	Skip	FIN acceleration/deceleration	WAIT-ON/OFF	Fixed position stop acceleration /deceleration time	Fixed position stop	
—	—	—	○	—	○	○	○	○	—	—	○	○	○	○	○	○	○	○	○	—	—	
1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	1	2	1	1	
2	2	2	1	1	2	1	1	1	1	1	2	1	*2 1/ 1(B)	—	2	*2 1(B)	*2 1(B)	1	*2 1(B)	1	*2 1(B)	
					△		△	△	△	△		△				△				○	○	6 to 19
					△		△	△	△	△		△				△				○	○	
					△	△	△	△	△	△		△				△						4 to 16
					△	△	△	△	△	△		△				△		△				3 to 15
				△	△	△	△	△	△	△	△	△				△		△				3 to 17
				△	△	△	△	△	△	△	△	△				△		△				4 to17
				△	△	△	△	△	△	△	△	△				△		△				
															△		△		△			2 to 10
															△		△		△			3 to 11
															△		△		△			4 to 12
															△		△		△			5 to 13
															△		△		△			5 to 14
															△		△		△			4 to 13
															△		△		△			
															△		△		△			
															△		△		△			
															△		△		△			5 to 14
															△		△		△			
															△		△		△			9 to 14
															△		△		△			8 to 13
															△		△		△			
															△		△		△			
															△		△		△			
															△		△		△			9 to 14
															△		△		△			

○ : Must be set. △ : Set if required.
*1 : Only reference axis speed specification.
*2 : (B) indicates a bit device.

Table 7.2 Servo Instruction List (continued)

Positioning control	Instruction symbol	Processing	Positioning data										
			Common							Circular			
			Parameter block No.	Axis	Address/travel value	Command speed	Dwell time	M-code	Torque limit value	Auxiliary point	Radius	Central point	Pitch
		Virtual enable	○	○	○	○	○	○	—	○	○	○	○
		Number of steps	1	1	1	1	1	1	1	1	1	1	1
		Number of indirect words	1	—	2	2	1	1	1	2	2	2	1
Constant-speed control	INC-1	Constant-speed control passing point incremental specification		○	○			△	△				
	INC-2			○	○			△	△				
	INC-3			○	○			△	△				
	INC-4			○	○			△	△				
	INC ⌒			○	○			△	△	○			
	INC ↷			○	○			△	△		○		
	INC ⟳			○	○			△	△		○		
	INC ↶			○	○			△	△		○		
	INC ⟲			○	○			△	△		○		
	INC ⤻			○	○			△	△			○	
	INC ⤺			○	○			△	△			○	
	INH ⌒	Constant-speed control passing point helical incremental specification		○	○			△	△	○			○
	INH ↷			○	○			△	△		○		○
	INH ⟳			○	○			△	△		○		○
	INH ↶			○	○			△	△		○		○
	INH ⟲			○	○			△	△		○		○
	INH ⤻			○	○			△	△			○	○
	INH ⤺			○	○			△	△			○	○
	CPEND	Constant-speed control end					△						

Positioning data																						Number of steps
OSC			*1	Parameter block									Others									
Starting angle	Amplitude	Frequency	Reference axis No.	Control unit	Speed limit value	Acceleration time	Deceleration time	Rapid stop deceleration time	Torque limit value	Deceleration processing at stop input	Allowable error range for circular interpolation	S-curve ratio	Repeat condition	Program No.	Command speed (constant speed)	Cancel	Skip	FIN acceleration/deceleration	WAIT-ON/OFF	Fixed position stop acceleration /deceleration time	Fixed position stop	
—	—	—	○	—	○	○	○	○	—	—	○	○	○	○	○	○	○	○	○	—	—	
1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	1	2	1	1	
2	2	2	1	1	2	1	1	1	1	1	2	1	*2 1/ 1(B)	—	2	*2 1(B)	*2 1(B)	1	*2 1(B)	1	*2 1(B)	
															△		△		△			2 to 10
															△		△		△			3 to 11
															△		△		△			4 to 12
															△		△		△			5 to 13
															△		△		△			5 to 14
															△		△		△			4 to 13
															△		△		△			
															△		△		△			
															△		△		△			
															△		△		△			5 to 14
															△		△		△			
															△		△		△			9 to 14
															△		△		△			8 to 13
															△		△		△			
															△		△		△			
															△		△		△			
															△		△		△			9 to 14
															△		△		△			
																						1 to 2

○ : Must be set. △ : Set if required.
*1 : Only reference axis speed specification.
*2 : (B) indicates a bit device.

Table 7.2 Servo Instruction List (continued)

Positioning control	Instruction symbol	Processing	Positioning data										
			Common							Circular			
			Parameter block No.	Axis	Address/travel value	Command speed	Dwell time	M-code	Torque limit value	Auxiliary point	Radius	Central point	Pitch
		Virtual enable	○	○	○	○	○	○	—	○	○	○	○
		Number of steps	1	1	1	1	1	1	1	1	1	1	1
		Number of indirect words	1	—	2	2	1	1	1	2	2	2	1
Repetition of same control (used in speed switching control, constant-speed control)	FOR-TIMES	Repeat range start setting											
	FOR-ON												
	FOR-OFF												
	NEXT	Repeat range end setting											
Simultaneous start	START	Simultaneous start											
Home position return	ZERO	Home position return start		○									
High speed oscillation	OSC	High-speed oscillation	△	○				△					
Current Value change	CHGA	Servomotor/Virtual Servomotor Shaft Current Value Change		○	○								
	CHGA-E	Encoder current value change		○	○								
	CHGA-C	CAM shaft current value change		○	○								

Positioning data																						Number of steps
OSC			*1	Parameter block									Others									
Starting angle	Amplitude	Frequency	Reference axis No.	Control unit	Speed limit value	Acceleration time	Deceleration time	Rapid stop deceleration time	Torque limit value	Deceleration processing at stop input	Allowable error range for circular interpolation	S-curve ratio	Repeat condition	Program No.	Command speed (constant speed)	Cancel	Skip	FIN acceleration/deceleration	WAIT-ON/OFF	Fixed position stop acceleration /deceleration time	Fixed position stop	
—	—	—	○	—	○	○	○	○	—	—	○	○	○	○	○	○	○	○	○	—	—	
1	1	1	1	1	2	1	1	1	1	1	1	1	1	1	2	2	2	1	2	1	1	
2	2	2	1	1	2	1	1	1	1	1	2	1	*2 1/ 1(B)	—	2	*2 1(B)	*2 1(B)	1	*2 1(B)	1	*2 1(B)	
													○									2
													○									
													○									
																						3
														○								2 to 3
																						2
○	○	○							△							△						5 to 10
																						3

○ : Must be set. △ : Set if required.
*1 : Only reference axis speed specification.
*2 : (B) indicates a bit device.

7.2 Servomotor/Virtual Servomotor Shaft Current Value Change

The current value of the specified axis is changed in the real mode.
The current value of the specified virtual servomotor shaft is changed in the virtual mode.

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	Rapid stop deceleration time	Torque limit value	Deceleration processing at stop input	Allowable error range for circular interpolation	S-curve ratio	Cancel	FIN acceleration/deceleration	
CHGA	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 accept 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 accept flag is turned off at completion of the current value change.

(2) The current value of the specified axis is changed in the real mode.

(3) The current value of the specified virtual servo-motor shaft is changed in the virtual mode.

(4) The used axis No. can be set within the following range.

Q172HCPU	Q173HCPU
Axis 1 to 8	Axis 1 to 32

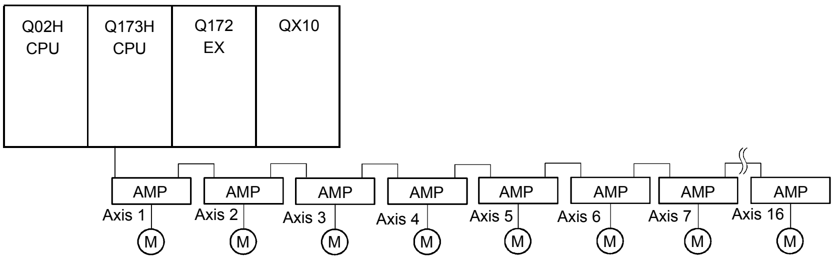
(5) The address which made the current value change by CHGA instruction is valid on the power supply turning on.

[Program example]

A program which made the current value change control in the real mode is described as the following conditions.

(1) System configuration

The current value change control of axis 2 is executed.

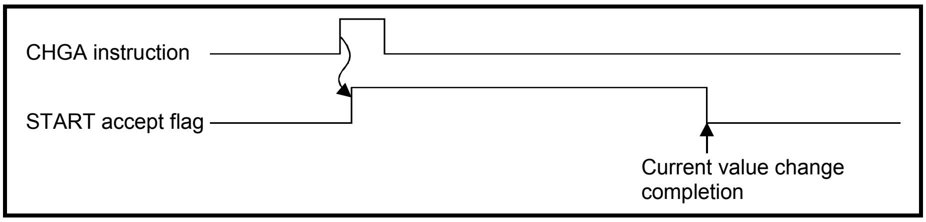


(2) The current value change control conditions

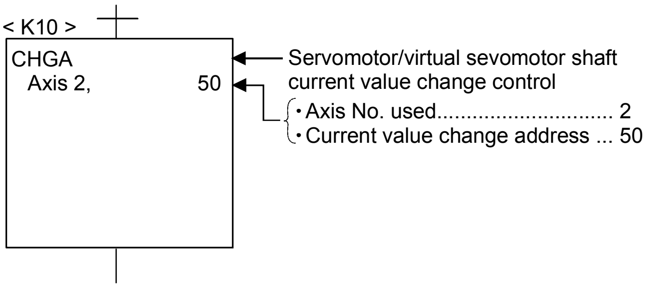
(a) The current value change control conditions are shown below.

Item	Setting
Servo program No.	10
Control axis No.	2
Current value change address	50

(3) Operation timing



(4) Servo program



POINT

(1) Current value changing instructions

- When PLC ready flag (M2000) or PCPU ready flag (M9074) is OFF, a minor error (Note) [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 (Note) [101] (start accept 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 (Note) [1004] occurs and the current value change is not made.
- If the corresponding axis is in a servo error, a major error (Note) [1005] occurs and the current value change is not made.

For SV22

- Set the current value change program of the virtual servomotor shaft within the virtual mode program No. range set in "program mode assignment".
- Set the current value change program of the servomotor (output) shaft within the real mode program No. range.
- If a virtual servomotor shaft current value change is executed in the real mode, a servo program setting error (Note) [903] occurs and the current value change is not made.
- If a servomotor (output) shaft current value change is executed in the virtual mode, a servo program setting error (Note) [904] occurs and the current value change is not made.
- If a current value change is made during mode changing, a servo program setting error (Note) [907] (real → virtual changing) or [908] (virtual → real changing) occurs and the current value change is not made.

(Note) : Refer to the "Q173HCPU/Q172HCPU Motion controller (SV13/SV22) Programming Manual (REAL MODE)"/ "Q173HCPU/Q172HCPU Motion controller (SV22) Programming Manual (VIRTUAL MODE)" for minor error, major error and servo program setting error.

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

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

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	Rapid stop deceleration time	Torque limit value	Deceleration processing at stop input	Allowable error range for circular interpolation	S-curve ratio	Cancel	FIN acceleration/deceleration	
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 (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 at completion of the current value change.

(2) The used axis No. can be set within the following range.

Q172HCPU	Q173HCPU
Axis 1 to 8	Axis 1 to 12

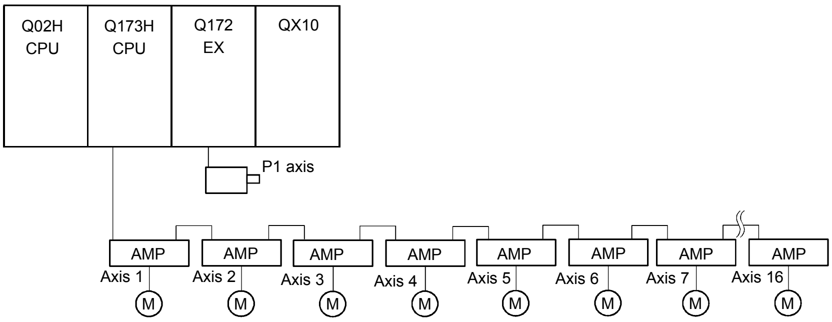
(3) The address which made the current value change by CHGA-E instruction is valid after also the power supply turned off.

[Program example]

A program which made the current value change control of the synchronous encoder shaft is described as the following conditions.

(1) System configuration

The current value change control of the synchronous encoder shaft P1 is executed.

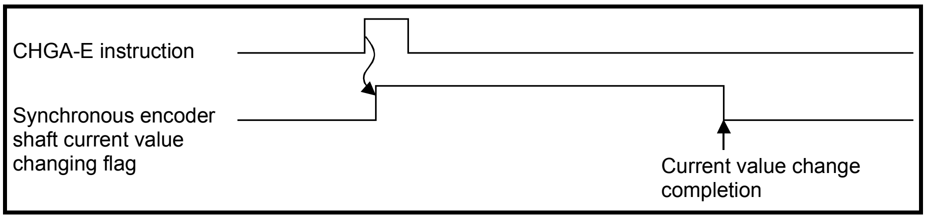


(2) The current value change control conditions

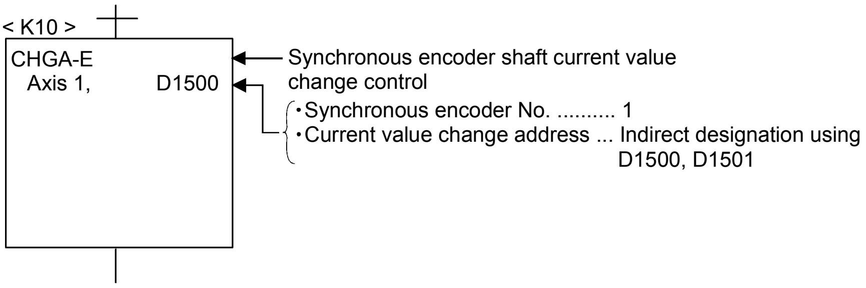
(a) The current value change control conditions are shown below.

Item	Setting
Servo program No.	10
Synchronous encoder No.	1
Current value change address	Indirect designation using D1500, D11501

(3) Operation timing



(4) Servo program



Synchronous encoder shaft current value change control

- Synchronous encoder No. 1
- Current value change address ... Indirect designation using D1500, D1501

POINT

(1) Synchronous encoder current value changing instructions
   - The current value change of the synchronous encoder is executed if operation is being performed in the virtual mode (during pulse input from the synchronous encoder).
     If 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 current value change program of the synchronous encoder shaft program within the virtual mode program No. range set in "program mode assignment".
   - When PLC ready flag (M2000) or PCPU ready flag (M9074) is OFF, a minor error (Note) [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 setting error (Note) [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 No. range, or 905 when it is set to within the real mode program No. range.)
   - If a current value change is made during mode changing, a servo program setting error (Note) [907] (real → virtual changing) or [908] (virtual → real changing) occurs and the current value change is not made.

(Note) : Refer to the "Q173HCPU/Q172HCPU Motion controller (SV13/SV22) Programming Manual (REAL MODE)"/"Q173HCPU/Q172HCPU Motion controller (SV22) Programming Manual (VIRTUAL MODE)" for minor error, major error and servo program setting error.

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

The current value of the specified cam shaft within-one-revolution is changed in the virtual mode.

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	Rapid stop deceleration time	Torque limit value	Deceleration processing at stop input	Allowable error range for circular interpolation	S-curve ratio	Cancel	FIN acceleration/deceleration	
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 address.

(2) The cam shaft may be starting.

(3) The used axis No. can be set within the following range.

Q172HCPU	Q173HCPU
Axis 1 to 8	Axis 1 to 32

(4) The address which made the current value change by the CHGA-C instruction is valid after also the power supply turned off.

[Program example]

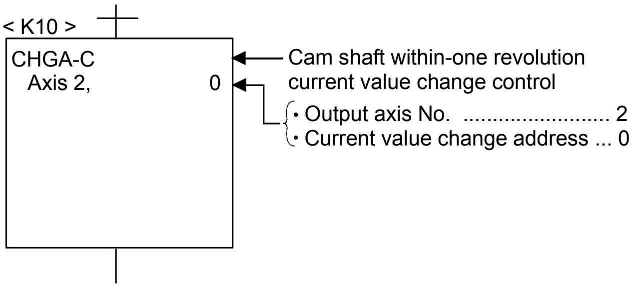
A program which made the current value change control of the cam shaft within-one-revolution current value change is described as the following conditions.

(1) Current value change control conditions

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

Item	Setting
Servo program No.	10
Output axis No.	2
Current value change address	0

(2) Servo program



Cam shaft within-one revolution current value change control

- Output axis No. 2
- Current value change address ... 0

POINT

(1) Cam shaft within-one revolution current value changing instructions

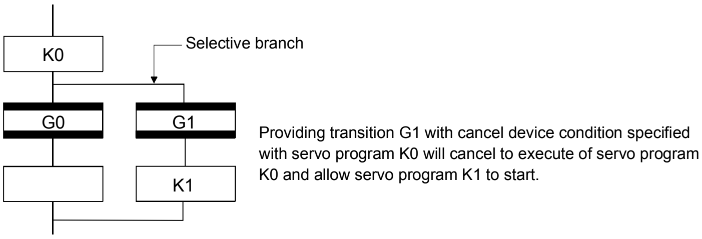
- If a new within-one revolution current value is outside the range 0 to (one-revolution pulse count - 1), a minor error (Note) [6120] occurs and current value change is not.
- Set the current value change program the cam shaft within-one-revolution within the virtual mode program No. range set in "program mode assignment".
- When PLC ready flag (M2000) or PCPU ready flag (M9074) is OFF, a minor error (Note) [100] occurs and a current value change is not made.
- If the cam shaft within-one-revolution current value change is executed in the real mode, a servo program setting error (Note) [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 No. range, or 905 when it is set to within the real mode program No. range.)
- If a current value change is made during mode changing, a servo program setting error (Note) [907] (real → virtual changing) or [908] (virtual → real changing) occurs and the current value change is not made.

(Note) : Refer to the "Q173HCPU/Q172HCPU Motion controller (SV13/SV22) Programming Manual (REAL MODE)"/"Q173HCPU/Q172HCPU Motion controller (SV22) Programming Manual (VIRTUAL MODE)" for minor error, major error and servo program setting error.

7.5 Programming Instructions

7.5.1 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 the Motion 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 the Motion SFC program which exercises control equivalent to a cancel start.



7.5.2 Indirect designation using motion devices

(1) The motion registers #0 to #8191 cannot be used to make indirect specification in the mechanical system programs.
The motion register values are used in the servo or mechanical system programs, substitutes them to data registers (D)/link registers (W).

(2) The coasting timer (FT) cannot used to make indirect specification in the servo program and mechanical system program.

8. MOTION DEVICES

The motion registers (#0 to #8191) and coasting timer (FT) are available as Motion CPU-dedicated devices.
They can be used in operation control (F/FS) programs or transition (G) programs.

8.1 Motion Registers (#0 to #8191)

Motion device	Item	Specifications
Motion register (#)	Number of points	8192 points (#0 to #8191)
	Data size	16-bit/point
	Latch	Only a user device is latched. (All points are cleared by latch clear operation.)
	Usable tasks	Normal, event and NMI
	Access	Read and write enabled in whole range

(1) Motion register list
   (a) Common to all operating system

Device No.	Application	Signal direction
#0 to	User devices (8000 points)	• Cleared by latch clear.
#8000 to	Motion SFC dedicated devices (64 points)	• Cleared at power on or reset only. • Cleared by the Motion SFC error history request flag on (keep at power on or reset).
#8064 to #8191	Servo monitor devices (128 points)	• Cleared at power on or reset only.

8

POINT

The motion registers (#) cannot be set as indirectly specified devices of mechanical system programs.

(a) Motion SFC dedicated devices (#8000 to #8063)

The Motion SFC dedicated devices are shown below.

The device's refresh cycle is indicated when the signal direction is "status", or its fetch cycle when the signal direction is "command".

Device No.	Signal name		Signal direction		Refresh cycle	Fetch cycle
			Status	Command		
#8000 to	Seventh error information in past (Oldest error information)	Motion SFC error history (8 errors) (64 points)	○	—	At error occurrence	—
#8008 to	Sixth error information in past					
#8016 to	Fifth error information in past					
#8024 to	Fourth error information in past					
#8032 to	Third error information in past					
#8040 to	Second error information in past					
#8048 to	First error information in past					
#8056 to #8063	Latest error information					

1) Motion SFC error history devices

The error information occurred is stored as a history of up to eight past errors. The latest error is stored in #8056 to #8063. All errors, including the Motion SFC control errors and the conventional minor, major, servo, servo program and mode changing errors are stored in this history. At error occurrence, the "Motion SFC error detection flag (M2039)" is also set.

Motion SFC error history can be cleared by the Motion SFC error history clear request flag or programming software.

The error information is shown below.

No.	Signal name		Description	
			Motion SFC control errors	Conventional errors
+0	Error Motion SFC program No.		0 to 255 : Motion SFC program No. in error -1 : Independent of Motion SFC program	-1
+1	Error type		1 :F/FS 2 :G -1 :K or other (not any of F/FS, G and SFC chart) -2 :Motion SFC chart	3 : Minor/major error 4 : Minor/major error (virtual servomotor shaft) (SV22 only) 5 : Minor/major error (synchronous encoder shaft) (SV22 only) 6 : Servo error 7 : Servo program setting error 8 : Mode change error (SV22 only) 9 : Manual pulse generator axis setting error 10 : Test mode request error 11 : WDT error 12 : Personal computer link communication error
+2	Error program No.		0 to 4095 : F/FS, G, K program No. 0 to 255 : GSUB program No. -1 : Independent of F/FS, G, K, GSUB	0 to 4095 : Servo program No. when error type is "3", "4" or "7" -1 : Others
+3	Error block No./ Motion SFC list line No./axis No.		0 to 8191 : F/FS or G program's block No. (line No.) when error type is "1" or "2" 0 to 8188 : Motion SFC list line No. 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 No. when error type is any of "3" to "6" -1 : Others
+4	Error code		16000 and later (Refer to Chapter "11 ERROR CODE LISTS".)	• 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 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 clock data at error occurrence (D9025, D9026, D9027) are set. (BCD code, year in its lower 2 digits)	
+6		Day/ hour		
+7		Minute/ second		

2) Motion SFC error detection flag (M2039)
(Refresh cycle : Scan time)

The Motion SFC error detection flag (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) Set the error code to each axis or error devices.

b) Turns on the error detection signal of each axis or error.

c) Set the error information to the above "Motion SFC error history devices (#8000 to #8063)".

d) Turns on the Motion SFC error detection flag (M2039) .

In the user program, reset the "Motion SFC error detection flag (M2039)" after reading the error history at the "Motion SFC error detection flag (M2039)".
After that, "Motion SFC error detection flag (M2039)" turns on again at occurrence of a new error.

POINT

(1) Resetting the "Motion SFC error detection flag (M2039)" will not reset (clear to zero) the "Motion SFC error history devices (#8000 to #8063)".
After power-on, they always controls the error history continuously.

(2) Set the clock data and clock data read request (M9028) in the user program.

(b) Servo monitor devices (#8064 to #8191)

Information about "servo amplifier type", "motor current" and "motor speed" for each axis is stored the servo monitor devices.
The details of the storage data are shown below.

Axis No.	Device No.	Signal name
1	#8064 to #8067	
2	#8068 to #8071	
3	#8072 to #8075	
4	#8076 to #8079	
5	#8080 to #8083	
6	#8084 to #8087	
7	#8088 to #8091	
8	#8092 to #8095	
9	#8096 to #8099	
10	#8100 to #8103	
11	#8104 to #8107	
12	#8108 to #8111	
13	#8112 to #8115	
14	#8116 to #8119	
15	#8120 to #8123	
16	#8124 to #8127	
17	#8128 to #8131	
18	#8132 to #8135	
19	#8136 to #8139	
20	#8140 to #8143	
21	#8144 to #8147	
22	#8148 to #8151	
23	#8152 to #8155	
24	#8156 to #8159	
25	#8160 to #8163	
26	#8164 to #8167	
27	#8168 to #8171	
28	#8172 to #8175	
29	#8176 to #8179	
30	#8180 to #8183	
31	#8184 to #8187	
32	#8188 to #8191	

	Signal name (Note-1)	Signal description	Refresh cycle	Signal direction
+0	Servo amplifier type	0 : Unused 256 : MR-J3-B 257 : MR-J3-B (Fully closed loop control) 258 : MR-J3-B (Linear)	When the servo amplifier power-on	Monitor devise
+1	Motor current	×0.1[%]	Operation cycle 1.7[ms] or less: Operation cycle Operation cycle 3.5[ms] or more: 3.5[ms]	
+2	Motor speed	×0.1[r/min]		
+3				

(Note-1) : The value that the lowest servo monitor device No. was added "+0, +1 ···" on each axis is shown.

8.2 Coasting Timer (FT)

Motion device	Item	Specification
Coasting timer (FT)	Number of points	1 point (FT)
	Data size	32-bit/point (-2147483648 to 2147483647)
	Latch	No latch. Cleared to zero at power-on or reset, a count rise is continued from now on.
	Usable tasks	Normal, event, NMI
	Access	Read only enabled
	Timer specifications	888μs timer (Current value (FT) is incremented by 1 per 888μs.)

9. MOTION SFC PARAMETER

Two different Motion SFC parameters are available: "task parameters" designed to control the tasks (normal task, event task, NMI task) and "program parameters" to be set per Motion SFC program.
Their details are shown below.

9.1 Task Definitions

When to execute the Motion SFC program processing can be set only once in the program parameter per program.
Roughly classified, there are the following three different tasks.

Task type	Contents
Normal task	Executes in motion main cycle (free time).
Event task	1. Executes in fixed cycle (0.88ms, 1.77ms, 3.55ms, 7.11ms, 14.2ms). 2. Executes when the input set to the event task factor among external interrupts (16 points of QI60) turns on. 3. Executes by an interrupt from the PLC CPU.
NMI task	Executes when the input set to the NMI task factor among external interrupts (16 points of QI60) turns on.

9.2 Number of Consecutive Transitions and Task Operation

9.2.1 Number of consecutive transitions

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 the Motion SFC program execution control in the execution cycle of the corresponding task, this operation is performed for the number of active steps to terminate processing once. And the same operation is processed continuously in the next cycle.
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).
In this case, set the number of consecutive transitions.
Control exercised is common to the Motion SFC programs executed by normal tasks.

POINT

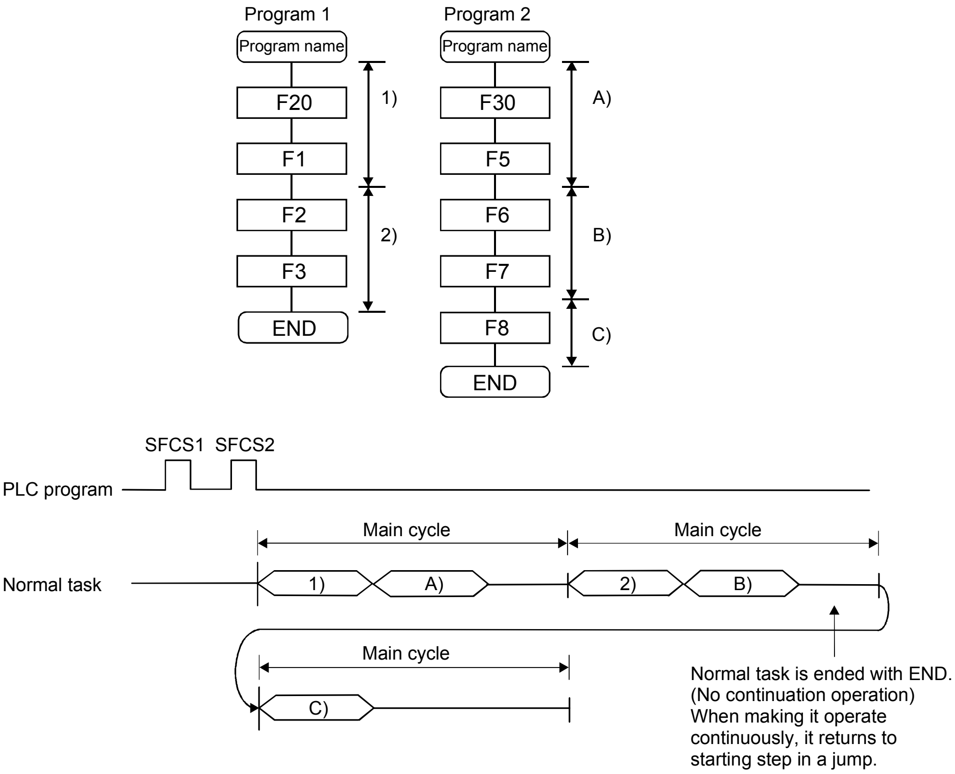
Set the number of consecutive transitions to the Motion SFC programs executed by event and NMI tasks for every program.

9.2.2 Task operation

(1) Normal task operation

[Operations]

The Motion SFC program is executed in the main cycle (free time) of the Motion CPU processing.



Normal task is ended with END. (No continuation operation) When making it operate continuously, it returns to starting step in a jump.

[Points]

(a) The Motion 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 operation

[Operations]

An event task executes the Motion SFC program at occurrence of an event. There are the following events.

(a) Fixed cycle

The Motion SFC program is executed periodically in any of 0.88ms, 1.77ms, 3.55ms, 7.11ms and 14.2ms cycles.

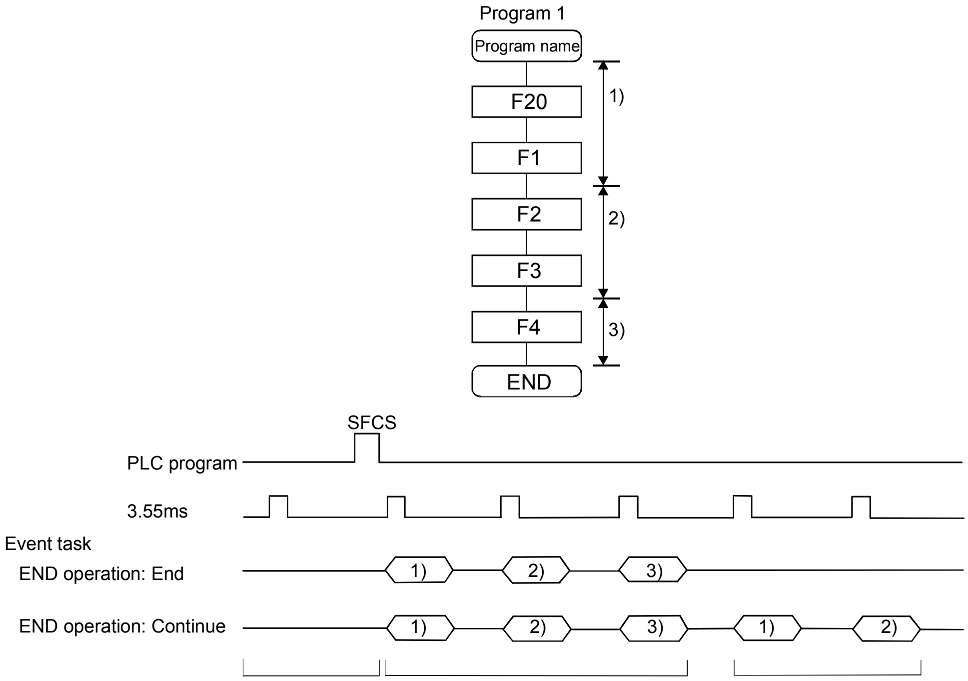
(b) External interrupt (16 points of I0 to I15)

Among 16 points of the QI60 (16-point interrupt module) loaded in the motion slot, the Motion SFC program is run when the input set for an event task turns on.

(c) PLC interrupt

The Motion SFC program is executed when the S(P).GINT instruction is executed in the PLC program.

<Example 1>

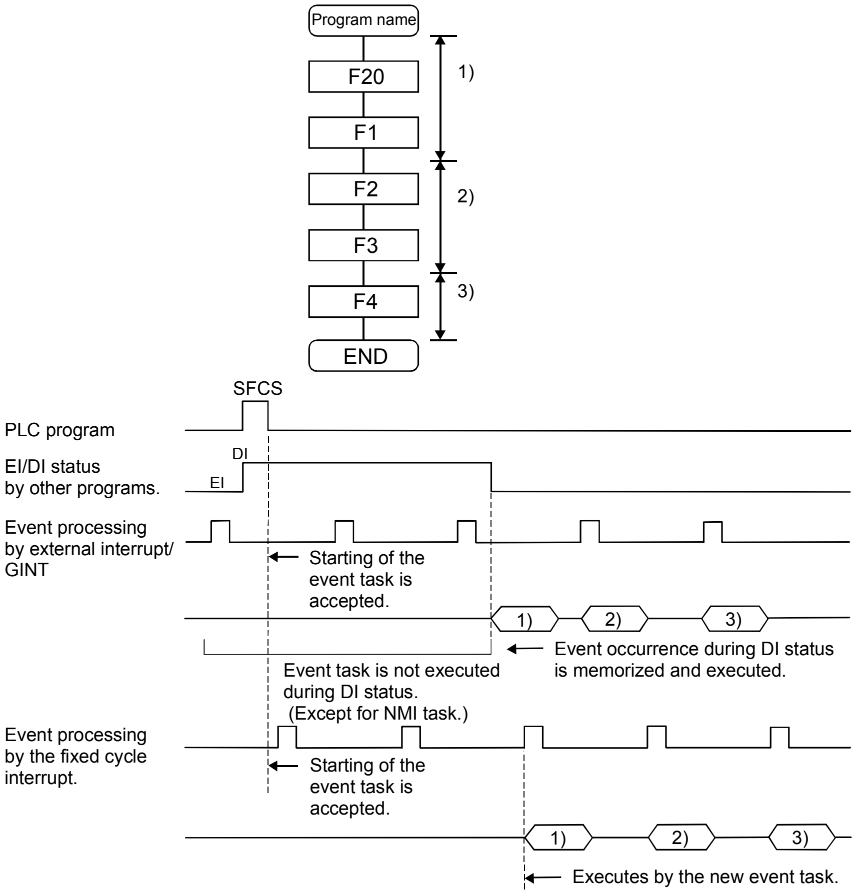


Do not execute a program before the event task starting .
(Note) : When making it always execute, set as automatic starting.

Execute the number of consecutive transition for every event occurrence.

When END operation is set as continuation, continuation operation is executed.
When stopping, it is make to execute CLR by other programs.

<Example 2>



[Points]

(a) Multiple events can be set to one Motion SFC program. However, multiple fixed cycles cannot be set.

(b) Multiple Motion SFC programs can be executed 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.

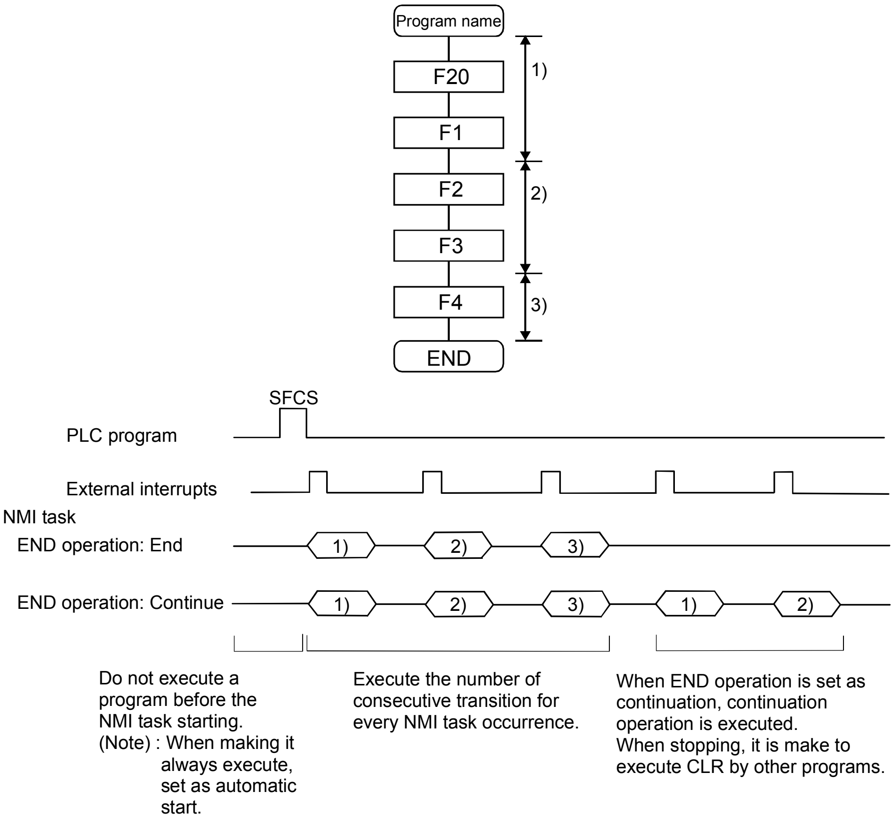
[Errors]

When the motion control step is executed by the Motion SFC program set to the event task, the Motion SFC program error [16113] occurs and stops the Motion SFC program running.

(3) NMI task operation

[Operations]

The Motion SFC program is executed when the input set to the NMI task factor among external interrupts (16 points of QI60) turns on.



[Points]

(a) NMI task has the highest priority among the normal, event and NMI tasks.

(b) If the event task is disabled (DI) by the normal task, the interruption of the NMI task is executed, without being masked.

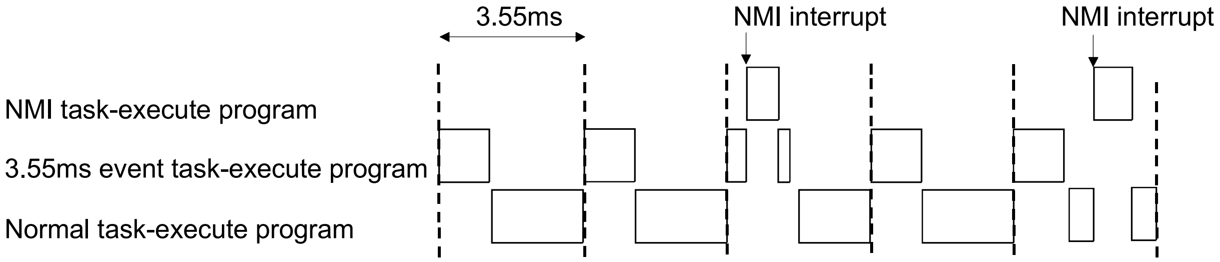
[Errors]

The motion control step is executed during NMI task.

If the motion control step is executed during NMI task, the Motion SFC program error [16113] occurs and stops the Motion SFC program.

9.3 Execution Status of The Multiple Task

Execution status of each Motion SFC program when the Motion SFC program is executed multiple tasks is shown below.



When there are programs which are executed by the NMI task, 3.55ms fixed-cycle even task with a program to run by the NMI task, and the normal task like a chart,

(1) The 3.55ms fixed-cycle event task is executed at intervals of 3.55ms;

(2) The NMI task is executed with the highest priority when an NMI interrupt is input; and

(3) The normal task is executed at free time.
as shown above.

[Points]
One Motion SFC program can be executed 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 Motion SFC program	Normal task
No. 1 Subroutine	Event task (3.55ms cycle)

⚠CAUTION

- A normal task may be hardly executed when a NMI task, an event task are executed in many.

9.4 Task Parameters

No.	Item		Setting item	Initial value	Remark
1	Number of consecutive transitions	Normal task (Normal task common)	1 to 30	3	These parameters are imported when PLC ready flag (M2000) turns off to on and used for control thereafter. When setting/changing the values of these parameters, turns the PLC ready flag (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 the Motion SFC program execution control in the execution cycle of the corresponding task, this operation is performed for the number of active steps to terminate processing once. And the same operation is processed continuously in the next cycle.
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).
In this case, the number of consecutive transitions can be set.
Controls in common to the Motion SFC programs executed by normal tasks.

POINT

Set the number of consecutive transitions to the Motion SFC programs executed by event and NMI tasks for every program.

[Errors]

These parameters are imported and checked when the PLC ready flag (M2000) turns off to on.
When the value that was set is outside the setting range, the following Motion SFC error is set and the initial value is used to control.

Error code (Note)	Error cause		Error processing	Corrective action
	Name	Contents		
17000	Normal task consecutive transition count error	The normal task's consecutive transition count of the Motion SFC program executed by the normal task is outside the range 1 to 30.	The initial value of 3 is used for control.	Turn PLC ready flag (M2000) off, make correction to set the value of within the range, and write it to the CPU.

(Note) : 0000H (normal)

(2) Interrupt setting

[Description]

Set whether 16 interrupt input points (I0 to I15) of the QI60 interrupt 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.

9.5 Program Parameters

Set the following parameters for every Motion SFC program.

No.	Item	Setting range	Initial value	Remark
1	Start setting	Automatically started or not	Not setting	These parameters are imported at starting of the PLC ready flag (M2000) and used for control there after. When setting/changing the values of these parameters, turn PLC ready flag (M2000) off.
2	Execute task	It is 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 It is one of 0.88ms, 1.77ms, 3.55ms, 7.11ms and 14.2ms or none. 2. External interrupt (make selection from those set to event task) Multiple 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 Multiple interrupt can be set from among I0, I1, I2, I3, I4, I5, I6, I7, I8, I9, I10, I11, I12, I13, I14 and I15. 1 to 3 can be set also by OR. (A duplication setting is possible.) The same event can be shared among multiple Motion 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) Multiple 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 toward the program set to the event or NMI task.	1	
4	END operation	End/continue Set the operation mode of the END step toward the program set to the event or NMI task.	End	

POINT

The settings of "END operation" are invalid for the subroutine called program. "END operation" is controlled as "end".

(1) Start setting

[Description]

The following control is changed 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 the PLC ready flag (M2000) turns off to on, the program is executed from the initial (first) step in accordance with the number of consecutive transitions of the normal task.	The program is started by the Motion SFC start instruction ($\boxed{\text{S(P).SFCS}}$) from the PLC or by a subroutine call/start (GSUB) made from the Motion SFC program. • When started by the $\boxed{\text{S(P).SFCS}}$ instruction In the main cycle after execution of the $\boxed{\text{S(P).SFCS}}$ instruction, the program is executed from the initial (first) step in accordance with the number of consecutive transitions of the normal task. • When subroutine started In the (next) main cycle after execution of GSUB, the program is executed from the first step in accordance with the number of consecutive transitions of the normal task. • When subroutine called The program is executed in the same cycle from the first step.
		After that, the program is executed continuously by the number of consecutive transitions of the normal task in the motion main cycle. (The settings of "executed task" and "number of consecutive transitions" of the subroutine called program are invalid. It is controlled as the normal task.)	
2	END control $\boxed{\text{END}}$	Ends the self program. Again, the program is started by the Motion SFC start instruction ($\boxed{\text{S(P).SFCS}}$) from the PLC or by a subroutine call/start (GSUB) made from the Motion 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 starting of the PLC ready flag (M2000), the program is executed from the initial (first) step in accordance with the number of consecutive transitions of the corresponding program.	The program is started by the Motion SFC start instruction ($\boxed{\text{S(P).SFCS}}$) from the PLC or by a subroutine call/start (GSUB) made from within the Motion SFC program. • When started by the $\boxed{\text{S(P).SFCS}}$ instruction At occurrence of a valid event after execution of the $\boxed{\text{S(P).SFCS}}$ 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 executed from the first step in accordance with the number of consecutive transitions of the corresponding program. • When subroutine called The program is executed immediately from the first step.
		After that, the program is executed continuously by the number of consecutive transitions of the corresponding program at occurrence of a valid event. (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 $\boxed{\text{END}}$	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 starting of the PLC ready flag (M2000), the program is executed from the initial (first) step in accordance with the number of consecutive transitions of the corresponding program.	The program is started by the Motion SFC start instruction (S(P).SFCS) from the PLC or by a subroutine call/start (GSUB) made from within the Motion SFC program. • When started by the S(P).SFCS instruction At occurrence of a valid event after execution of the S(P).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 executed from the first step in accordance with the number of consecutive transitions of the corresponding program. • When subroutine called The program is executed immediately from the first step.
		After that, the program is executed continuously by the number of consecutive transitions of the corresponding program at occurrence of a valid event.	
2	END control END	As specified for END operation.	

[Errors]
None.

POINT

In the case of the program which is executed by the normal task, write the program so that it is not ended by END but it returns to the starting step by a jump when starting of the automatically from an initial again.

(2) Execute task

[Description]
Set the timing (task) to execute a program.
Specify whether the program will be run by only one of the "normal task (main cycle), event task (fixed cycle, external interrupt, PLC interrupt) and NMI task (external interrupt)".

When the event task is set, multiple events among the "fixed cycle, external interrupt (for event task) and PLC interrupt".
However, multiple fixed cycles cannot be set toward one Motion SFC program.
Example) Interrupt setting: Inputs for event task I6, I7, I8, I9, I10, I11, I12, I13, I14 and I15

Motion SFC program No. 10 – event : Fixed cycle (3.55ms)
Motion SFC program No. 20 – event :
Fixed cycle (1.77ms) + external interrupt (I6)
Motion SFC program No. 30 – event :
External interrupts (I7, I15) + PLC CPU interrupt

When the NMI task is set, multiple interrupt inputs among the external interrupts (for NMI task) can be set.

Example) Interrupt setting: Inputs for NMI task I0, I1, I2, I3, I4, I5

Motion SFC program No. 10 – NMI : I0
Motion SFC program No. 20 – NMI : I1 + I2
Motion SFC program No. 30 – NMI : I5

[Errors]
This program parameter is imported when the PLC ready flag (M2000) turns off to on, and is checked at starting of the Motion SFC program (automatic start, start from PLC or subroutine start).
When the value is illegal, either of the following Motion SFC errors is set and the initial value is controlled.

Error code (Note)	Error cause		Error processing	Corrective action
	Name	Contents		
17010	Execute task setting is illegal	Multiple events among the normal, event and NMI tasks are set, or one is not set.	The initial value (normal task) is controlled.	Turn PLC ready flag (M2000) off, make correction to set the value of within the range, and write it to the CPU.
17011	Executed task setting is illegal (event)	Two or more fixed cycles of the event task have been set.		

(Note) : 0000H (normal)

POINT

Since the execute task can be set for every Motion SFC program No., multiple programs need not be written for single control (machine operation) to divide execution timing-based processing's.
For example, it can be achieved easily by subroutine starting the areas to be run in fixed cycle and to be run by external interrupt partially in the Motion SFC program run by the normal task.

(3) Number of consecutive transitions

[Description]
Set the number of consecutive transitions to program executed by the event or NMI task for every program.
Refer to Section "9.4 Task Parameters" for number of consecutive transitions.

[Errors]
This program parameter is imported when the PLC ready flag (M2000) turns off to on, and is checked at starting of the Motion SFC program (automatic start, start from PLC or subroutine start).
When the value is illegal, either of the following Motion SFC errors is set and the initial value is controlled.

Error code (Note)	Error cause		Error processing	Corrective action
	Name	Contents		
17001	Event task consecutive transition count error	The number of consecutive transitions of the Motion SFC program started by the event task is outside the range 1 to 10.	The initial value of 1 is controlled.	Turn PLC ready flag (M2000) off, make correction to set the value of within the range, and write it to the CPU.
17002	NMI task consecutive transition count error	The number of consecutive transitions of the Motion SFC program started by the NMI task is outside the range 1 to 10.		

(Note) : 0000H (normal)

(4) END operation

[Description]

Set the operation at execution of the END step toward the program executed by the event or NMI task.

This varies the specifications for the following items.

- Program run by NMI task

No.	Item	When "ended"	When "continued"
1	Control at END execution	Ends the self program.	Ends to execute the self program with this event/interrupt.
2	Restart after END execution	Again, the program is started by the Motion SFC start instruction (S(P).SFCS) from the PLC or by a subroutine call/start (GSUB) made from the Motion SFC program.	Restarted at occurrence of the next event/interrupt, and run from the initial (first) step in accordance with the number of consecutive transitions of the corresponding program. After that, at occurrence of an event/interrupt, the program is executed in accordance with the number of consecutive transitions of the corresponding program.
3	Restart after end by clear step CLR	Again, the program is started by the Motion SFC start instruction (S(P).SFCS) from the PLC or by a subroutine call/start (GSUB) made from the Motion SFC program.	

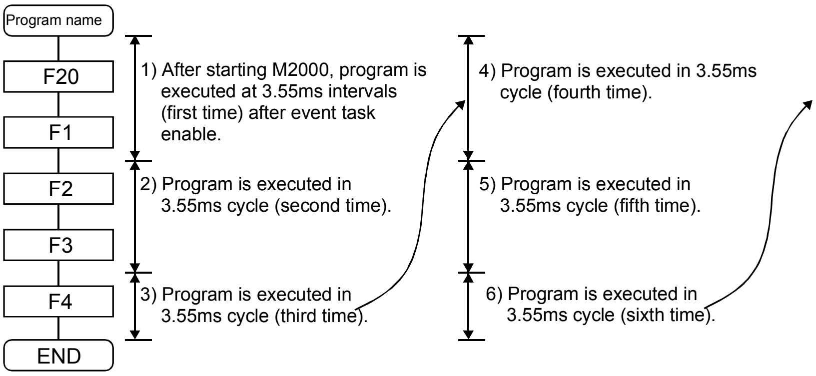
POINT

The END operation of subroutine called program is controlled as an "end".

- The following operation example assumes that the END operation is "continued."

Program parameters

- Automatically started
- Execute task = event 3.55ms
- Number of consecutive transitions = 2
- End operation "continued"



9.6 How to Start The Motion SFC Program

The Motion SFC program is executed during PLC ready flag (M2000) is on.
The Motion SFC program may be started by any of the following three methods.

(1) Automatic start
(2) Start from the Motion SFC program
(3) Start from the PLC

Set the starting method in the program parameter for every Motion SFC program.
Refer to Section "9.5 Program Parameters" for parameter setting.

9.6.1 Automatic start

[Operations]
An automatic start is made by turning PLC ready flag (M2000) on.

9.6.2 Start from the Motion SFC program

[Operations]
A start is made by executing a subroutine call/start step in the SFC program.
Refer to Chapter "4 MOTION SFC PROGRAMS" for details of the subroutine call/start step.

9.6.3 Start from PLC (PLC instruction S(P).SFCS)

The SFC program is started by executing the S(P).SFCS in the PLC program.

Refer to Chapter "3 MOTION DEDICATED PLC INSTRUCTION" for details.

9.7 How to End The Motion SFC Program

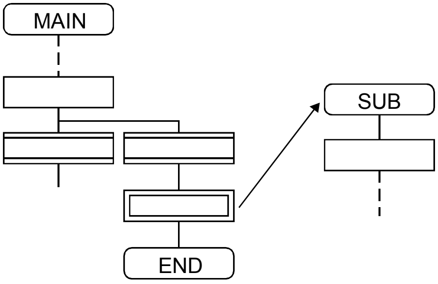
[Operations]
(1) The Motion SFC program is ended by executing END set in itself.
(2) The Motion SFC program is stopped by turning off the PLC ready flag (M2000).
(3) The program can be ended by the clear step.
Refer to Section "4.5.4 Clear step" for details of the clear step.

[Point]
(1) Multiple ENDs can be set in one Motion SFC program.

9.8 How to Change from One Motion SFC Program to Another

Use a subroutine start to stop the Motion SFC program running and switch it to another Motion SFC program.

Motion SFC program changing example using subroutine start



9.9 How to Manage The Executing Program

There are no specific information that indicates which the Motion SFC program is executing. Use a user program (Motion SFC program/PLC program) to control the executing program.

9.10 Operation Performed at CPU Power-Off or Reset

When the CPU is powered off or reset operation is performed, Motion SFC programs run are shown below.

(1) When the CPU is powered off or reset operation is performed, Motion SFC programs stop to execute.

(2) At CPU power-off or key-reset, the contents of the motion registers #0 to #7999 are held. Initialize them in the Motion SFC programs as required.

(3) After CPU power-on or reset processing, Motion SFC programs run is shown below.
  - The SFC programs set to start automatically are run from the beginning by turning PLC ready flag (M2000) on in the PLC program.
  - The other Motion SFC programs are also executed from the first at starting.

9.11 Operation Performed when CPU is Switched from RUN/STOP

When a RUN/STOP switch is operated, PLC ready flag (M2000) turns on/off in accordance with "Operation at STOP to RUN" of a setting of a basic systems.
Refer to Section "3.1.3 Individual parameters" of the "Q173HCPU/Q172HCPU Motion controller Programming Manual (COMMON)" for the details of "Operation at STOP to RUN".
And, refer to the next section for PLC ready flag (M2000) off/on.

9.12 Operation Performed when PLC Ready flag (M2000) Turns OFF/ON

This section explains about the turns off/on of PLC ready flag (M2000).
The on/off condition of PLC ready flag (M2000) differences in "Operation at STOP to RUN" of a setting of a basic systems.
Refer to Section "3.1.3 Individual parameters" of the "Q173HCPU/Q172HCPU Motion controller Programming Manual (COMMON)" for details.

[M2000 OFF → ON]
If there is no fault when PLC ready flag (M2000) turns off to on, the PCPU ready flag (M9074) turns on.
When this PCPU ready flag (M9074) turns on, Motion SFC programs can be executed.
An automatic start Motion SFC program starts execution from the first.

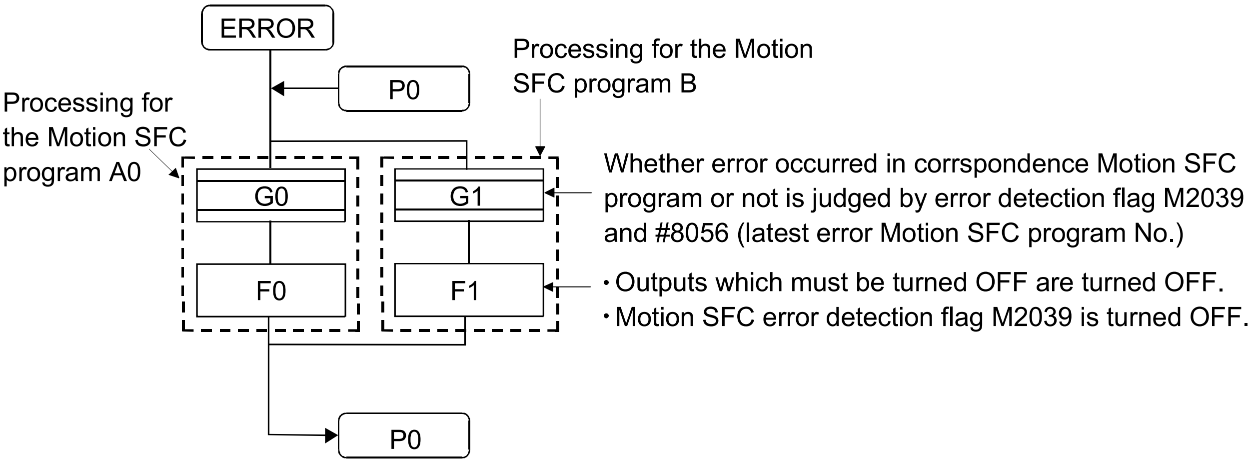
[M2000 ON → OFF]
When PLC ready flag (M2000) turns off, Motion SFC programs stops to execute and the PCPU ready flag (M9074) turns off.
Since actual outputs PY has whole point turn off.

POINT

When the PLC ready flag (M2000) turns off, Motion SFC programs stop but actual outputs PY in the Motion SFC programs do not turn off.

9.13 Operation at The Error Occurrence

Outputs are held if Motion SFC programs stop due to error occurrence.
To turn off outputs at error occurrence, executes the following Motion SFC program.



10. USER FILES

A user file list and directory structure are shown below

10.1 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

- Set the "project name" on the project management screen.

- The "project name" is restricted to 230 characters in length.

- The "project path name" + "project name" are restricted to 230 characters in length.
  ((Example) "C:\Usr\.........\project name\")

10.2 User File List

A user file list is shown below.

(＊) : Indicates the file(data) stored in CPU memory.

Project name folder … Folder of user-set "project name"

Sub folders (fixed)

- Sfc
  - (1) Project file | Project name.prj (×1pc.)
    … Information file of correspondence between Motion SFC program No. (0 to 255) and SFC program names (SFC files)
  - (2) Motion SFC chart file | SFC program name.sfc (×256 pcs.)
    … Motion SFC chart edit information and comment information file of one Motion SFC program
  - (3) Motion SFC list file | SFC program name.txt (×256 pcs.)
    … Text file after conversion of Motion SFC chart of one Motion SFC program into list
  - (4) Motion SFC code file | SFC program name.cod (×256 pcs.)
    … File after conversion of list file of one Motion SFC program into internal codes (including comment information)
- Glist
  - (5) G list file | g0000.bin to g4095.bin
    … LIst file of transition programs (G0 to G4095)
- Gcode
  - (6) G code file | g0000.cod to g4095.cod
    … File after conversion of transition program(G0 to G4095) list file gn.bin(0≦n≦4095) into internal codes
- Flist
  - (7) F/FS list file | f0000.bin to f4095.bin
    … List file of operation control programs (F/FS0 to F/FS4095) list file
- Fcode
  - (8) F/FS code file | f0000.cod to f4095.cod
    … File after conversion of operation control program(F/FS0 to F/FS4095) list file fn.bin(0≦n≦4095) into internal codes
- (＊) (9a) Motion SFC program conversion file(control code) | sfcprog.cod
  … File where SFC code, G-code and F/FS code files are combined and converted into CPU's Motion SFC program code memory storage format
- (＊) (9b) Motion SFC program conversion file(text) | sfcprog.bin
  … File where G list and F/FS list files are combined and converted into CPU's Motion SFC program text memory storage format
  (Note-2) : The above two files are always updated simultaneously.
- (＊) (10) Motion SFC parameter file | sfcprm.bin … Motion SFC control parameter setting information files
- (＊) (11) K code file | svprog.bin … Internal code files of servo program (K0 to K4095) (file size is fixed length)
- (12) Automatic numbering setting information file | autono.inf … Automatic numbering setting information files
- (13) PC type file | gsvp.cnf … CPU type information files
- (＊) (14)
  - System setting data file | svsystemH.bin … System setting data information files
  - High speed read setting file | svlatch.bin … High speed read setting information files
  - Optinal data monitor setting file | svsysmon.bin … Optinal data monitor information files
- (＊) (15) Servo data file
  - svdataH.bin … Axis data (fixed, home position return, JOG operation) parameter block information files
  - svparaH.bin … Servo parameter information files
  - svls.bin … Limit switch setting data information files
- (16) Mechanical system program editing file (Note-1) : 32-axes only
  - svedtda1.bin … Mechanical system program edit information files (page 1 to 8)
  - svedtda2.bin (Note-1) … Mechanical system program edit information files (page 9 to 16)
  - svedtda3.bin (Note-1) … Mechanical system program edit information files (page 17 to 24)
  - svedtda4.bin (Note-1) … Mechanical system program edit information files (page 25 to 32)
- (＊) (17) Mechanical system program conversion file | svmchprm.bin … File after conversion of mechanical system program edit information file svedtdan.bin into internal codes
- (＊) (18) Cam data conversion file (Note-1) : 32-axes only
  - svcamprm.bin … Cam data files of cam No.1 to 64
  - svcampr2.bin (Note-1) … Cam data files of cam No.101 to 164
  - svcampr3.bin (Note-1) … Cam data files of cam No.201 to 264
  - svcampr4.bin (Note-1) … Cam data files of cam No.301 to 364

  (16) to (18): SV22 only
- (19) Backup data file
  - svbackup9.bin … Information file 9 for backup and load
  - svbackup10.bin … Information file 10 for backup and load
- (20)
  - Motion register file | modevice.bin … Motion registers (#0 to #8191) read file
    Only user device range(#0 to #7999) is written.
  - Device memory file | devmem.bin … Devices (X,Y, M/L, B, F, D, W, Special relay, Special register) except # read file
- (21) Device setting screen information file | devset.inf … Device setting information files
- (22) Q series PLC common parameter file | param.wpa … Multiple CPU, I/O assigment data files
- (23) Optional data monitor setting file | svsysmon.bin … Optional data monitor information files
- (24) Communication setting information file | communi.inf … Communication setting information files
- temp … Program editing temporary directory

10.3 Online Change in The Motion SFC Program

The online change is used to write to the Motion SFC program to the internal SRAM during the positioning control (M.RUN LED: ON).
Program correction and a check of operation can be executed repeatedly at the Multiple CPU system start.

Data in which online change is possible are shown below.

Applicable data		Online change	Remarks
System setting data	System setting	×	
	Servo setting data	×	
Motion SFC program	Motion SFC parameter	×	
	Motion SFC chart	○	Online change is possible for the only program during stop.
	Operation control step (F/FS)	○	
	Transition (G)	○	
	Servo program (K)	○	Online change of mode assignment setting is not possible.
Mechanical system program (SV22)		×	
Cam data (SV22)		×	

○ : Possible × : Not possible

POINT

(1) Program writing is executed during the positioning control in the online change. Be safely careful enough for work.
(2) Programs writing to the internal SRAM of Motion CPU at the mode operated by ROM in the online change. If the online change is executed at the mode operated by ROM, it returns to the contents of program written in the internal FLASH ROM by the next power ON or resetting.
(3) If the online change is executed simultaneously to one Motion CPU from the multiple personal computers, a program writing may not be executed. Please do not perform.
(4) If the online changes are executed by other personal computer during the following operation by SW6RN-GSV□P, injustice of a monitor value and operation failure may occur. Please do not perform.
• Monitor mode of the Motion SFC program • Test mode
• Debug mode of the Motion SFC program
(5) If the online change of Motion SFC chart added newly is executed, since the online change of Motion SFC parameter cannot be executed, it operates as the normal task (default value).
(6) When using the SV22, if the online change is executed by changing the "program/servo program editor screen – [Mode assignment setting]", the contents of change are not reflected.
(7) If the cables between the peripheral devices and Motion CPU fall out, or the power supply of the Motion CPU turns OFF or resets, the program is corrupted.
Write the program again with the communication screen of SW6RN- GSV□P.

10.3.1 Operating method for The Online Change

Select the "Online change OFF/ON" of Motion SFC program with the "program editor screen [Convert] menu – [Online change setting]" of SW6RN- GSV☐P.

There are following three methods for the online change of Motion SFC program.

- When the program editor screen [SFC diagram write] is used ---- Online change of the Motion SFC program
- When the operation control/transition program editor screen [Convert] is used ---- Online change of the operation control/transition program editor screen
- When the servo program editor screen [Store] is used ---- Online change of the servo program

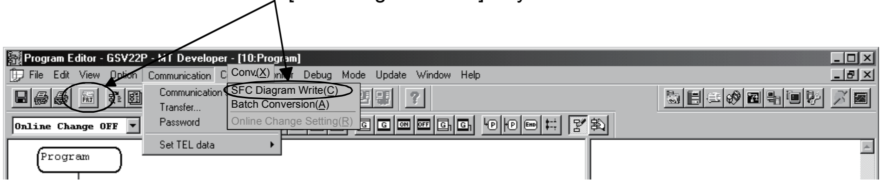
(1) When the program editor screen [SFC diagram write] is used.

Online change of the Motion SFC program during edit is executed by selecting the [SFC diagram write] key.

Online change is possible to the Motion SFC program during stop.

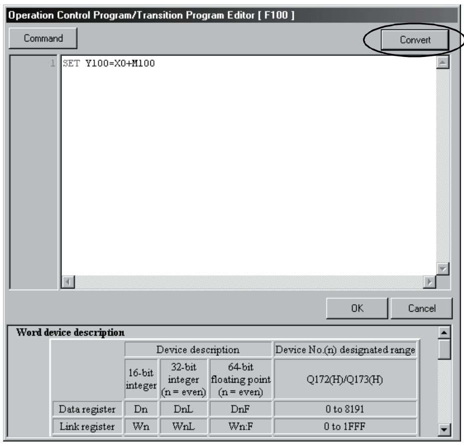
If the online change is made to the program during execution, an alarm message indicates. (Execution/stop state of the Motion SFC program can be checked with the program batch monitor.)

If the start request is made to the program during online change, the Motion SFC start error (error code16007: online change) will occur and the program does not start.



• [SFC Diagram Write] Key

(2) When the operation control/transition program editor screen [Convert] is used.
Online change of the operation control/transition program during edit is executed by selecting the [Convert] key.
Online change is possible to the operation control/transition program during execution.
A program that the online change was made is executed from the next scan.



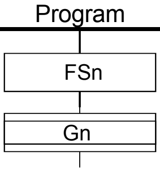
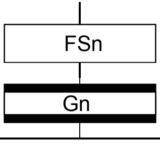
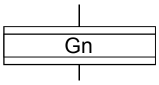
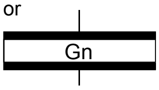
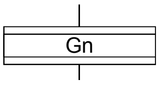
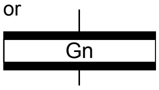
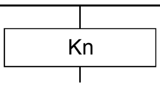
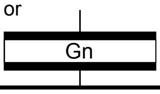
```
SET Y100=X0+M100
```

Word device description

	Device description			Device No.(n) designated range
	16-bit integer	32-bit integer (n = even)	64-bit floating point (n = even)	Q172(H)/Q173(H)
Data register	Dn	DnL	DnF	0 to 8191
Link register	Wn	WnL	Wn:F	0 to 1FFF

• [Convert] Key

Operations for which made the online change to the operation control/transition program during execution in the following conditions are shown below. Be careful to execute the online change in the following conditions.

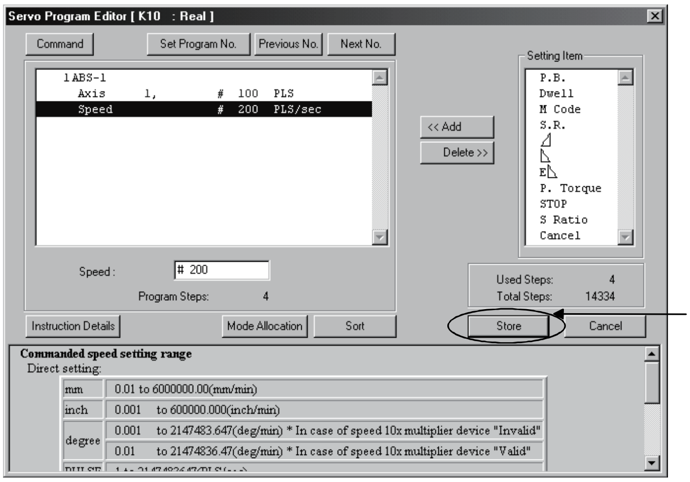
Program	Condition	Operation
 or 	• Online change of the FSn operation control program is executed during FSn execution in the state of waiting for the completion of condition for Gn.	• After completion of online change, the FSn repeats the operation control program that the online change was made until the completion of condition for Gn.
 or 	• Online change of the Gn program is executed in the state of waiting for the completion of condition for Gn. (The conditional sentences of program to write are except the TIME instruction.)	• After completion of online change, the Gn does not transit to the next step until the completion of condition for program that the online change was made.
 or 	• Online change of the Gn program including the TIME instruction is executed in the state of waiting for the completion of condition for Gn.	• After completion of online change, Gn is ended regardless of the waiting time of TIME instruction and the next step is executed.
 or 	• Online change of the Gn program during the servo program execution for Kn.	• After execution of servo program, the program of changed Gn is executed.

(3) When the servo program editor screen [Store] is used.

Online change of the servo program during edit is executed by selecting the [Store] key.

Online change is possible to the servo program during execution.

A program that the online change was made is executed at the next servo program start.



```
1 ABS-1
  Axis      1,        #  100  PLS
  Speed               #  200  PLS/sec
```

Speed : # 200

Program Steps: 4

Used Steps: 4

Total Steps: 14334

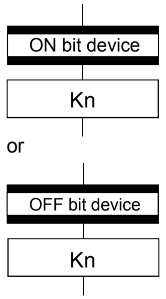
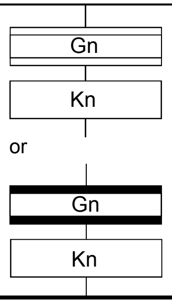
Commanded speed setting range

Direct setting:

mm	0.01 to 6000000.00(mm/min)
inch	0.001 to 600000.000(inch/min)
degree	0.001 to 2147483.647(deg/min) * In case of speed 10x multiplier device "Invalid"
	0.01 to 21474836.47(deg/min) * In case of speed 10x multiplier device "Valid"

• [Store] Key

Operations for which made the online change to the servo program in the following conditions during execution are shown below. Be careful to execute the online change in the following conditions.

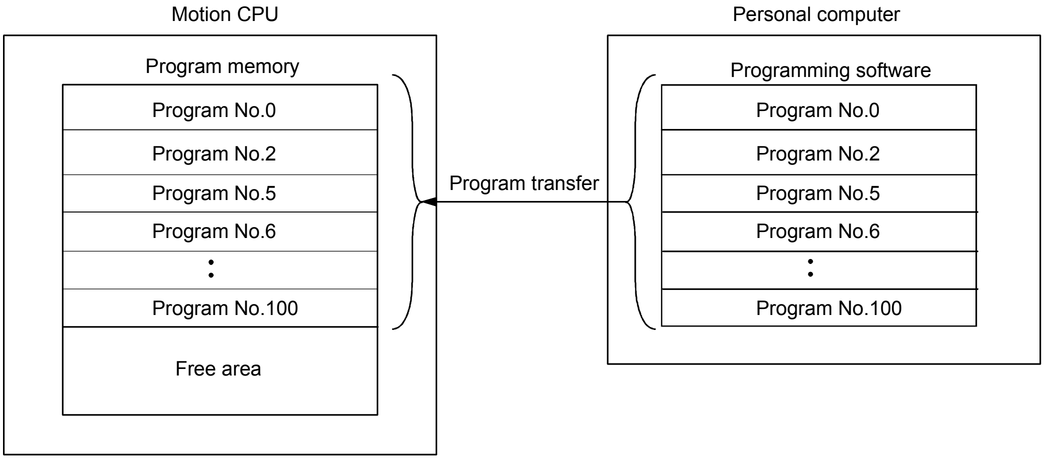
Program	Condition	Operation
	• Online change of the servo program Kn at the WAITON or after WAITOFF is executed in the state of waiting for the completion of condition for WAITON/WAITOFF.	• After completion of condition for WAITON/WAITOFF, the servo program before the online change is started. • The servo program that the online change was made is executed at the next servo program start.
	• Online change of the servo program Kn after Gn is executed in the state of waiting for the completion of condition for Gn.	• After completion of condition for Gn, the servo program that online change was made is executed.

10.3.2 Transfer of program

The outline operations to transfer the program from SW6RN- GSV□P to the program memory of Motion CPU are described.

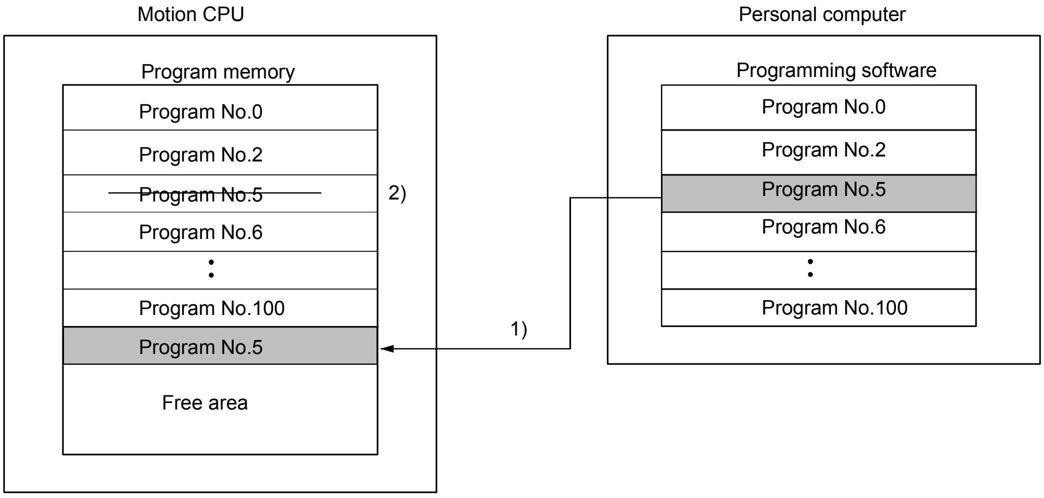
(1) Program writing by the [Communication] menu - [Transfer]

(a) After transfer, programs are stored in the program memory of Motion CPU stuffing to the front for every kind.

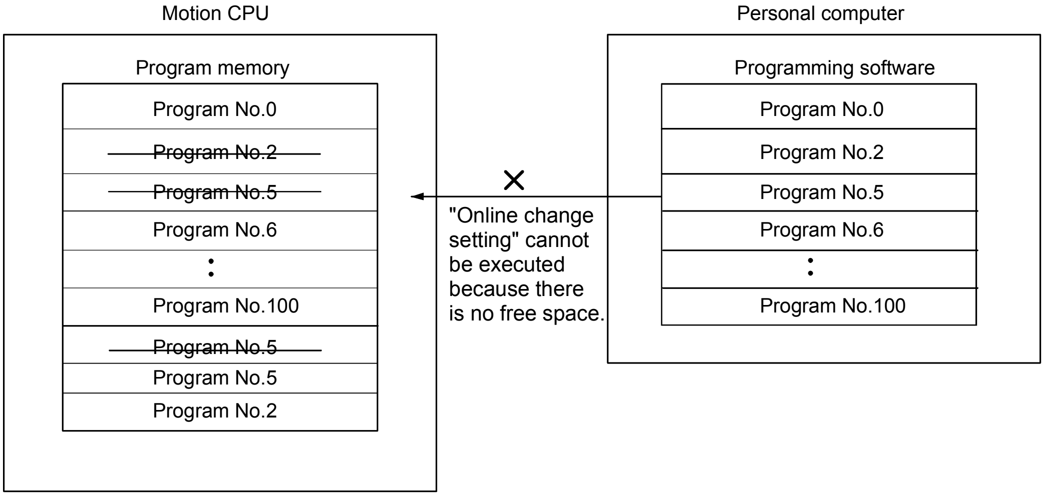


(2) Program writing by the [Online change]

(a) After online change, a program to execute the online change is stored in the free area after the program stored previously. (Refer to 1))
After that, the program written in previously is made invalid and the new program is made valid. (Refer to 2))

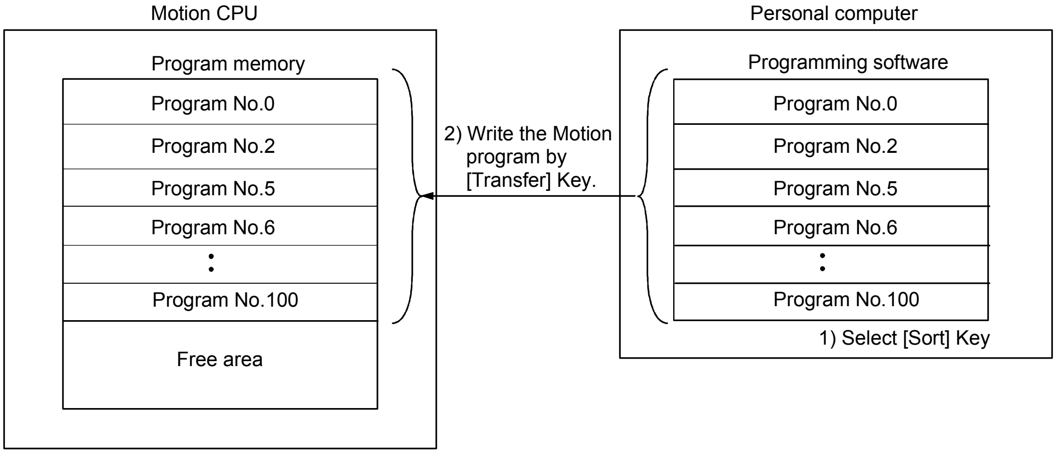


(b) If the online change is executed repeatedly, the free space in program memory is lost and the online change may not be executed. In this case, an error message is displayed by SW6RN-GSV□P at the online change, and "Online change OFF" is set.



(c) In the case of b), arrange to stuff to the front the invalid programs. Operation procedures to stuff to the front are shown below.

1) Select the "program editor screen [Option] menu – [Sort]" of SW6RN-GSV□P. In this case, the invalid programs in the personal computer arranges by SW6RN-GSV□P.
2) Execute the program writing with the [Communication] menu – [Transfer] in the stop state of Motion CPU.



11. ERROR CODE LISTS

When an error occurs while the Motion CPU is running, the error information is stored in the error history devices (#8000 to #8063), special relay M and special register D.

11.1 Reading Procedure for Error Codes

When an error occurs while the Motion SFC program is operating, the error code and error message can be read by the SW6RN-GSV☐P.
The procedure for reading error codes by the SW6RN-GSV☐P is shown below.

(1) Start the SW6RN-GSV☐P.
(2) Connect the Q173HCPU/Q172HCPU to the peripheral devices.
(3) Select [New project] create the project- [Read from Motion CPU] Menu by the SW6RN-GSV☐P, and also read the project from the Motion CPU.
(4) Select the [Monitor] - [Error list] - [Motion SFC error history] and [Error list] Menu.
(5) Display the error code and error message.

Refer to the applicable the help of the SW6RN-GSV☐P for details of the SW6RN-GSV☐P operating method.
The occurrence date of the Motion CPU error history uses a watch function with the internal Motion CPU.
Make the set of the clock data and the clock data read request (M9028) by user programs.
As for the self-diagnosis error code, confirmation can be done by the PC diagnosis of GX Developer.
Refer to the GX Developer operation manual for the GX Developer operation procedure.

11.2 Motion SFC Error Code List

Eight errors that occurred in the past during the Motion SFC control are stored in the "error history devices (#8000 to #8063)" of the motion registers. (Check by SW6RN-GSV□P.)
The "error codes" for the Motion SFC program are shown below.
Refer to the "Q173HCPU/Q172HCPU Motion controller (SV13/SV22) Programming Manual (REAL MODE)"/ "Q173HCPU/Q172HCPU Motion controller (SV22) Programming Manual (VIRTUAL MODE)" for minor errors, major errors, servo errors and servo program setting errors.

Table 11.1 Motion SFC dedicated devices (#8000 to #8063)

<table>
<tr><th rowspan="2">Device No.</th><th rowspan="2" colspan="2">Signal name</th><th colspan="2">Signal direction</th><th rowspan="2">Refresh cycle</th><th rowspan="2">Fetch cycle</th></tr>
<tr><th>Status</th><th>Command</th></tr>
<tr><td>#8000
to</td><td>Seventh error information in past
(Oldest error information)</td><td rowspan="8">Motion SFC error history
(8 errors)
(64 points)<table><tr><th>No.</th><th colspan="2">Signal name</th></tr><tr><td>+0</td><td colspan="2">Error Motion SFC program No.</td></tr><tr><td>+1</td><td colspan="2">Error type</td></tr><tr><td>+2</td><td colspan="2">Error program No.</td></tr><tr><td>+3</td><td colspan="2">Error block No. / Motion SFC list / Line No. / Axis No.</td></tr><tr><td>+4</td><td colspan="2">Error code</td></tr><tr><td>+5</td><td rowspan="3">Error occurrence time</td><td>Year/ Month</td></tr><tr><td>+6</td><td>Day/ Hour</td></tr><tr><td>+7</td><td>Minute/ Second</td></tr></table></td><td rowspan="8">○</td><td rowspan="8">—</td><td rowspan="8">At error occurrence</td><td rowspan="8">—</td></tr>
<tr><td>#8008
to</td><td>Sixth error information in past</td></tr>
<tr><td>#8016
to</td><td>Fifth error information in past</td></tr>
<tr><td>#8024
to</td><td>Fourth error information in past</td></tr>
<tr><td>#8032
to</td><td>Third error information in past</td></tr>
<tr><td>#8040
to</td><td>Second error information in past</td></tr>
<tr><td>#8048
to</td><td>First error information in past</td></tr>
<tr><td>#8056
to
#8063</td><td>Latest error information</td></tr>
</table>

Table 11.2 Motion SFC program start errors (16000 to 16099)

Error code	Error factor		Error Processing	Corrective Action
	Name	Description		
16000	PLC ready OFF (SFCS)	• At a start by S(P).SFCS instruction, PLC ready flag (M2000) or PCPU ready flag (M9074) is OFF.	The specified Motion SFC program does not start.	Provide ON of the PLC ready flag (M2000) and PCPU ready flag (M9074) as start interlocks.
16001	Motion SFC program No. error (SFCS)	• At a start by S(P).SFCS instruction, the range of 0 to 255 is specified in the Motion SFC program No..		Check the Motion SFC program No., and correct a PLC program.
16002	None Motion SFC program (SFCS)	• At a Motion SFC program start by S(P).SFCS instruction, the specified Motion SFC program does not exist.		Check the Motion SFC program No., and correct a PLC program, or create the non-created Motion SFC program.
16003	Double start error	• At a Motion SFC program start by S(P).SFCS instruction, the same Motion SFC program starts.		Double start should be managed on the user side. Provide the user's starting signal as a start interlock in the PLC program.
16004	PLC ready OFF (GINT)	• S(P).GINT instruction was executed with PLC ready flag (M2000) or PCPU ready flag (M9074) is OFF.	The active step of Motion SFC program executed by "PLC interrupt" is not processed. "Interrupt instruction No. " is set in the error Motion SFC program No..	Provide ON of PLC ready flag (M2000) and PCPU ready flag (M9074) as S(P).GINT execution interlocks.
16005	None Motion SFC program	• At a Motion SFC program start by automatic start setting or GSUB, the specified Motion SFC program does not exist.	The specified Motion SFC program does not start. When it started by GSUB, the start source Motion SFC also stop to execute.	Check the Motion SFC program No., and correct a program, or create the non-created Motion SFC program.
16006	Double start error	• At a Motion SFC program start by automatic start setting or GSUB, the same Motion SFC program is already starting.		Double start should be managed on the user side. Provide the user's starting signal as an interlocks in the transition condition.
16007	Online change	• The Motion SFC program which is rewriting the Motion SFC chart by online change was started.	The specified Motion SFC program does not start.	Start after the completion of online change.

Table 11.3 Motion SFC interpreter detection errors (16100 to 16199)

Error code	Error factor		Error Processing	Corrective Action
	Name	Description		
16100	Motion 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.	Stop to execute the applicable Motion SFC program No.. For the subroutine called program, the call source program also stops to execute.	The Motion SFC program code is corrupted. Turn PLC ready flag (M2000) OFF and write the Motion SFC program again. Or, replace the external battery if it passed over a 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	Motion 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 No.) 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.		

Table 11.3 Motion SFC interpreter detection errors (16100 to 16199) (continued)

Error code	Error factor		Error Processing	Corrective Action
	Name	Description		
16110	GSUB setting error 1	• The self program was called/started by GSUB.	Stop to execute the applicable Motion SFC program No.. For the subroutine called program, the call source program also stops to execute.	GSUB cannot call its own or main program. Correct the Motion 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 executed in the Motion SFC programs executed by the event and NMI tasks.
16120	Simultaneously active step count excess	• The number of simultaneously active steps exceeded 256 during execution.		Number of simultaneously active steps is maximum 256. Re-examine the Motion SFC program.

Table 11.4 Motion SFC program run errors (16200 to 16299)

Error code	Error factor		Error Processing	Corrective Action
	Name	Description		
16200	No specified program (Kn)	• The servo program (Kn) specified with the motion control step does not exist.	Stop to execute the applicable Motion SFC program No.. For the subroutine called program, the call source program also stops to execute.	Create the specified servo program.
16201	No specified program (Fn/FSn)	• The operation control program (Fn/FSn) specified with the operation control step does not exist.		Create the specified operation control program.
16202	No specified program (Gn)	• The program (Gn) specified with the transition does not exist.		Create the specified transition program.
16203	No specified program (Motion SFC)	• The Motion SFC program specified with the clear step does not exist.		Correct the specified Motion SFC program name or create the specified Motion SFC program.
16204	No setting of operation expression/conditional expression	• The program (Gn) specified with the transition does not have a conditional expression setting.		Be sure to set a conditional expression in the last block of the transition program.
16205	Fn/FSn program code error	• Internal code error in the operation control program (Fn/FSn)		The Motion SFC program code is corrupted. Turn PLC ready flag (M2000) OFF and write the Motion SFC program again. Or, replace the external battery if it passed over a life.
16206	Gn program code error	• Internal code error in the transition program (Gn)		
16207	Specified the invalid device	• The invalid device (T, C) in the program is set up.		Correct the program which does set the effective device.

Table 11.5 Operation control/transition execution errors (16300 to 16599)

Error code	Error factor		Error Processing	Corrective Action
	Name	Description		
16301	Event task enable (EI) execution error	• Event task enable was executed at except for the normal task.	The block processing on executing 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 at except for 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 with (D) or (S) is not yet registered to the Motion controller. • The resolution of the cam No. specified with (D) or (S) differs from the number of transferred words specified with (n). • (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. • (S) is a bit device and the device number is not a multiple of 16. • (D) is a bit device and the device number is not a multiple of 16. • PX/PY is set in (S) to (S)+(n-1). • PX/PY is set in (D) to (D)+(n-1).		• 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. • Change (n) so that the block transfer range is within the device range. • Change (n) to a positive number. • When (S) or (D) is a bit device, set the device number to be multiple of 16. • When (S) or (D) is a bit device, do not set PX/PY.
16304	Time to wait (TIME) execution error	• The device No. which indirectly specifies (S) is illegal. • The (S) data is outside the range 0 to 2147483647.		• Correct the program so that the device No. which indirectly specifies (S) is proper. • Correct the program so that the (S) data is within the range of 0 to 2147483647.
16305	Same data block transfer (FMOV) execution error	• (D) to (D)+(n-1) is outside the device range. • (n) is 0 or a negative number. • (S) is a bit device and the device number is not a multiple of 16. • (D) is a bit device and the device number is not a multiple of 16. • PX/PY is set in (S). • PX/PY is set in (D) to (D)+(n-1).		• Change (n) so that the block transfer range is within the device range. • When (S) or (D) is a bit device, set the device number to be multiple of 16. • When (S) or (D) is a bit device, do not set PX/PY.
16308	Speed change request (CHGV) execution error	• The specified axis No. is outside the range.		Correct the program so that the specified axis No. 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 No. 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 No. 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			
16322	Device set (SET) execution error	• The device No. which indirectly specifies (D) is illegal. • (D) is a device which is write-disabled.		• Correct the program so that the device No. 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			
16337	Device input (DIN) execution error	• The device No. which indirectly specifies (D) is illegal.		Correct the program so that the device No. which indirectly specifies (D) is proper.

Table 11.5 Operation control/transition execution errors (16300 to 16599) (continued)

Error code	Error factor		Error Processing	Corrective Action
	Name	Description		
16338	Bit device output (OUT=) execution error	• The device No. which indirectly specifies (D) is illegal.	The block processing on executing is stopped and the next block is executed.	Correct the program so that the device No. which indirectly specifies (D) is proper.
16380	Signed 16-bit integer value conversion (SHORT) execution error	• The (S) data is outside the signed 16-bit integer value range.		Correct the program so that the (S) data is within the signed 16-bit integer value range.
16381	Unsigned 16-bit integer value conversion (USHORT) execution error	• The (S) data is outside the unsigned 16-bit integer value range.		Correct the program so that the (S) data is within the unsigned 16-bit integer value range.
16382	Signed 32-bit integer value conversion (LONG) execution error	• The (S) data is outside the signed 32-bit integer value range.	The block processing in execution is stopped and the next block is executed.	Correct the program so that the (S) data is within the signed 32-bit integer value range.
16383	Unsigned 32-bit integer value conversion (ULONG) execution error	• The (S) data is outside the unsigned 32-bit integer value range.		Correct the program so that the (S) data is within the signed 32-bit integer 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 of -1.0 to 1.0.		Correct the program so that (S) is within the range of -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.

Table 11.5 Operation control/transition execution errors (16300 to 16599) (continued)

Error code	Error factor		Error Processing	Corrective Action
	Name	Description		
16420	Write device data to shared CPU memory of the self CPU (MULTW) execution error	• Number of words (n) to be written is outside the range of 1 to 256. • The shared CPU memory address (D) of self CPU of the writing destination device is outside the range (800H to FFFH) of the shared CPU memory address. • The shared CPU memory address (D) of self CPU of the writing destination device + number of words (n) to be written is outside the range (800H to FFFH) of the shared CPU memory address. • First device No. (S) which writing data are stored + number of words (n) to be written is outside the device range. • MULTW instruction was executed again before MULTW instruction is executed and complete bit device is turned on. • (D1) is a write-disabled device. • (S) is a bit device and the device number is not a multiple of 16. • PX/PY is set in (S) to (S)+(n-1).	The block processing in execution is stopped and the next block is executed.	• Correct the program so that the number of words (n) to be written is within the range of 1 to 256. • Correct the program so that the shared CPU memory address (D) of self CPU of the writing destination is within the range of shared CPU memory address. • Correct the program so that the shared CPU memory address (D) of self CPU of the writing destination + number of words (n) to be written is within the range of shared CPU memory address. • Correct the program so that first device No. (S) which writing data are stored + number of words (n) to be written is within the device range. • Execute MULTW instruction again after the complete bit device of MULTW instruction is turned on. • Correct the program to set a write-enabled device at (D1). • When (S) is a bit device, set the device number to be multiple of 16. • When (S) is a bit device, do not set PX/PY.
16421	Read device data from shared CPU memory of the other CPU (MULTR) execution error	• Number of words (n) to be read is outside the range of 1 to 256. • The shared CPU memory first address (S2) of the data which it will be read is outside the range (000H to FFFH) of the shared CPU memory address. • The shared CPU memory first address (S2) of the data which it will be read + number of words (n) to be read is outside the range (000H to FFFH) of the shared CPU memory address. • First device No. (D) which stores the reading data + number of words (n) to be read is outside the device range. • Except 3E0H/3E1H/3E2H/3E3H is set at (S1). • The self CPU is specified with (S1). • The CPU which reads is resetting. • The errors are detected in the CPU which read. • (D) is a bit device and the device number is not a multiple of 16. • PX/PY is set in (D) to (D)+(n-1).		• Correct the program so that the number of words (n) to be read is within the range of 1 to 256. • Correct the program so that the shared CPU memory first address (S2) of the data which it will be read is within the device range of shared CPUmemory address. • Correct the program so that the shared CPU memory first address (S2) of the data which it will be read + number of words (n) to be read is within the range of shared CPU memory address. • Correct the program so that first device No. (D) which stores the reading data + number of words (n) to be read is within the device range. • Correct the program so that 3E0H/3E1H/3E2H/3E3H is set at (S1). • Correct the program so that the self CPU is not specified with (S1). • Check that the reset flag (M9240 to M9243) is OFF, then correct the program to execute the MULTR instruction. • If the errors are detected in the CPU which read, exchange the CPU. • When (D) is a bit device, set the device number to be multiple of 16. • When (D) is a bit device, do not set PX/PY.

Table 11.5 Operation control/transition execution errors (16300 to 16599) (continued)

Error code	Error factor		Error Processing	Corrective Action
	Name	Description		
16422	Write device data to intelligent function module/special function module (TO) execution error	• Number of words (n) to be written is outside the range of 1 to 256. • Motion CPU cannot communicate with intelligent function module/special function module at the instruction execution. • Abnormalities of the intelligent function module/special function module were detected at the instruction execution. • I/O No.s specified with (D1) differ from the intelligent function module/special function module controlled by the self CPU. • The address specified with (D2) is outside the buffer memory range. • First device No. (S) which writing data are stored + number of words (n) to be written is outside the device range. • (S) is a bit device and the device number is not a multiple of 16. • PX/PY is set in (S) to (S)+(n-1).	The block processing in execution is stopped and the next block is executed.	• Correct the program so that the number of words (n) to be written is within the range of 1 to 256. • Replace the intelligent function module/ special function module if there is a fault. • Correct the program so that the first I/O No.s specified with (D1) is intelligent function module/special function module controlled by the self CPU. • Correct the program so that the address specified with (D2) is within the buffer memory range. • Correct the program so that first device No. (S) which writing data are stored + number of words (n) to be written is within the device range. • When (S) is a bit device, set the device number to be multiple of 16. • When (S) is a bit device, do not set PX/PY.
16423	Read device data from intelligent function module/ special function module (FROM) execution error	• Number of words (n) to be read is outside the range of 1 to 256. • Motion CPU cannot communicate with intelligent function module/special function module at the instruction execution. • Abnormalities of the intelligent function module/special function module were detected at the instruction execution. • I/O No.s specified with (S1) differ from the intelligent function module/special function module controlled by the self CPU. • The address specified with (S2) is outside the range buffer memory. • First device No. (D) which stores the reading data + number of words (n) to be read is outside the device range. • (D) is a bit device and the device number is not a multiple of 16. • PX/PY is set in (D) to (D)+(n-1).		• Correct the program so that the number of words (n) to be read is within the range of 1 to 256. • Replace the intelligent function module/ special function module if there is a fault. • Correct the program so that I/O No.s specified with (S1) is intelligent function module/special function module controlled by the self CPU. • Correct the program so that the address specified with (S2) is within the buffer memory range. • Correct the program so that first device No. (D) which stores the reading data + number of words (n) to be read is within the device range. • When (D) is a bit device, set the device number to be multiple of 16. • When (D) is a bit device, do not set PX/PY.

Table 11.5 Operation control/transition execution errors (16300 to 16599) (continued)

Error code	Error factor		Error Processing	Corrective Action
	Name	Description		
16462	Indirectly specified 16-bit motion device (#(n)) read error	• The indirectly specified device No. 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 No. is proper.
16463	Indirectly specified 32-bit motion device (#(n)L) read error	• The indirectly specified device No. 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 No. is outside the range.		
16466	Indirectly specified 32-bit data register (D(n)L) read error	• The indirectly specified device No. is outside the range or an odd number.		
16467	Indirectly specified 64-bit data register (D(n)F) read error			
16468	Indirectly specified 16-bit link register (W(n)) read error	• The indirectly specified device No. is outside the range.		
16469	Indirectly specified 32-bit link register (W(n)L) read error	• The indirectly specified device No. is outside the range or an odd number.		
16470	Indirectly specified 64-bit link register (W(n)F) read error			
16486	Indirectly specified input relay (X(n)) read error	• The indirectly specified device No. is outside the range.		
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			

Table 11.5 Operation control/transition execution errors (16300 to 16599) (continued)

Error code	Error factor		Error Processing	Corrective Action
	Name	Description		
16516	Indirectly specified 16-bit batch input relay (X(n)) read error	• The indirectly specified device No. 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 No. is proper.
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	• The indirectly specified device No. is outside the range or is not a multiple of 16.		
16521	Indirectly specified 32-bit batch internal/latch relay (M(n)/L(n)) read error			
16522	Indirectly specified 16-bit batch internal/latch relay (B(n)) read error	• The indirectly specified device No. 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 No. is proper.
16523	Indirectly specified 32-bit batch internal/latch 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			

11.3 Motion SFC Parameter Errors

Motion SFC parameters are checked by SW6RN-GSV□P.

Table 11.6 PLC ready flag (M2000) OFF → ON errors (17000 to 17009)

Error code	Error factor		Error Processing	Corrective Action
	Name	Description		
17000	Normal task consecutive transition count error	• The normal task's consecutive transition count of the Motion 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 flag (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 Motion 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 Motion SFC program started by the NMI task is outside the range 1 to 10.		

Table 11.7 SFC Program start errors (17010 to 17019)

Error code	Error factor		Error Processing	Corrective Action
	Name	Description		
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 flag (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.		

MEMO

APPENDICES

APPENDIX 1 Processing Times

APPENDIX 1.1 Processing time of operation control/Transition instruction

(1) Operation instructions

Processing time of operation instructions

Classifications	Symbol	Instruction	Operation expression	Q173HCPU/Q172HCPU Unit [μs]
Binary operation	=	Substitution	#0=#1	5.85
			D800=D801	6.15
			#0L=#2L	6.70
			D800L=D802L	8.50
			#0F=#4F	7.85
			D800F=D804F	9.40
	+	Addition	#0=#1+#2	7.00
			D800=D801+D802	10.00
			#0L=#2L+#4L	10.25
			D800L=D802L+D804L	11.75
			#0F=#4F+#8F	10.35
			D800F=D804F+D808F	14.00
	-	Subtraction	#0=#1-#2	8.40
			D800=D801-D802	9.00
			#0L=#2L-#4L	9.85
			D800L=D802L-D804L	12.00
			#0F=#4F-#8F	11.10
			D800F=D804F-D808F	13.55
	*	Multiplication	#0=#1*#2	8.55
			D800=D801*D802	9.75
			#0L=#2L*#4L	10.50
			D800L=D802L*D804L	11.65
			#0F=#4F*#8F	9.90
			D800F=D804F*D808F	13.60
	/	Division	#0=#1/#2	9.05
			D800=D801/D802	10.85
			#0L=#2L/#4L	11.80
			D800L=D802L/D804L	13.60
			#0F=#4F/#8F	11.40
			D800F=D804F/D808F	13.85
	%	Remainder	#0=#1%#2	9.15
			D800=D801%D802	11.55
			#0L=#2L%#4L	12.00
			D800L=D802L%D804L	13.45
Bit operation	~	Bit inversion (complement)	#0=~#1	7.10
			D800=~D801	8.00
			#0L=~#2L	8.45
			D800L=~D802L	9.15

Processing time of operation instructions (Continued)

Classifications	Symbol	Instruction	Operation expression	Q173HCPU/Q172HCPU Unit [μs]
Bit operation	&	Bit logical AND	#0=#1&#2	7.70
			D800=D801&D802	10.00
			#0L=#2L&#4L	9.70
			D800L=D802L&D804L	12.95
	|	Bit logical OR	#0=#1|#2	7.75
			D800=D801|D802	9.05
			#0L=#2L|#4L	10.20
			D800L=D802L|D804L	11.10
	^	Bit exclusive OR	#0=#1^#2	7.70
			D800=D801^D802	9.35
			#0L=#2L^#4L	10.10
			D800L=D802L^D804L	11.55
	>>	Bit right shift	#0=#1>>#2	8.45
			D800=D801>>D802	10.05
			#0L=#2L>>#4L	10.40
			D800L=D802L>>D804L	11.25
	<<	Bit left shift	#0=#1<<#2	8.20
			D800=D801<<D802	9.40
			#0L=#2L<<#4L	10.10
			D800L=D802L<<D804L	11.60
Sign	-	Sign inversion (complement of 2)	#0=-#1	7.15
			D800=-D812	8.00
			#0L=-#2L	9.20
			D800L=-D802L	9.15
			#0F=-#4F	7.95
			D800F=-D804F	10.70
Standard function	SIN	Sine	#0F=SIN(#4F)	17.40
			D800F=SIN(D804F)	18.55
	COS	Cosin	#0F=COS(#4F)	23.05
			D800F=COS(D804F)	25.50
	TAN	Tangent	#0F=TAN(#4F)	27.70
			D800F=TAN(D804F)	30.65
	ASIN	Arcsin	#0F=ASIN(#4F)	31.30
			D800F=ASIN(D804F)	34.60
	ACOS	Arccosin	#0F=ACOS(#4F)	33.10
			D800F=ACOS(D804F)	35.15
	ATAN	Arctangent	#0F=ATAN(#4F)	28.40
			D800F=ATAN(D804F)	31.10
	SQRT	Square root	#0F=SQRT(#4F)	9.05
			D800F=SQRT(D804F)	10.65
	LN	Natural logarithm	#0F=LN(#4F)	14.35
			D800F=LN(D804F)	23.95
	EXP	Exponential operation	#0F=EXP(#4F)	17.80
			D800F=EXP(D804F)	27.05
	ABS	Absolute value	#0F=ABS(#4F)	8.70
			D800F=ABS(D804F)	10.85
	RND	Round-off	#0F=RND(#4F)	11.60
			D800F=RND(D804F)	13.00

Processing time of operation instructions (Continued)

Classifications	Symbol	Instruction	Operation expression	Q173HCPU/Q172HCPU Unit [μs]
Standard function	FIX	Round-down	#0F=FIX(#4F)	11.30
			D800F=FIX(D804F)	13.20
	FUP	Round-up	#0F=FUP(#4F)	11.70
			D800F=FUP(D804F)	13.05
	BIN	BCD→BIN conversion	#0=BIN(#1)	8.40
			D800=BIN(D801)	9.00
			#0L=BIN(#2L)	10.60
			D800L=BIN(D802L)	10.60
	BCD	BIN→BCD conversion	#0=BCD(#1)	12.80
			D800=BCD(D801)	14.25
			#0L=BCD(#2L)	18.10
			D800L=BCD(D802L)	27.05
Type conversion	SHORT	Converted into 16-bit integer type (signed)	#0=SHORT(#2L)	9.75
			#0=SHORT(#4F)	11.35
			D800=SHORT(D802L)	10.60
			D800=SHORT(D804F)	11.85
	USHORT	Converted into 16-bit integer type (unsigned)	#0=USHORT(#2L)	8.85
			#0=USHORT(#4F)	11.95
			D800=USHORT(D802L)	11.40
			D800=USHORT(D804F)	13.90
	LONG	Converted into 32-bit integer type (signed)	#0L=LONG(#2)	7.85
			#0L=LONG(#4F)	12.45
			D800L=LONG(D802)	9.85
			D800L=LONG(D804F)	13.80
	ULONG	Converted into 32-bit integer type (unsigned)	#0L=ULONG(#2)	7.95
			#0L=ULONG(#4F)	12.45
			D800L=ULONG(D802)	10.10
			D800L=ULONG(D804F)	13.95
	FLOAT	Regarded as signed data and converted into 64-bit floating point type	#0F=FLOAT(#4)	7.30
			#0F=FLOAT(#4L)	8.55
			D800F=FLOAT(D804)	9.55
			D800F=FLOAT(D804L)	11.75
	UFLOAT	UFLOAT Regarded as unsigned data and converted into 64-bit floating point type	#0F=UFLOAT(#4)	7.05
			#0F=UFLOAT(#4L)	9.35
			D800F=UFLOAT(D804)	9.90
			D800F=UFLOAT(D804L)	10.75
Bit device status	(None)	ON (normally open contact)	SET M1000 = M0	10.40
			SET M1000 = X100	10.35
			SET M1000 = PX0	18.10
	!	OFF (normally closed contact)	SET M1000 = !M0	11.55
			SET M1000 = !X100	12.00
			SET M1000 = !PX0	16.70
Bit device control	SET	Device set	SET M1000	8.80
			SET Y100	9.30
			SET PY0	14.55
	RST	Device reset	RST M1000	9.30
			RST Y100	10.05
			RST PY0	12.60

Processing time of operation instructions (Continued)

Classifications	Symbol	Instruction	Operation expression	Q173HCPU/Q172HCPU Unit [μs]
Bit device control	DOUT	Device output	DOUT M0,#0	8.60
			DOUT M0,#0L	10.50
			DOUT Y100,#0	9.90
			DOUT Y100,#0L	11.75
			DOUT PY0,#0	14.65
			DOUT PY0,#0L	20.20
	DIN	Device input	DIN #0,M0	8.10
			DIN #0L,M0	8.45
			DIN #0,X0	7.60
			DIN #0L,X0	10.00
			DIN #0,PX0	13.85
			DIN #0L,PX0	18.05
	OUT	Bit device output	OUT M1000 = M0	8.45
			OUT Y0 = M0	10.90
			OUT PY0 = M0	15.70
Logical operation	*	Logical AND	SET M1000 = M0*M1	11.45
			SET M1000 = X100*X101	12.40
			SET M1000 = PX0*PX1	18.10
	+	Logical OR	SET M1000 = M0+M1	11.95
			SET M1000 = X100+X101	12.95
			SET M1000 = PX0+PX1	20.45
Comparison operation	==	Equal to	SET M1000 = #0==#1	13.35
			SET M1000 = D800==D801	13.75
			SET M1000 = #0L==#2L	14.75
			SET M1000 = D800L==D802L	14.00
			SET M1000 = #0F==#4F	14.20
			SET M1000 = D800F==D804F	15.30
	!=	Not equal to	SET M1000 = #0!=#1	13.00
			SET M1000 = D800!=D801	14.25
			SET M1000 = #0L!=#2L	14.30
			SET M1000 = D800L!=D802L	14.80
			SET M1000 = #0F!=#4F	14.00
			SET M1000 = D800F!=D804F	15.50
	<	Less than	SET M1000 = #0<#1	14.00
			SET M1000 = D800<D801	14.50
			SET M1000 = #0L<#2L	14.20
			SET M1000 = D800L<D802L	14.90
			SET M1000 = #0F<#4F	14.50
			SET M1000 = D800F<D804F	15.50
	<=	Less than or equal to	SET M1000 = #0<=#1	14.05
			SET M1000 = D800<=D801	14.00
			SET M1000 = #0L<=#2L	15.20
			SET M1000 = D800L<=D802L	14.85
			SET M1000 = #0F<=#4F	14.95
			SET M1000 = D800F<=D804F	19.90

Processing time of operation instructions (Continued)

Classifications	Symbol	Instruction	Operation expression	Q173HCPU/Q172HCPU Unit [μs]
Comparison operation	>	More than	SET M1000 = #0>#1	14.25
			SET M1000 = D800>D801	22.50
			SET M1000 = #0L>#2L	21.25
			SET M1000 = D800L>D802L	16.70
			SET M1000 = #0F>#4F	14.70
			SET M1000 = D800F>D804F	16.75
	>=	More than or equal to	SET M1000 = #0>=#1	13.00
			SET M1000 = D800>=D801	14.00
			SET M1000 = #0L>=#2L	13.85
			SET M1000 = D800L>=D802L	14.25
			SET M1000 = #0F>=#4F	15.00
			SET M1000 = D800F>=D804F	15.30
Motion dedicated function	CHGV	Speed change request	CHGV(K1,#0)	13.55
			CHGV(K1,D800)	14.20
			CHGV(K1,#0L)	13.85
			CHGV(K1,D800L)	14.05
	CHGT	Torque limit value change request	CHGT(K1,#0)	7.00
			CHGT(K1,D800)	7.80
			CHGT(K1,#0L)	8.45
			CHGT(K1,D800L)	8.45
Others	EI	Event task enable	EI	1.70
	DI	Event task disable	DI	4.45
	NOP	No operation	NOP	2.85
	BMOV	Block transfer	BMOV #0,#100,K10	11.60
			BMOV D800,D100,K10	14.20
			BMOV #0,#100,K100	30.80
			BMOV D800,D100,K100	53.10
			BMOV N1,#0,K512	134.50
			BMOV N1,D800,K512	157.20
	FMOV	Same data block transfer	FMOV #0,#100,K10	10.85
			FMOV D800,D100,K10	12.95
			FMOV #0,#100,K100	16.50
			FMOV D800,D100,K100	39.40
	MULTW	Write device data to shared CPU memory of the self CPU	MULTW H800,#0,K1,M0	15.00
			MULTW H800,D800,K1,M0	14.85
			MULTW H800,#0,K10,M0	16.85
			MULTW H800,D800,K10,M0	17.45
			MULTW H800,D800,#0,K100,M0	39.95
			MULTW H800,D800,K100,M0	44.55
			MULTW H800,#0,K256,M0	66.05
			MULTW H800,D800,K256,M0	77.85

Processing time of operation instructions (Continued)

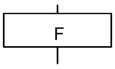
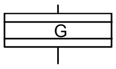
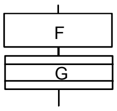
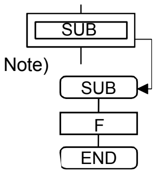
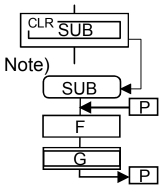
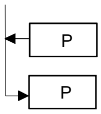
Classifications	Symbol	Instruction	Operation expression	Q173HCPU/Q172HCPU Unit [μs]
Others	MULTR	Read device data from shared CPU memory of the other CPU	MULTR #0,H3E0,H800,K1	34.10
			MULTR D800,H3E0,H800,K1	34.35
			MULTR H800,#0,K10,M0	40.15
			MULTR #0,H3E0,H800,K10	41.35
			MULTR D800,H3E0,H800,K10	126.15
			MULTR #0,H3E0,H800,K100	128.40
			MULTR #0,H3E0,H800,K256	260.50
			MULTR D800,H3E0,H800,K256	261.85
	TO	Write device data to intelligent function module/special function module	TO H0,H0,#0,K1	25.95
			TO H0,H0,D800,K1	26.50
			TO H0,H0,#0,K10	32.60
			TO H0,H0,D800,K10	34.30
			TO H0,H0,#0,K100	108.35
			TO H0,H0,D800,K100	118.70
			TO H0,H0,#0,K256	229.50
			TO H0,H0,D800,K256	257.00
	FROM	Read device data from intelligent function module/special function module	FROM #0,H0,H0,K1	26.50
			FROM D800,H0,H0,K1	29.55
			FROM #0,H0,H0,K10	33.45
			FROM D800,H0,H0,K10	34.00
			FROM #0,H0,H0,K100	116.95
			FROM D800,H0,H0,K100	113.95
			FROM #0,H0,H0,K256	240.70
			FROM D800,H0,H0,K256	244.30
	TIME	Time to wait	TIME K1	9.65
			TIME #0	8.60
			TIME D800	8.85

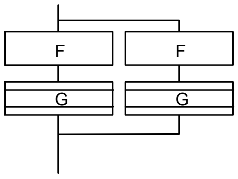
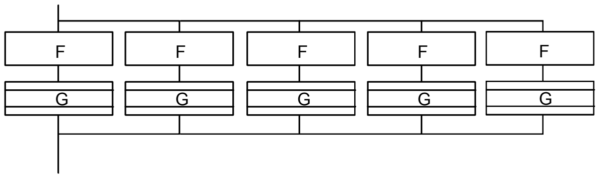
(2) Transition conditional expressions

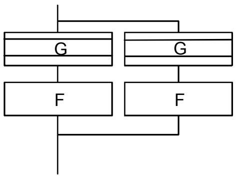
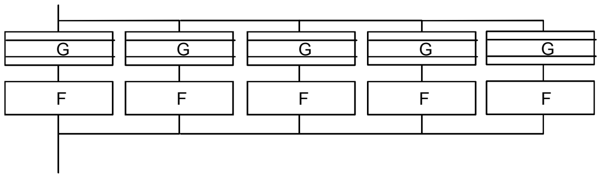
Processing time of transition condition expressions

Classifications	Symbol	Instruction	Operation expression	Q173HCPU/Q172HCPU Unit [μs]
Bit device control	(None)	ON (Normally open contact) (When condition enables)	M0	6.00
			X100	5.40
			PX0	10.40
	!	OFF (Normally closed contact) (When condition enables)	!M0	6.00
			!X100	6.65
			!PX0	9.80
Logical operation	*	Logical AND	M0*M1	7.70
			X100*X101	8.10
			PX0*PX1	14.30
	+	Logical OR	M0+M1	7.85
			X100+X101	7.55
			PX0+PX1	14.25
Comparison operation	==	Equal to	#0==#1	6.15
			D800==D801	6.00
			#0L==#2L	7.40
			D800L==D802L	7.85
			#0F==#4F	6.90
			D800F==D804F	8.55
	!=	Not equal to	#0!=#1	5.35
			D800!=D801	6.55
			#0L!=#2L	7.50
			D800L!=D802L	7.50
			#0F!=#4F	7.45
			D800F!=D804F	8.80
	<	Less than	#0<#1	5.35
			D800<D801	6.85
			#0L<#2L	5.95
			D800L<D802L	6.80
			#0F<#4F	7.35
			D800F<D804F	8.10
	<=	Less than or equal to	#0<=#1	6.50
			D800<=D801	6.60
			#0L<=#2L	7.15
			D800L<=D802L	8.00
			#0F<=#4F	7.25
			D800<=D804F	7.95
	>	More than	#0>#1	5.80
			D800>D801	5.75
			#0L>#2L	7.35
			D800L>D802L	7.75
			#0F>#4F	6.70
			D800F>D804F	8.35
	>=	More than or equal to	#0>=#1	5.10
			D800>=D801	5.95
			#0L>=#2L	7.55
			D800L>=D802L	7.30
			#0F>=#4F	7.70
			D800F>=D804F	7.70

(3) Processing time by the combination F and G (program described in F/G is NOP)

	F alone	G alone	F+G	GSUB	CLR	JMP/coupling
						
Q173HCPU/ Q172HCPU [μs]	28.85	26.10	31.45	81.65	36.15	16.70

	Parallel branch (2 Pcs.)		Parallel branch (5 Pcs.)	
				
	At branch	At coupling	At branch	At coupling
Q173HCPU/ Q172HCPU [μs]	49.30	41.80	98.05	67.40

	Selective branch (2 Pcs.)	Selective branch (5 Pcs.)
		
Q173HCPU/ Q172HCPU [μs]	124.05	176.00

(Note) : Varies greatly with the started or cleared program.

POINT

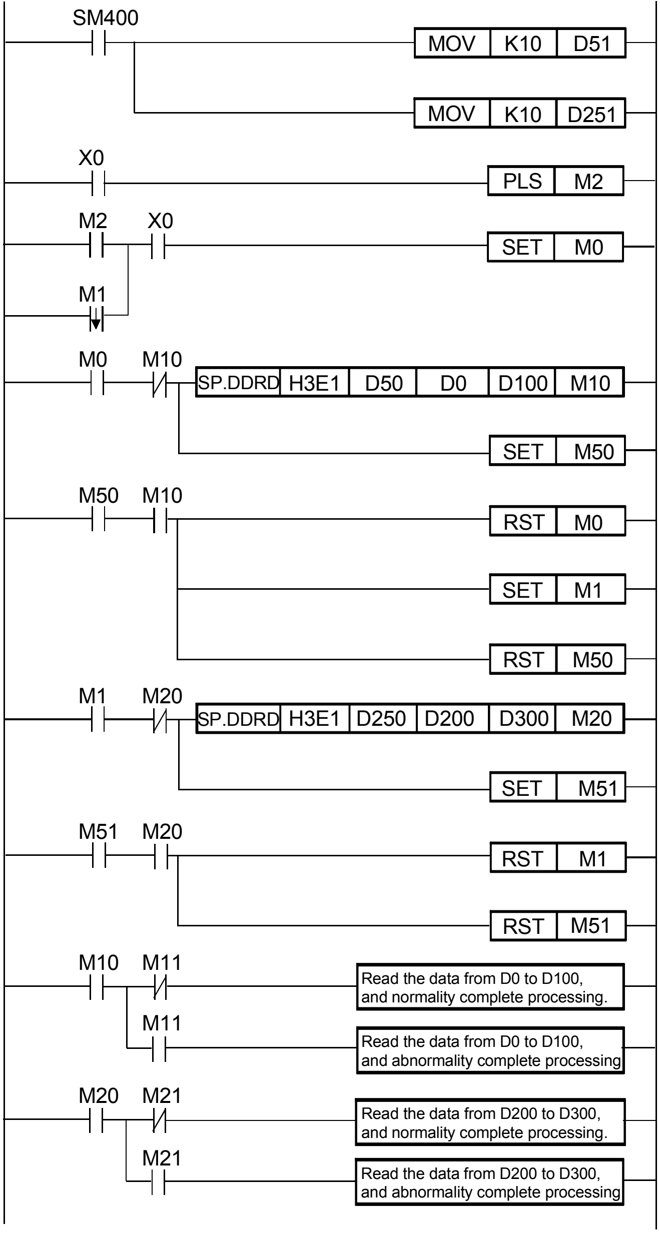
Long processing time may cause a Motion CPU WDT error or servo fault.
Especially for the Motion SFC programs run by event/NMI tasks, take care so that the processing time will not be too long.

APPENDIX 2 Sample Program

APPENDIX 2.1 Program example to execute the Multiple CPU dedicated instruction continuously

This is the program example which publishes the instruction continuously toward the same Motion CPU in the Multiple dedicated instruction toward the Motion CPU. When an instruction cannot be accepted even if it is executed, it becomes "No operation".
The following is program example which repeats reading data for 10 points from D0 of the Motion CPU installing the CPU No.2 to since D100 of the PLC CPU, and the data for 10 points from D200 of the Motion CPU to since D300 of the PLC CPU by turns continuously during X0 is ON. Make a circuit to execute the next S(P).DDRD instruction after the device which it is made to turn on by the instruction completion of the S(P).DDRD instruction execute 1-scan turns it on.

<Example>



There is the following restriction in the case as an example.

1) The Multiple CPU instruction of Motion CPU cannot be used Interrupt program/fixed cycle executive type program and low speed executive type program. When it is used, an instruction may not operate by the timing.

APPENDIX 2.2 The program example to execute plural Multiple CPU instruction by the instructions of one time

This is the program example which executes to the Multiple same Motion CPU at high speed by one instruction.

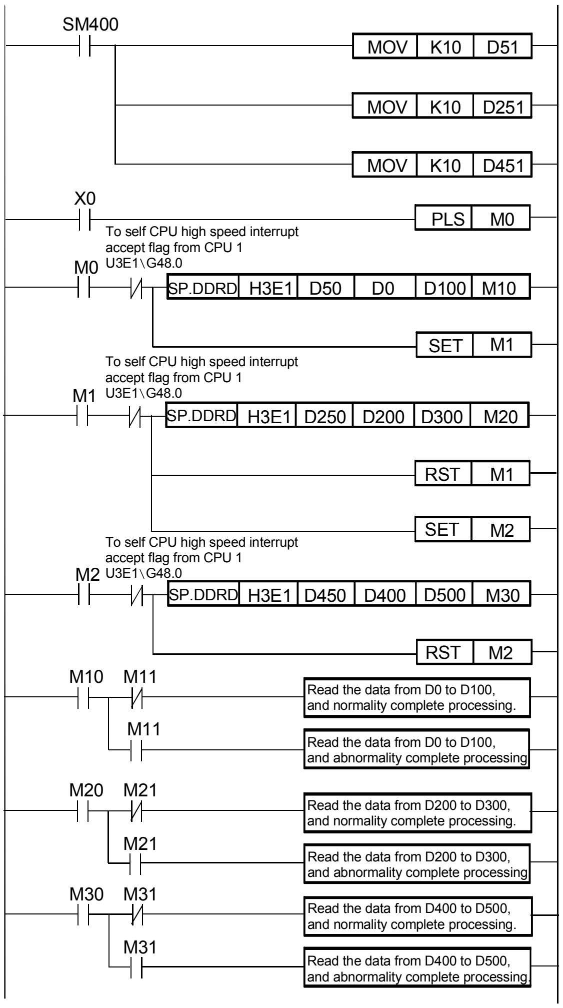
In this case, you must take an interlock with "To self CPU high speed interrupt accept flag from CPU". When an instruction cannot be accepted even if it is executed, it becomes "No operation".

The program which read the data for 10 points from D0 of the Motion CPU installing the CPU No.2 to since D100 of the PLC CPU, the data for 10 points from D200 of the Motion CPU to since D300 of the PLC CPU, and the data for 10 points from D400 of the Motion CPU to since D500 of the PLC CPU by starting of X0 is shown as an example 1.

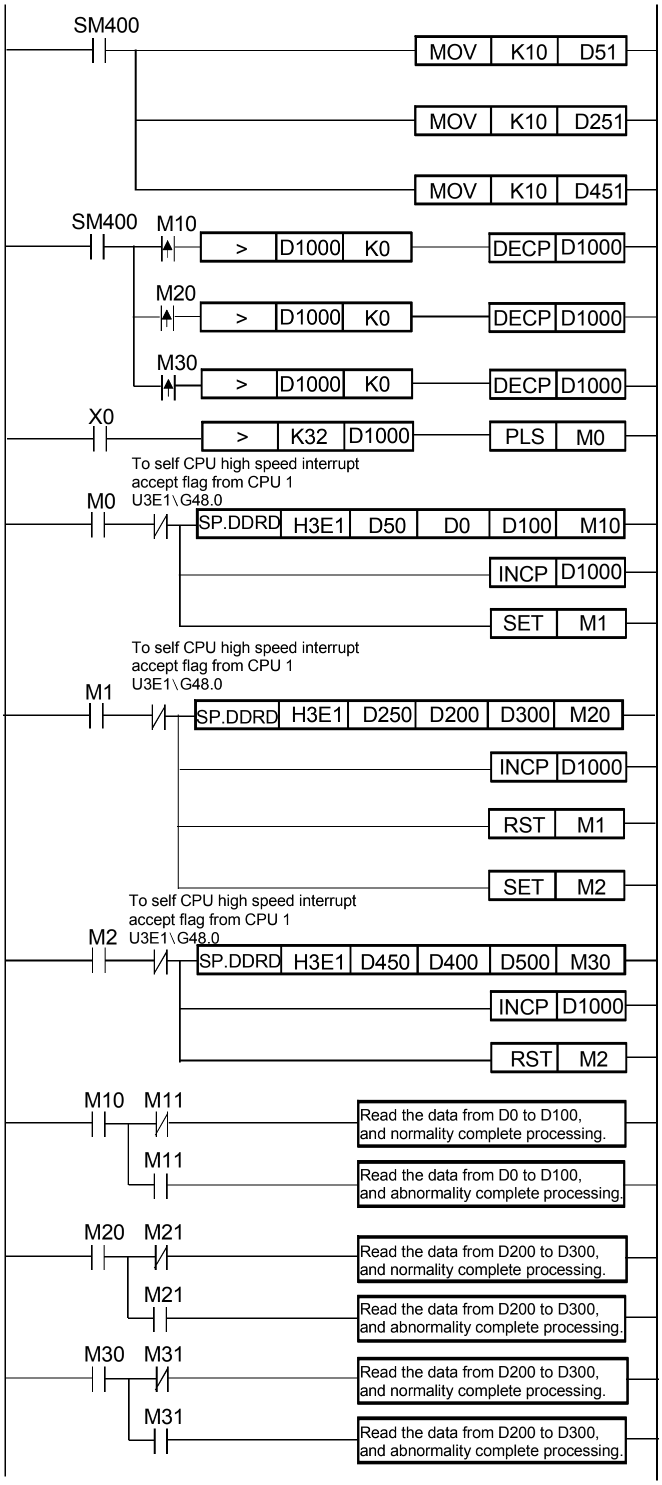
At this time, number of multiple CPU dedicated execute instructions at one command should no exceed the maximum acceptable number of instructions (Refer to Chapter 3.) of one Motion CPU.

When an maximum acceptable number of instructions is 32, the program which made not to execute the multiple dedicated instructions when number of the Multiple CPU dedicated execute instructions exceeds 32 is shown as an example 2.

<Example 1>



<Example 2>



There is the following restriction in the case as the example 2.

1) The Multiple CPU dedicated instruction of Motion CPU cannot be used Interrupt program/fixed cycle executive type program and low speed executive type program. When it is used, an instruction may not work by the timing.

APPENDIX 2.3 Motion control example by Motion SFC program

(1) The Motion SFC program composition example to execute motion control.

This sample program example is described to every following function.

Function list of sample program

No.	Item	Description
1	Monitor of the positioning dedicated device	The positioning dedicated device status of the Motion CPU (CPU No.2) is reflected on "M2400 to" and "D0 to" of the PLC CPU (CPU No.1).
2	Reading of the clock data	The clock data read request (M9028) is turned on so that clock data may be set to the error history.
3	Forced stop	When the forced stop input assigned to PX0 is on, all axes turn on, and motion control is executed. When the forced stop input turn off, servo amplifier is made to forced stop, and motion control is suspended, and actual output (PY) turn off.
4	Motion control	Motion control is executed according to the condition of PX and PX2 in each following mode. • PX2 : OFF PX1 : OFF JOG mode • PX2 : OFF PX1 : ON Manual pulse generator mode • PX2 : On PX1 : OFF Home position return mode • PX2 : On PX1 : On Programming operation mode
5	JOG mode	The following JOG operation is executed when each signal of PX3 to PX6 is turned on. • PX3 : 1 axis JOG forward rotation • PX4 : 1 axis JOG reverse rotation • PX5 : 2 axes JOG forward rotation • PX6 : 2 axes JOG reverse rotation
6	Manual pulse generator mode	The following the manual pulse generator operation is executed. • Manual pulse generator operation of 1 axis is executed with the manual pulse generator P1. • Manual pulse generator operation of 2 axes is executed with the manual pulse generator P1.
7	Home position return mode	The following home position return is executed. • When PX3 is on, the home position return of 1 axis is executed. • When PX4 is on, the home position return of 2 axes is executed.
8	Programming operation mode	The following program operation is executed. • When PX3 detects OFF to ON, axis No.1 locates and 1000[ms] standing by, after the location of axis No.2 is executed. • When PX4 turn on, axis No.1, 2 locates of the linear control and in-position check is executed, after the location of axis No.2 is executed, the program stands by until No.1, 2 locates of the linear control is executed at a double speed in the opposition direction and PX4 turns off.

(2) Contents processing of the Motion SFC program

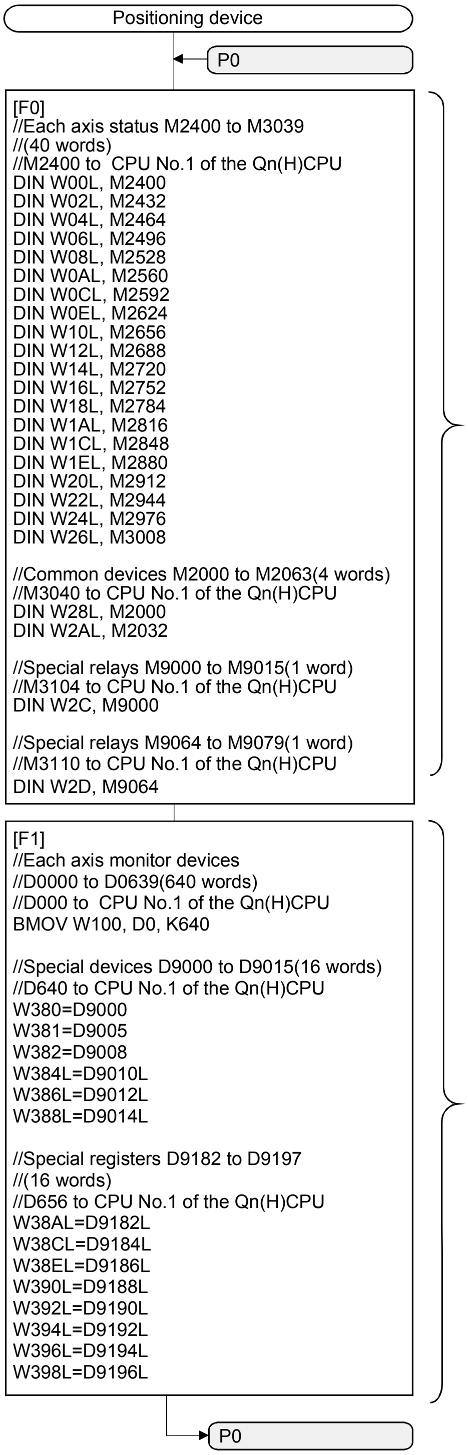
Motion SFC program list

No.	Program name	Task	Automatic operation	Number of connective transitions	Contents of processing
0	Positioning device	Normal	Start	3	• This program starts automatically at the time of run of Q173HCPU, and it is always executed. • The positioning dedicated device (bit device) for monitor is transferred to "W0 to". • The positioning dedicated device (word device) for monitor is transferred to "W100 to". (Note) : "W0 to" is assigned to "M2400 to" of the PLC CPU (CPU No.1), and "W100 to" is assigned to "D0 to" by the automatic refresh setting.
20	Main	Normal	Start	3	• This program starts automatically at the time of run of Q173HCPU, and it is always executed. • Watch data is taken out, and clock data read request (M9028) is turned on. • When a forced stop is canceled, a subroutine starts a "No.110 : Motion control". • "No.110 : Motion control" is stopped at the time of the forced stop, and real output (PY) is turned off.
110	Motion control	Normal	Not start	3	• All axes servo on. • The call of the subroutine of the following program is executed by the condition of PX1, PX2. 1) PX2 : OFF PX1 : OFF No.120 : JOG 2) PX2 : OFF PX1 : ON No.130 : Manual pulse generator 3) PX2 : ON PX1 : OFF No.140 : Home position return 4) PX2 : ON PX1 : ON No.150 : Programming operation
120	JOG	Normal	Not start	3	(1) The JOG operation speed of 1 axis and 2 axes is set. (2) 1 axis JOG forward command is turned on when PX3 is on, and the reverse command is turned on when PX4 is on. (3) 2 axes JOG forward command is turned on when PX5 is on, and the reverse command is turned on when PX6 is on. (4) The above (2), (3) are repeated during PX2/PX1 is off, when except for it, the JOG forward and reverse command of 1 axis and 2 axes are turned off and the program is ended.
130	Manual pulse generator	Normal	Not start	3	• 1 pulse input magnification of the 1 axis and 2 axes is set up. • 1 axis is controlled with P1, and set up to control 2 axes with P2, and Manual pulse generator enable flag of P1, P2 is turned on. • When except for PX2 : OFF, PX1 : ON (Manual pulse generator mode), Manual pulse generator enable flag of P1, P2 is turned off, and a program is ended.

Motion SFC program list (Continued)

No.	Program name	Task	Automatic operation	Number of connective transitions	Contents of processing
140	Home position return	Normal	Not start	3	• "K140 : The home position return of 1 axis" is started when PX3 is on,"K141 : The home position return of 2 axes" is started when PX4 is on. • PX2 : ON, PX1 : The program is ended when they become to except for off (Home position return mode).
150	Programming operation	Normal	Not start	3	• When PX3 detects OFF to ON, after positioning of 1 axis, standing by for 1000[ms] and positioning of 2 axes is executed. • When PX4 turn on, after positioning of linear interpolation in-position check is executed, positioning of axis No. 1, 2 linear interpolation is executed at a double speed in the opposition direction, and it stand by until PX4 turned off. • PX2 : ON, PX1 : The program is fended when they become to except for ON (Programming operation mode).

(a) No.0 : Positioning device



```
[F0]
//Each axis status M2400 to M3039
//(40 words)
//M2400 to  CPU No.1 of the Qn(H)CPU
DIN W00L, M2400
DIN W02L, M2432
DIN W04L, M2464
DIN W06L, M2496
DIN W08L, M2528
DIN W0AL, M2560
DIN W0CL, M2592
DIN W0EL, M2624
DIN W10L, M2656
DIN W12L, M2688
DIN W14L, M2720
DIN W16L, M2752
DIN W18L, M2784
DIN W1AL, M2816
DIN W1CL, M2848
DIN W1EL, M2880
DIN W20L, M2912
DIN W22L, M2944
DIN W24L, M2976
DIN W26L, M3008

//Common devices M2000 to M2063(4 words)
//M3040 to CPU No.1 of the Qn(H)CPU
DIN W28L, M2000
DIN W2AL, M2032

//Special relays M9000 to M9015(1 word)
//M3104 to CPU No.1 of the Qn(H)CPU
DIN W2C, M9000

//Special relays M9064 to M9079(1 word)
//M3110 to CPU No.1 of the Qn(H)CPU
DIN W2D, M9064
```

(1) Each axis status M2400 to M3039 (for 32 axes) transferred to "W0 to".
(2) Common devices M2000 to M2064 transferred to "W28 to".
(3) Special relay M9000 to M9015 transferred to "W2C to".

Automatic refresh of the between Multiple CPU, and "W0 to" of Q173HCPU (CPU No.2) sets it up to have refresh by "M2400 to" of Qn(H)CPU (CPU No.1), therefore the condition of Q173HCPU (CPU No.2) can be grasped with Qn(H)CPU of the CPU No.1 by monitoring the following device.

Devices of QnHCPU (CPU No.1)	Correspond with devices of Q173HCPU (CPU No.2)
M2400 to M3039	M2400 to M3039
M3040 to M3103	M2000 to M2064
M3104 to M3119	M9000 to M9015

(Note) : Refresh does data for 32 axes by this sample example, number of refresh points is made a necessary minimum corresponding to the system for processing time shortening.

```
[F1]
//Each axis monitor devices
//D0000 to D0639(640 words)
//D000 to  CPU No.1 of the Qn(H)CPU
BMOV W100, D0, K640

//Special devices D9000 to D9015(16 words)
//D640 to CPU No.1 of the Qn(H)CPU
W380=D9000
W381=D9005
W382=D9008
W384L=D9010L
W386L=D9012L
W388L=D9014L

//Special registers D9182 to D9197
//(16 words)
//D656 to CPU No.1 of the Qn(H)CPU
W38AL=D9182L
W38CL=D9184L
W38EL=D9186L
W390L=D9188L
W392L=D9190L
W394L=D9192L
W396L=D9194L
W398L=D9196L
```

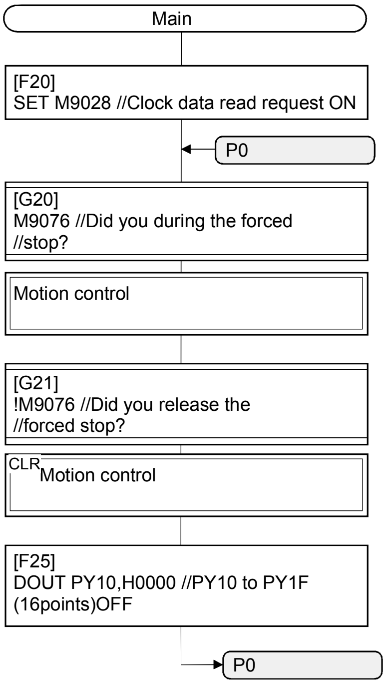
(1) Each monitor devices D0 to D639 (for 32 axes) transferred to "W100 to".
(2) Special register D9000 to D9015 transferred to "W380 to".
(3) Special register D9182 to M9197 transferred to "W38A to".

Automatic refresh of the between Multiple CPU, and "W100 to" of Q173HCPU (CPU No.2) sets it up to have refresh by "D0 to" of Qn(H)CPU (CPU No.1), therefore the condition of Q173HCPU (CPU No.2) can be grasped with Qn(H) CPU of the CPU No.1 by monitoring the following device.

Devices of QnHCPU (CPU No.1)	Correspond with devices of Q173HCPU (CPU No.2)
D0000 to D0639	D0000 to D0630
D0640 to D0655	D9000 to D9015
D0656 to D0671	D9182 to D9197

(Note) : Refresh does data for 32 axes by this sample example, number of refresh points is made a necessary minimum corresponding to the system for processing time shortening.

(b) No.20 : Main



When a forced stop is released, a subroutine starts "No.110 : Motion control". (Because the next step is a shift, it becomes a subroutine start, and the next step is executed at the same time with subroutine practice, too.)

"No.110 : Motion control" is made to stop at the time of the forced stop, and (The program that a subroutine call is executed from No.110 stops, too.) actual output (PY) is turned off.

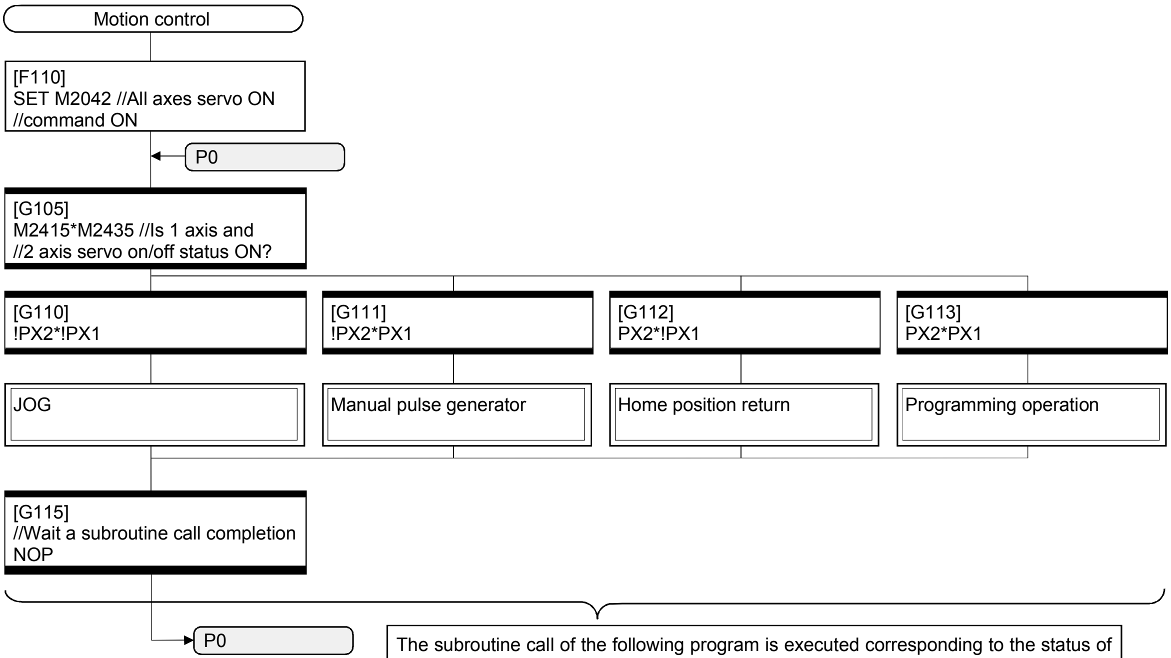
(Note-1) : The program that a subroutine was started is made to stop if necessary when a subroutine start program is added because it does not stop.

(Note-2) : Real output is turned off if necessary.

(Note-3) : The occurrence detection of servo error and so on is added to the stop status with forced stop if necessary.

When a forced stop is released, it is the structure which starts the program which does motion control from the initials again by sample program. Therefore it is the system example that motion control is resumed when a forced stop release is executed after it stops forced for while.

(c) No.110 : Motion control



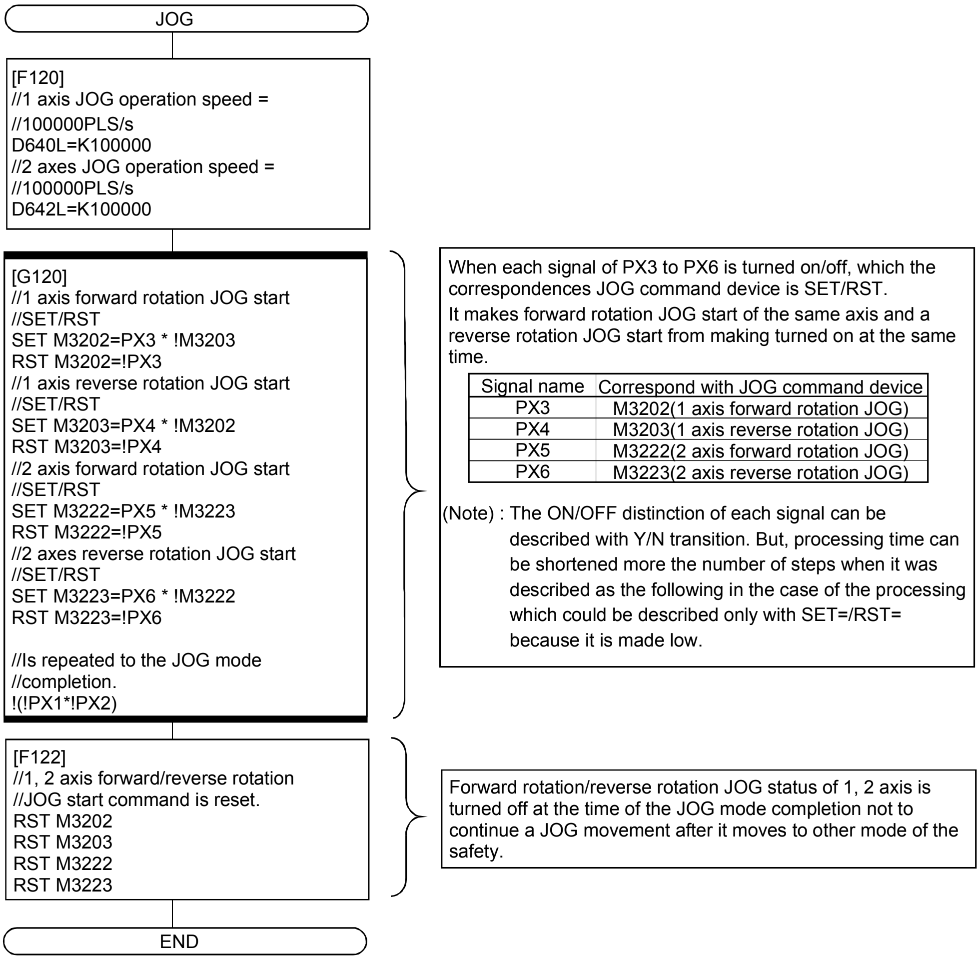
The subroutine call of the following program is executed corresponding to the status of PX1, PX2.

Selective branch is made the program so that each subroutine may never start it in the plural at the same time.

And, each subroutine makes the next step "WAIT" to become a subroutine call to make it stop when this program is suspended by the clear step of "No.20 : Main", too.

Condition of PX1, PX2		Subroutine call program	
PX2	PX1	No.	Program name
OFF	OFF	120	JOG
OFF	ON	130	Manual pulse generator
ON	OFF	140	Home position return
ON	ON	150	Programming operation

(d) No.120 : JOG



```
[F120]
//1 axis JOG operation speed =
//100000PLS/s
D640L=K100000
//2 axes JOG operation speed =
//100000PLS/s
D642L=K100000
```

```
[G120]
//1 axis forward rotation JOG start
//SET/RST
SET M3202=PX3 * !M3203
RST M3202=!PX3
//1 axis reverse rotation JOG start
//SET/RST
SET M3203=PX4 * !M3202
RST M3203=!PX4
//2 axis forward rotation JOG start
//SET/RST
SET M3222=PX5 * !M3223
RST M3222=!PX5
//2 axes reverse rotation JOG start
//SET/RST
SET M3223=PX6 * !M3222
RST M3223=!PX6

//Is repeated to the JOG mode
//completion.
!(!PX1*!PX2)
```

When each signal of PX3 to PX6 is turned on/off, which the correspondences JOG command device is SET/RST.
It makes forward rotation JOG start of the same axis and a reverse rotation JOG start from making turned on at the same time.

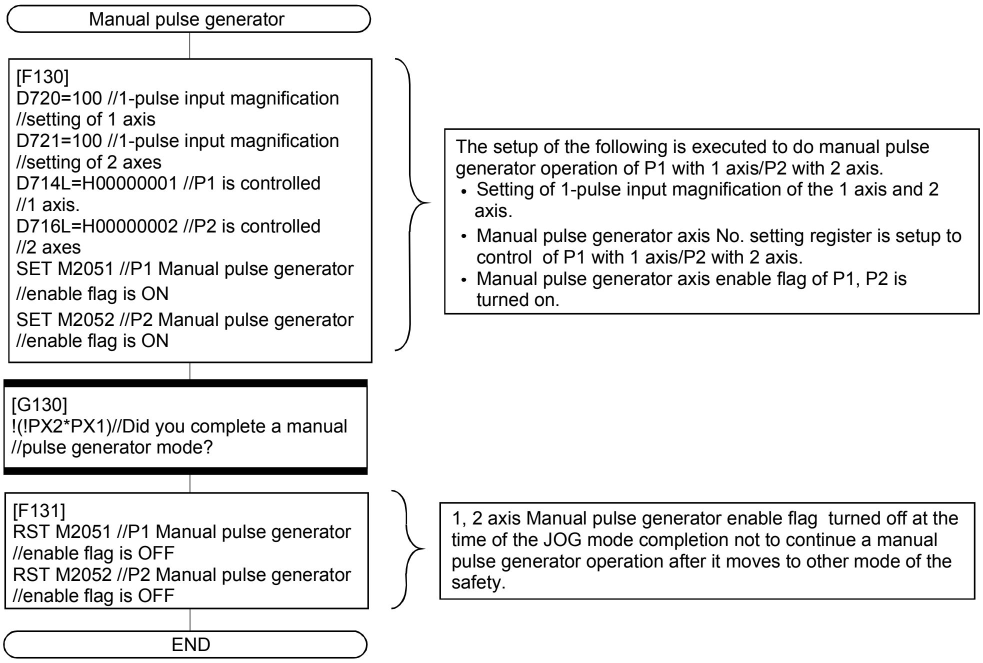
Signal name	Correspond with JOG command device
PX3	M3202(1 axis forward rotation JOG)
PX4	M3203(1 axis reverse rotation JOG)
PX5	M3222(2 axis forward rotation JOG)
PX6	M3223(2 axis reverse rotation JOG)

(Note) : The ON/OFF distinction of each signal can be described with Y/N transition. But, processing time can be shortened more the number of steps when it was described as the following in the case of the processing which could be described only with SET=/RST= because it is made low.

```
[F122]
//1, 2 axis forward/reverse rotation
//JOG start command is reset.
RST M3202
RST M3203
RST M3222
RST M3223
```

Forward rotation/reverse rotation JOG status of 1, 2 axis is turned off at the time of the JOG mode completion not to continue a JOG movement after it moves to other mode of the safety.

(e) No.130 : Manual pulse generator



```
[F130]
D720=100 //1-pulse input magnification
//setting of 1 axis
D721=100 //1-pulse input magnification
//setting of 2 axes
D714L=H00000001 //P1 is controlled
//1 axis.
D716L=H00000002 //P2 is controlled
//2 axes
SET M2051 //P1 Manual pulse generator
//enable flag is ON
SET M2052 //P2 Manual pulse generator
//enable flag is ON
```

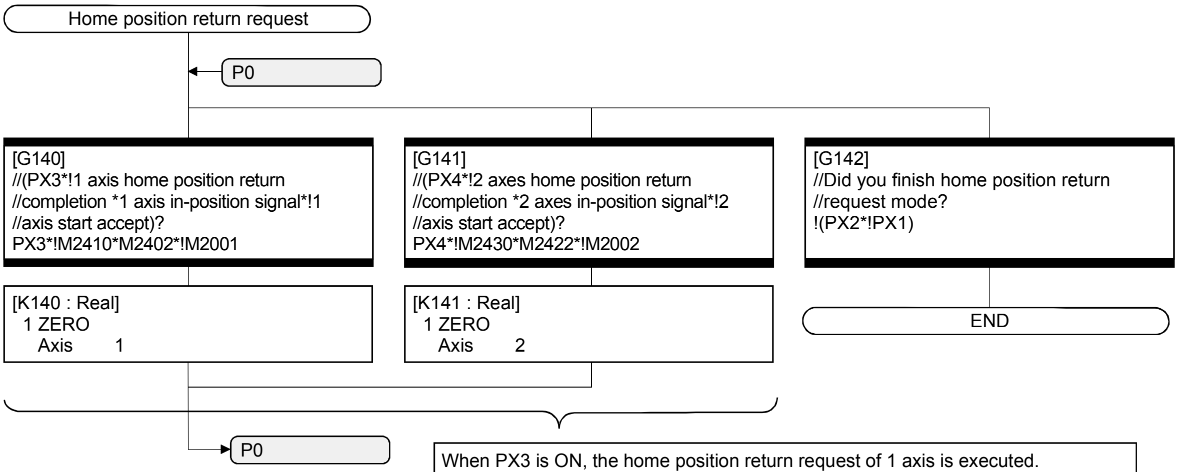
The setup of the following is executed to do manual pulse generator operation of P1 with 1 axis/P2 with 2 axis.
- Setting of 1-pulse input magnification of the 1 axis and 2 axis.
- Manual pulse generator axis No. setting register is setup to control of P1 with 1 axis/P2 with 2 axis.
- Manual pulse generator axis enable flag of P1, P2 is turned on.

```
[G130]
!(!PX2*PX1)//Did you complete a manual
//pulse generator mode?
```

```
[F131]
RST M2051 //P1 Manual pulse generator
//enable flag is OFF
RST M2052 //P2 Manual pulse generator
//enable flag is OFF
```

1, 2 axis Manual pulse generator enable flag turned off at the time of the JOG mode completion not to continue a manual pulse generator operation after it moves to other mode of the safety.

(f) No.140 : Home position return

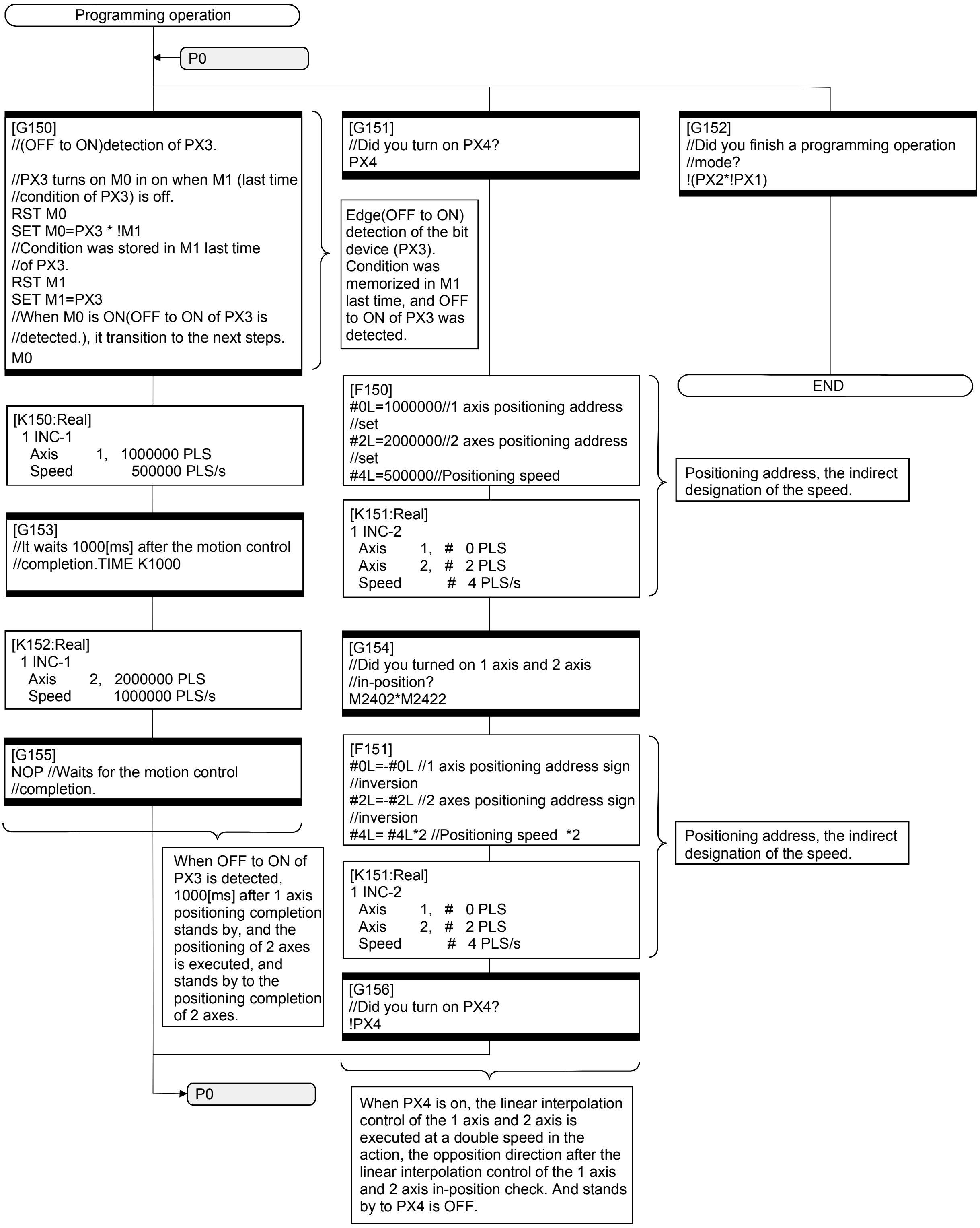


```
[G140]
//(PX3*!1 axis home position return
//completion *1 axis in-position signal*!1
//axis start accept)?
PX3*!M2410*M2402*!M2001

[K140 : Real]
  1 ZERO
    Axis       1

[G141]
//(PX4*!2 axes home position return
//completion *2 axes in-position signal*!2
//axis start accept)?
PX4*!M2430*M2422*!M2002

[K141 : Real]
  1 ZERO
    Axis       2

[G142]
//Did you finish home position return
//request mode?
!(PX2*!PX1)
```

When PX3 is ON, the home position return request of 1 axis is executed.
When PX4 is ON, the home position return request of 2 axes is executed.
At this time in-position signal ON and start accept OFF confirmed, and home position return request program is started.

(Note) : This program is the structure which does not have WAIT that it waits for the completion of the home position return in the next of the motion control step, because it possible a thing during "K140" practice and "K141" are started.
(You must take the initial start of each axis to interlock condition to prevent the double start of K140 and K141.)

(g) No.150 : Programming operation



```
[G150]
//(OFF to ON)detection of PX3.

//PX3 turns on M0 in on when M1 (last time
//condition of PX3) is off.
RST M0
SET M0=PX3 * !M1
//Condition was stored in M1 last time
//of PX3.
RST M1
SET M1=PX3
//When M0 is ON(OFF to ON of PX3 is
//detected.), it transition to the next steps.
M0
```

Edge(OFF to ON) detection of the bit device (PX3). Condition was memorized in M1 last time, and OFF to ON of PX3 was detected.

```
[K150:Real]
 1 INC-1
  Axis        1,   1000000 PLS
  Speed             500000 PLS/s
```

```
[G153]
//It waits 1000[ms] after the motion control
//completion.TIME K1000
```

```
[K152:Real]
 1 INC-1
  Axis      2,   2000000 PLS
  Speed          1000000 PLS/s
```

```
[G155]
NOP //Waits for the motion control
//completion.
```

When OFF to ON of PX3 is detected, 1000[ms] after 1 axis positioning completion stands by, and the positioning of 2 axes is executed, and stands by to the positioning completion of 2 axes.

```
[G151]
//Did you turn on PX4?
PX4
```

```
[F150]
#0L=1000000//1 axis positioning address
//set
#2L=2000000//2 axes positioning address
//set
#4L=500000//Positioning speed
```

```
[K151:Real]
1 INC-2
 Axis       1,  #   0 PLS
 Axis       2,  #   2 PLS
 Speed          #   4 PLS/s
```

Positioning address, the indirect designation of the speed.

```
[G154]
//Did you turned on 1 axis and 2 axis
//in-position?
M2402*M2422
```

```
[F151]
#0L=-#0L //1 axis positioning address sign
//inversion
#2L=-#2L //2 axes positioning address sign
//inversion
#4L= #4L*2 //Positioning speed  *2
```

```
[K151:Real]
1 INC-2
 Axis       1,  #   0 PLS
 Axis       2,  #   2 PLS
 Speed          #   4 PLS/s
```

Positioning address, the indirect designation of the speed.

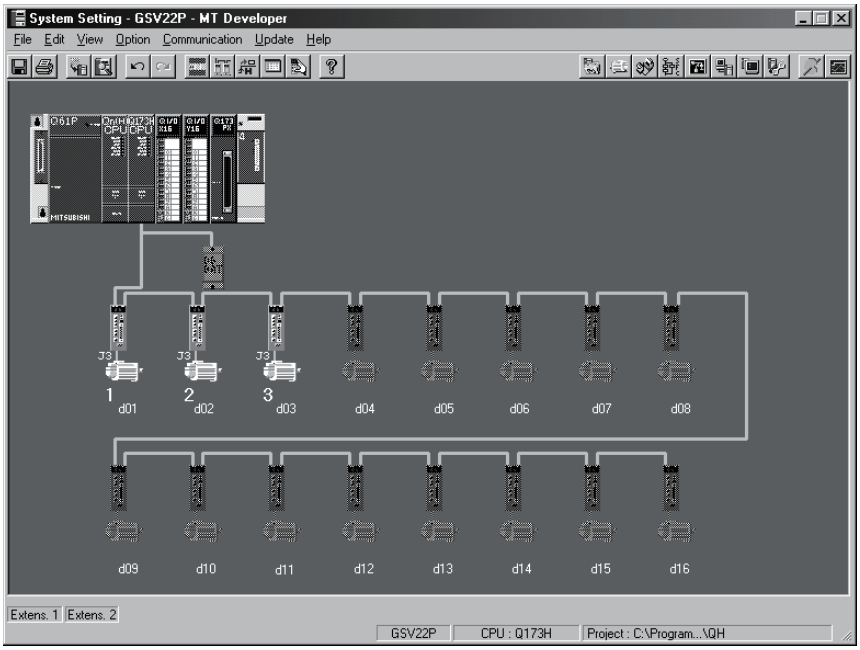
```
[G156]
//Did you turn on PX4?
!PX4
```

When PX4 is on, the linear interpolation control of the 1 axis and 2 axis is executed at a double speed in the action, the opposition direction after the linear interpolation control of the 1 axis and 2 axis in-position check. And stands by to PX4 is OFF.

```
[G152]
//Did you finish a programming operation
//mode?
!(PX2*!PX1)
```

(3) System setting data of the Motion CPU

System setting is shown below.



(a) Module setting

Manual pulse generator interface module (Q173PX : Slot 3)

Axis No.	Description
P1	Manual pulse generator, Synchronous encoder (INC)
P2	Manual pulse generator, Synchronous encoder (INC)
P3	Manual pulse generator, Synchronous encoder (INC)
I/O response time	0.4[ms]

(b) Basic setting

1) Multiple CPU setting

Setting items	Description
Number of Multiple CPU	2 modules
Operating mode	All CPU stop by stop error of CPU No.1/2

2) Automatic refresh setting 1

CPU	Send range for each CPU			CPU side device	
	CPU share memory G			Dev. starting	W0
	Point	Start	End	Start	End
No.1	0	—	—	—	—
No.2	50	0800	0831	W0	W31
No.3					
No.4					

This device area is set up in "M2400" with the Qn(H) CPU No.1. (The bit device for monitor is transferred to "W0 to" by the Motion SFC program on the Q173HCPU side.).

3) Automatic refresh setting 2

CPU	Send range for each CPU			CPU side device	
	CPU share memory G			Dev. starting	W100
	Point	Start	End	Start	End
No.1	0	—	—	—	—
No.2	640	0832	0AB1	W100	W37F
No.3					
No.4					

This device area is set up in "D0" with the Qn (H) CPU No.1. (The ward device for monitor is transferred to "W100" to by the Motion SFC program on the Q173HCPU side.).

4) Automatic refresh setting 3

CPU	Send range for each CPU			CPU side device	
	CPU share memory G			Dev. starting	
	Point	Start	End	Start	End
No.1					
No.2					
No.3					
No.4					

5) Automatic refresh setting 4

CPU	Send range for each CPU			CPU side device	
	CPU share memory G			Dev. starting	
	Point	Start	End	Start	End
No.1					
No.2					
No.3					
No.4					

This setting area is used for the use except for the positioning device for the monitor.

6) System setting

Setting items	Description
Operation cycle setting	Auto
Operation mode	M2000 is turned on with switch (Stop to Run)
Emergency shout down input	PX0

7) Latch range setting

Item	Symbol	Latch (1)		Latch (2)	
		Start	End	Start	End
Internal relay	M				
Link relay	B				
Annunciator	F				
Data register	D				
Link register	W				

Latch (1) : It is possible to clear using the latch clear.

Latch (2) : Clearing using the latch clear is disabled.

(c) PLC module setting

Type of the module	Number of points	Occupation device	Base	Slot No.	I/O response time
Input	16	000-00F	CPU base unit	1	10[ms]
Output	16	010-01F	CPU base unit	2	

(4) Parameter setting of the Qn(H) CPU No.1

<table>
<tr><th colspan="3" rowspan="2">PC parameter item</th><th colspan="5">Qn(H) parameter</th></tr>
<tr><th colspan="5">Description</th></tr>
<tr><td>1</td><td colspan="2">Number of CPU</td><td colspan="5">2 modules</td></tr>
<tr><td rowspan="3">2</td><td colspan="2">Operation mode</td><td colspan="5">The error operating mode in the CPU stop.</td></tr>
<tr><td></td><td>CPU No.1</td><td colspan="5">All station stop by stop error</td></tr>
<tr><td></td><td>CPU No.2</td><td colspan="5">All station stop by stop error</td></tr>
<tr><td rowspan="2">3</td><td colspan="2">Out of group input settings</td><td colspan="5">The input condition outside the group is taken.</td></tr>
<tr><td colspan="2">Out of group output settings</td><td colspan="5">The output condition outside the group is not taken.</td></tr>
<tr><td rowspan="11">4</td><td colspan="7">Refresh setting</td></tr>
<tr><td colspan="2" rowspan="3">Setting No.1
CPU</td><td colspan="3">Send range for each CPU</td><td colspan="2">CPU side device</td></tr>
<tr><td colspan="3">Shared CPU memory G</td><td>First device</td><td>M2400</td></tr>
<tr><td>Point</td><td>Start</td><td>End</td><td>Start</td><td>END</td></tr>
<tr><td></td><td>CPU No.1</td><td>0</td><td>—</td><td>—</td><td>—</td><td>—</td></tr>
<tr><td></td><td>CPU No.2</td><td>50</td><td>0800</td><td>0831</td><td>M2400</td><td>M3199</td></tr>
<tr><td colspan="2" rowspan="3">Setting No.2
CPU</td><td colspan="3">Send range for each CPU</td><td colspan="2">CPU side device</td></tr>
<tr><td colspan="3">Shared CPU memory G</td><td>First device</td><td>D0</td></tr>
<tr><td>Point</td><td>Start</td><td>End</td><td>Start</td><td>END</td></tr>
<tr><td></td><td>CPU No.1</td><td>0</td><td>—</td><td>—</td><td>—</td><td>—</td></tr>
<tr><td></td><td>CPU No.2</td><td>640</td><td>0832</td><td>0AB1</td><td>D0</td><td>D639</td></tr>
</table>

APPENDIX 2.4 Continuation execution example at the subroutine re-start by the Motion SFC program

(1) Explanation of the operation

This is the program example which execute continuously from the motion control step which stopped on the way when it re-started after stopping the subroutine program with the clear step during the motion control is running.

The servo is turned on by the forced stop release and the positioning control of the 2 axes liner interpolation is executed when PX4 is ON in this program. One cycle operation is completed after confirmation that PX4 became OFF. When the forced stop is executed during the positioning operating, the positioning operation is interrupted and the servomotor is stopped. It is resumed from the interrupted positioning operation when the forced stop was released next.
Continuation execution of the subroutine re-start is executed by this program example by the following processing.

(a) While motion control with the subroutine is executed, it is memorized whether the positioning of which motion control step was completed in the user device.

(b) The subroutine re-start is resumed from the motion control step of stopping the information memorized by the above (a).

(c) A motion control step should locate absolute to cope with it when it is resumed after it stops on the way of the positioning.

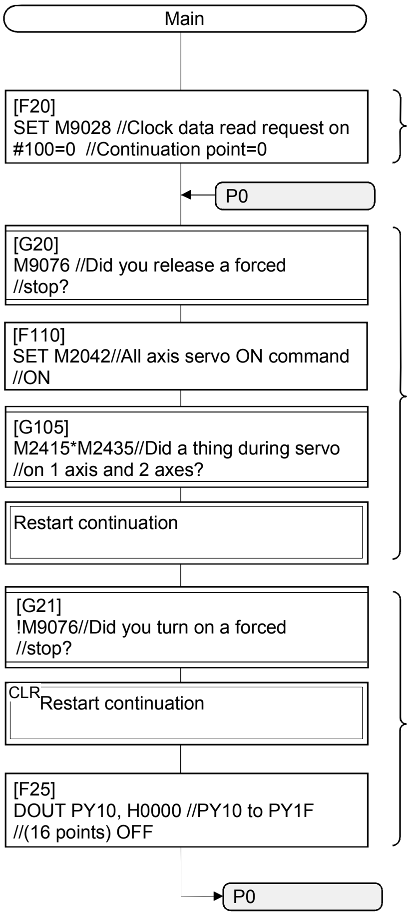
(d) A positioning complete signal (M2401+20n) is used for the decision, whether servomotor is stopped during the positioning.

(2) Contents of processing the Motion SFC program

Motion SFC program list

No.	Program name	Task	Automatic operation	Number of connective transitions	Contents of processing
20	Main	Normal	Start	3	• This program starts automatically at the time of RUN of Q173HCPU, and it is always executed. • Watch data is taken out, and clock data read request (M9028) is turned on. • "0" is set on the continuation point (#100 : user device) as an initial value. • The subroutine starts a "No.160 : Re-start continuation" after all axes servo are turned on and servo on of 1 axis and 2 axes is confirmed when a forced stop is released. • "No.160 : Re-start continuation" is stopped at the time of the forced stop, and actual output (PY) is turned off.
160	Restart continuation	Normal	Not start	3	(1) This program jumps corresponding to the value of the continuation point (#100) of the following (2) 1) to 9). #100 Jump destination 0 Following (2) 1) 10 Following (2) 3) 20 Following (2) 5) 30 Following (2) 8) (2) The following motion control is executed. 1) This program stands by until PX4 is turned on. 2) "10" is set on continuation point (#100). 3) 1 axis, 2 axes are located in (0,0) in the linear control (absolute 2 axes positioning). 4) Positioning completion signal on of 1 axis, 2 axes is confirmed, and "20" is set on the continuation point (#100). 5) In-position on of 1 axis and 2 axes is confirmed. 6) 1 axis, 2 axes are located in (1000000, 2000000) in the linear control (absolute 2 axes positioning). 7) Positioning completion signal on of 1 axis, 2 axes is confirmed, and "30" is set on the continuation point (#100). 8) This program stands by until PX4 is turned off. 9) "0" is set on continuation point (#100).

(a) No.20 : Main



"0" is set on the continuation point (#100) as an initial value.

The subroutine starts "No.160 : Restart continuation" after all axis servo are turned on and servo on of 1 axis and 2 axes is confirmed when a forced stop is released.
(Because the next step is a shift, it becomes a subroutine start, and the next step is executed at the same time with subroutine practice, too.)

"No.160 : Restart continuation" is made to stop by the forced stop, and real output (PY) is turned off.

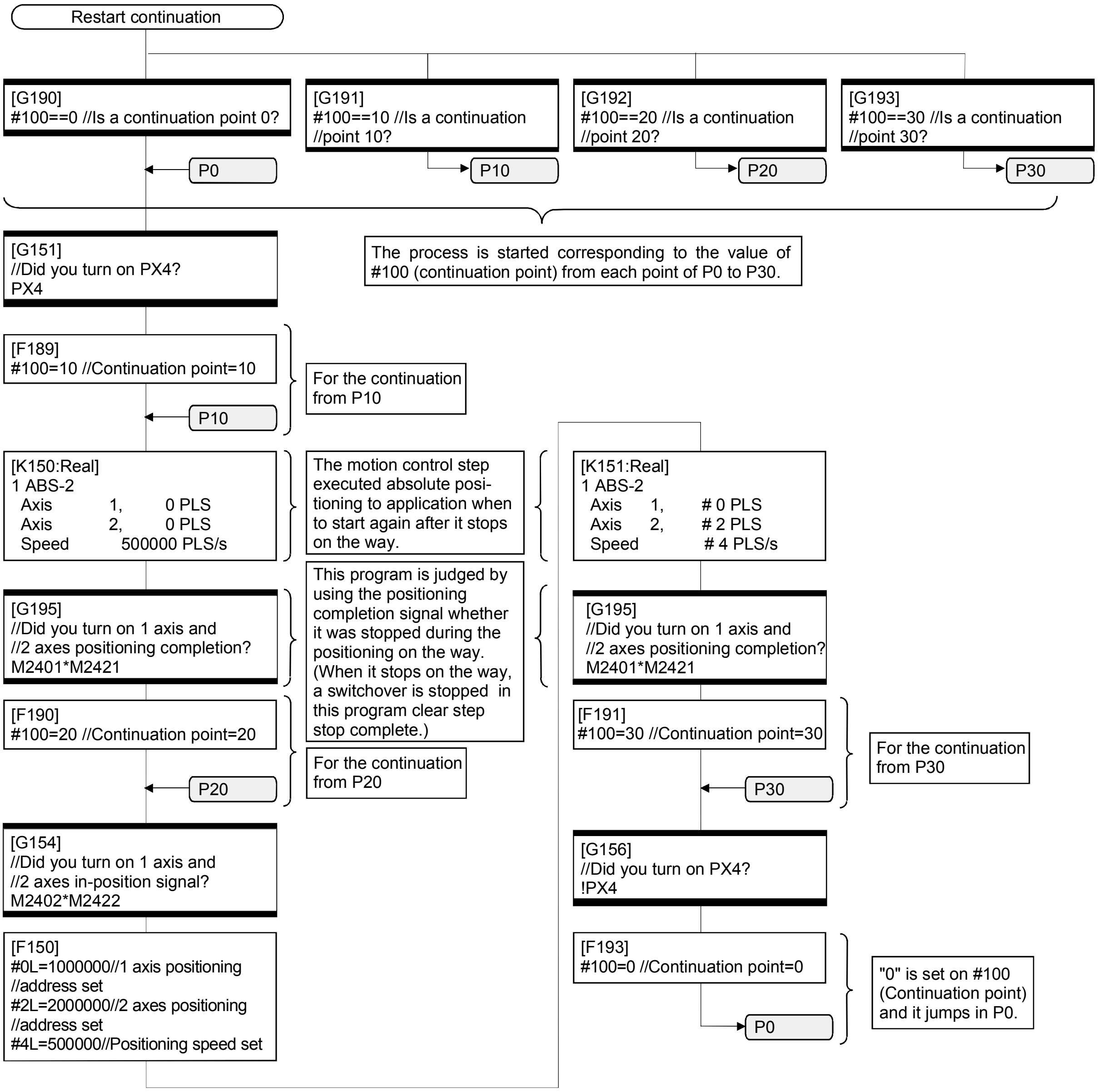
(Note-1) : The program that a subroutine was started is made to stop if necessary when a subroutine start program is added because it does not stop.

(Note-2) : Actual output is turned off if necessary.

(Note-3) : The occurrence detection of servo error and so on is added to the stop condition with forced stop if necessary.

When a forced stop is released, it is the structure which starts the program which does motion control from the initials again by sample program. Therefore it is the system example that motion control is resumed when a forced stop release is executed after it stops forced for while.

(b) No.160 : Restart continuation



APPENDIX 2.5 Continuation execution example after the stop by the Motion SFC program

(1) The explanation of the operation

The program example that the Motion SFC program is stopped by external input signal ON for the forced stop from the input module, and it is executed continuously by external signal OFF for the stop is shown below.

The servo is turned on by the forced stop release and the positioning control of the 2 axes liner interpolation is executed when PX4 is ON in this program. One cycle operation is completed after confirmation that PX4 became OFF.
When PX5 turns ON during the positioning operating, the positioning operation is stopped by the stop instruction and it is resumed from the interrupted positioning operation at turning PX5 on. The transition to the next step is not executed during PX5 is ON in the WAIT transition.
When the forced stop is executed during the positioning operating, the positioning operation is interrupted and the servomotor is stopped. It is resumed from the interrupted positioning operation when the forced stop was released next.
Continuation execution of the stop and stop after is executed by this program example by the following processing.

(a) While PX5 turns it on, it is made to turn on a stop command (M3200+20n) and an internal relay (M100) for the stop.

(b) While PX5 turns it off, it is made to turn off a stop command (M3200+20n) and an internal relay (M100) for the stop.

(c) A motion control step does absolute position to cope with it when it is resumed after it stops on the way of the positioning.

(d) A positioning completion signal (M2401+20n) is used for the decision whether it is stopped during the positioning on the way.

(e) The motion control step is resumed after it waits to turn it off, when it was stepped during positioning.

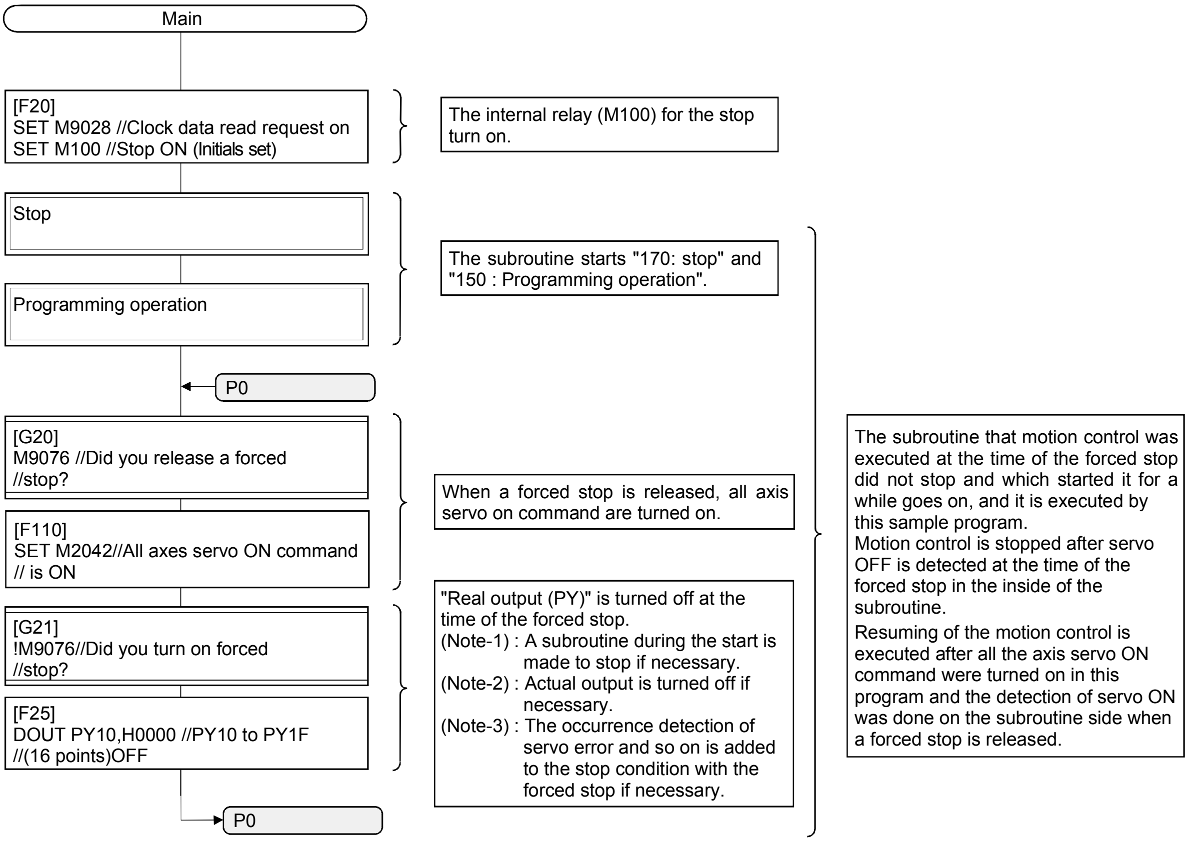
(f) "The internal relay (M100) for the stop turn off." is substituted for the WAIT transition condition that you must stop.

(2) Contents of processing SFC program

SFC program list

No.	Program name	Task	Automatic operation	Number of connective transitions	Contents of processing
20	Main	Normal	Start	3	• This program starts automatically at the time of RUN of Q173HCPU, and it is always executed. • Watch data is taken out, and clock data read request (M9028) is turned on. • The initials condition of the internal relay (M100) for the stop is turned on. • The subroutine starts "No.170 : Stop". • The subroutine starts "No.150 : Programming operation". • When an forced stop is released, all axes servo are turned on. • Turns off actual output (PY) at the time of the forced stop.
170	Stop	Normal	Not start	3	(1) When a stop input signal (PX5) from the input unit is off, the treatment of the following (2) is executed, and 1 axis and 2 axes executed the following (3) during servo on in the case of the one except for it. (2) 1 axis and 2 axes stop command are turned off, and an internal relay (M100) for the stop is turned off. (3) 1 axis and 2 axes stop command are turned on, and an internal relay (M100) for the stop is turned on.
150	Program operation	Normal	Not start	3	(1) The following motion control is executed. 1) This program stands by until PX4 is turned on. 2) 1 axis and 2 axes are located in (0,0) in the linear interpolation control (absolute 2 axes positioning). 3) Positioning completion signal on of 1 axis and 2 axes are confirmed. 4) In-position on of 1 axis and 2 axes are confirmed. 5) 1 axis and 2 axes are located in (1000000, 2000000) in the linear control (absolute 2 axes positioning). 6) Positioning completion signal on of 1 axis and 2 axes are confirmed. 7) This program stands by until PX4 is turned off. (2) When a positioning completion signal of the above (1) 3) and 6) is off, it waits to turn off, and (When a positioning was suspended on the way.) execute the motion control step (1) 2) or 5) again. (3) Until an internal relay (M100) for the stop turns it on, it does not move to the next step of the above (1) 1) and 7).

(a) No.20 : Main



```
[F20]
SET M9028 //Clock data read request on
SET M100 //Stop ON (Initials set)
```

The internal relay (M100) for the stop turn on.

The subroutine starts "170: stop" and "150 : Programming operation".

```
[G20]
M9076 //Did you release a forced
//stop?
```

```
[F110]
SET M2042//All axes servo ON command
// is ON
```

When a forced stop is released, all axis servo on command are turned on.

```
[G21]
!M9076//Did you turn on forced
//stop?
```

```
[F25]
DOUT PY10,H0000 //PY10 to PY1F
//(16 points)OFF
```

"Real output (PY)" is turned off at the time of the forced stop.

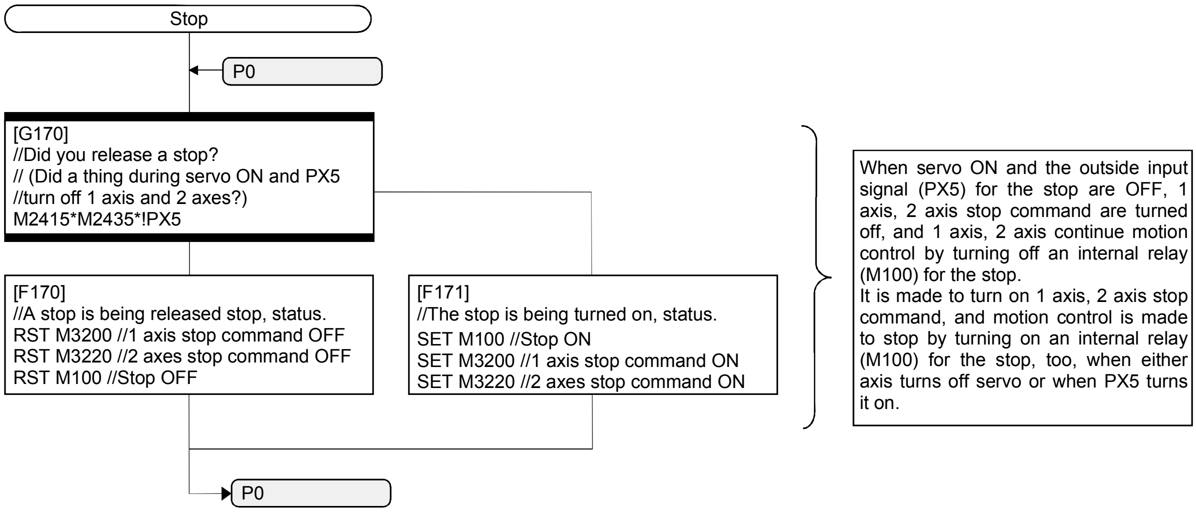
(Note-1) : A subroutine during the start is made to stop if necessary.

(Note-2) : Actual output is turned off if necessary.

(Note-3) : The occurrence detection of servo error and so on is added to the stop condition with the forced stop if necessary.

The subroutine that motion control was executed at the time of the forced stop did not stop and which started it for a while goes on, and it is executed by this sample program.
Motion control is stopped after servo OFF is detected at the time of the forced stop in the inside of the subroutine.
Resuming of the motion control is executed after all the axis servo ON command were turned on in this program and the detection of servo ON was done on the subroutine side when a forced stop is released.

(b) No.170 : Stop



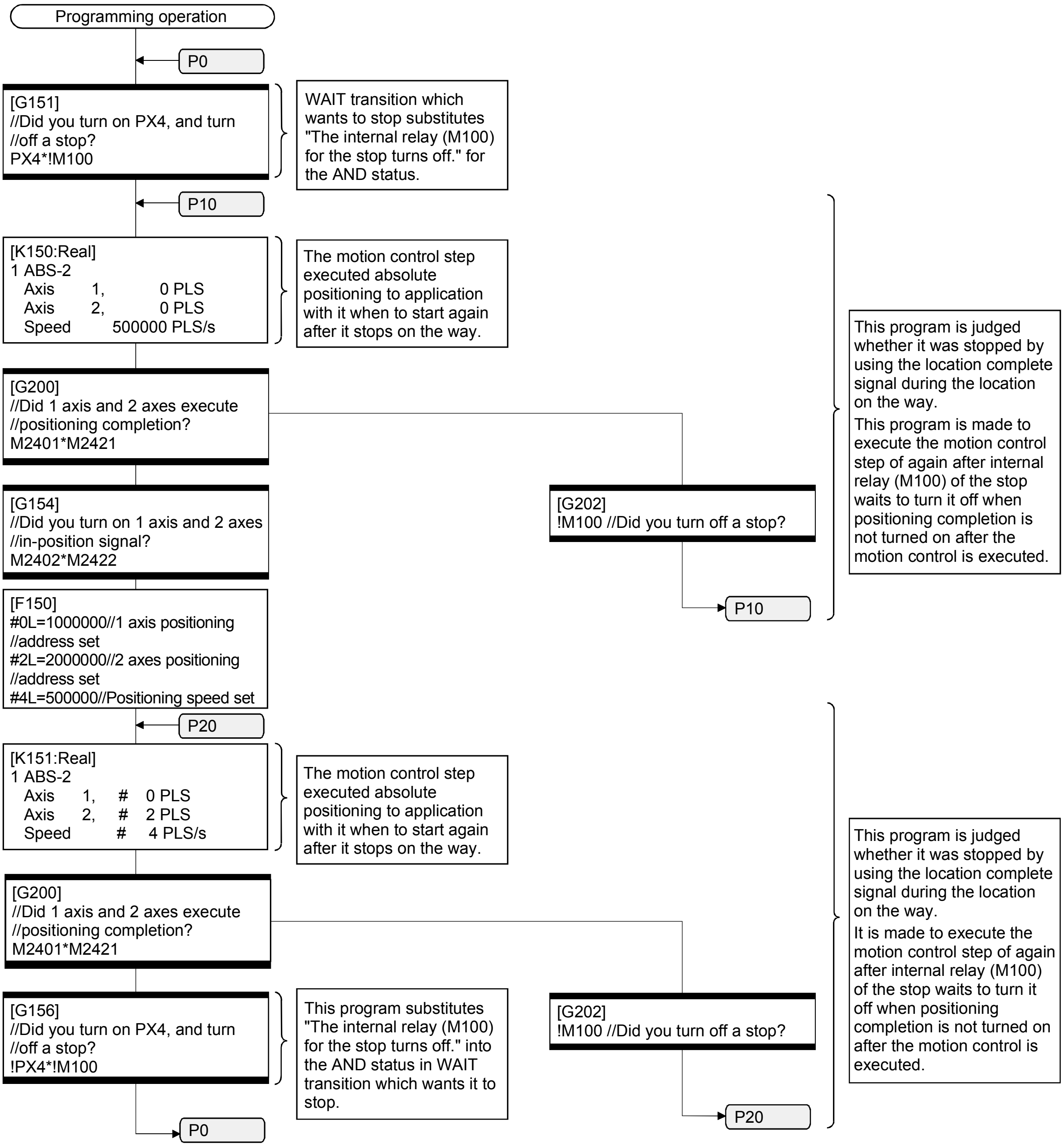
```
[G170]
//Did you release a stop?
// (Did a thing during servo ON and PX5
//turn off 1 axis and 2 axes?)
M2415*M2435*!PX5
```

```
[F170]
//A stop is being released stop, status.
RST M3200 //1 axis stop command OFF
RST M3220 //2 axes stop command OFF
RST M100 //Stop OFF
```

```
[F171]
//The stop is being turned on, status.
SET M100 //Stop ON
SET M3200 //1 axis stop command ON
SET M3220 //2 axes stop command ON
```

When servo ON and the outside input signal (PX5) for the stop are OFF, 1 axis, 2 axis stop command are turned off, and 1 axis, 2 axis continue motion control by turning off an internal relay (M100) for the stop.
It is made to turn on 1 axis, 2 axis stop command, and motion control is made to stop by turning on an internal relay (M100) for the stop, too, when either axis turns off servo or when PX5 turns it on.

(c) No.150 : Programming operation



```
[G151]
//Did you turn on PX4, and turn
//off a stop?
PX4*!M100
```

WAIT transition which wants to stop substitutes "The internal relay (M100) for the stop turns off." for the AND status.

```
[K150:Real]
1 ABS-2
   Axis      1,          0 PLS
   Axis      2,          0 PLS
   Speed        500000 PLS/s
```

The motion control step executed absolute positioning to application with it when to start again after it stops on the way.

```
[G200]
//Did 1 axis and 2 axes execute
//positioning completion?
M2401*M2421
```

```
[G154]
//Did you turn on 1 axis and 2 axes
//in-position signal?
M2402*M2422
```

```
[G202]
!M100 //Did you turn off a stop?
```

```
[F150]
#0L=1000000//1 axis positioning
//address set
#2L=2000000//2 axes positioning
//address set
#4L=500000//Positioning speed set
```

This program is judged whether it was stopped by using the location complete signal during the location on the way.

This program is made to execute the motion control step of again after internal relay (M100) of the stop waits to turn it off when positioning completion is not turned on after the motion control is executed.

```
[K151:Real]
1 ABS-2
   Axis     1,    #   0 PLS
   Axis     2,    #   2 PLS
   Speed          #    4 PLS/s
```

The motion control step executed absolute positioning to application with it when to start again after it stops on the way.

```
[G200]
//Did 1 axis and 2 axes execute
//positioning completion?
M2401*M2421
```

```
[G156]
//Did you turn on PX4, and turn
//off a stop?
!PX4*!M100
```

This program substitutes "The internal relay (M100) for the stop turns off." into the AND status in WAIT transition which wants it to stop.

```
[G202]
!M100 //Did you turn off a stop?
```

This program is judged whether it was stopped by using the location complete signal during the location on the way.

It is made to execute the motion control step of again after internal relay (M100) of the stop waits to turn it off when positioning completion is not turned on after the motion control is executed.

WARRANTY

Please confirm the following product warranty details before using this product.

1. Gratis Warranty Term and Gratis Warranty Range

If any faults or defects (hereinafter "Failure") found to be the responsibility of Mitsubishi occurs during use of the product within the gratis warranty term, the product shall be repaired at no cost via the sales representative or Mitsubishi Service Company.
However, if repairs are required onsite at domestic or overseas location, expenses to send an engineer will be solely at the customer's discretion. Mitsubishi shall not be held responsible for any re-commissioning, maintenance, or testing on-site that involves replacement of the failed module.

[Gratis Warranty Term]

Note that an installation period of less than one year after installation in your company or your customer's premises or a period of less than 18 months (counted from the date of production) after shipment from our company, whichever is shorter, is selected.

[Gratis Warranty Range]

(1) Diagnosis of failure

As a general rule, diagnosis of failure is done on site by the customer.
However, Mitsubishi or Mitsubishi service network can perform this service for an agreed upon fee upon the customer's request.
There will be no charges if the cause of the breakdown is found to be the fault of Mitsubishi.

(2) Breakdown repairs

There will be a charge for breakdown repairs, exchange replacements and on site visits for the following four conditions, otherwise there will be a charge.

1) Breakdowns due to improper storage, handling, careless accident, software or hardware design by the customer
2) Breakdowns due to modifications of the product without the consent of the manufacturer
3) Breakdowns resulting from using the product outside the specified specifications of the product
4) Breakdowns that are outside the terms of warranty

Since the above services are limited to Japan, diagnosis of failures, etc. are not performed abroad.
If you desire the after service abroad, please register with Mitsubishi. For details, consult us in advance.

2. Exclusion of Loss in Opportunity and Secondary Loss from Warranty Liability

Mitsubishi will not be held liable for damage caused by factors found not to be the cause of Mitsubishi; opportunity loss or lost profits caused by faults in the Mitsubishi products; damage, secondary damage, accident compensation caused by special factors unpredictable by Mitsubishi; damages to products other than Mitsubishi products; and to other duties.

3. Onerous Repair Term after Discontinuation of Production

Mitsubishi shall accept onerous product repairs for seven years after production of the product is discontinued.

4. Delivery Term

In regard to the standard product, Mitsubishi shall deliver the standard product without application settings or adjustments to the customer and Mitsubishi is not liable for on site adjustment or test run of the product.

5. Precautions for Choosing the Products

(1) These products have been manufactured as a general-purpose part for general industries, and have not been designed or manufactured to be incorporated in a device or system used in purposes related to human life.
(2) Before using the products for special purposes such as nuclear power, electric power, aerospace, medicine, passenger movement vehicles or under water relays, contact Mitsubishi.
(3) These products have been manufactured under strict quality control. However, when installing the product where major accidents or losses could occur if the product fails, install appropriate backup or failsafe functions in the system.
(4) When exporting any of the products or related technologies described in this catalogue, you must obtain an export license if it is subject to Japanese Export Control Law.

IB(NA)-0300112-B(0609)MEE
MODEL: Q173H-P-SV13/22-SFCE
MODEL CODE: 1XB912

MITSUBISHI ELECTRIC CORPORATION

HEAD OFFICE : TOKYO BUILDING, 2-7-3 MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN
NAGOYA WORKS : 1-14 , YADA-MINAMI 5-CHOME , HIGASHI-KU, NAGOYA , JAPAN