for a greener tomorrow

MITSUBISHI ELECTRIC
Changes for the Better

Mistubishi Electric Servo System Motion Module Quick Start Guide

Let's Start!

Quick Start Guide

MELSEC iQ-R Series Motion Module

MITSUBISHI ELECTRIC SERVO SYSTEM MELSERVO-J5

MELSEC iQ-R series

Applicable Model

- -RD78G
- -RD78GH
- -MR-J5-G

● SAFETY PRECAUTIONS ●

(Read these precautions before using this product.)

Before using this product, please read this manual and the relevant manuals carefully and pay full attention to safety to handle the product correctly.
The precautions given in this manual are concerned with this product only. Refer to the MELSEC iQ-R Module Configuration Manual for a description of the PLC system safety precautions.
In this manual, the safety precautions are classified into two levels: "⚠WARNING" and "⚠CAUTION".

⚠WARNING	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 minor or moderate injury or property damage.

Under some circumstances, failure to observe the precautions given under "⚠CAUTION" may lead to serious consequences.
Observe the precautions of both levels because they are important for personal and system safety.
Make sure that the end users read this manual and then keep the manual in a safe place for future reference.

[Design Precautions]

⚠WARNING

● Configure safety circuits external to the programmable controller to ensure that the entire system operates safely even when a fault occurs in the external power supply or the programmable controller. Failure to do so may result in an accident due to an incorrect output or malfunction.

(1) Emergency stop circuits, protection circuits, and protective interlock circuits for conflicting operations (such as forward/reverse rotations or upper/lower limit positioning) must be configured external to the programmable controller.

(2) When the programmable controller detects an abnormal condition, it stops the operation and all outputs are:

• Turned off if the overcurrent or overvoltage protection of the power supply module is activated.

• Held or turned off according to the parameter setting if the self-diagnostic function of the CPU module detects an error such as a watchdog timer error.

(3) All outputs may be turned on if an error occurs in a part, such as an I/O control part, where the CPU module cannot detect any error. To ensure safety operation in such a case, provide a safety mechanism or a fail-safe circuit external to the programmable controller. For a fail-safe circuit example, refer to "General Safety Requirements" in the MELSEC iQ-R Module Configuration Manual.

(4) Outputs may remain on or off due to a failure of a component such as a relay and transistor in an output circuit. Configure an external circuit for monitoring output signals that could cause a serious accident.

● In an output circuit, when a load current exceeding the rated current or an overcurrent caused by a load short-circuit flows for a long time, it may cause smoke and fire. To prevent this, configure an external safety circuit, such as a fuse.

● Configure a circuit so that the programmable controller is turned on first and then the external power supply. If the external power supply is turned on first, an accident may occur due to an incorrect output or malfunction.

● For the operating status of each station after a communication failure, refer to manuals for the network used. For the manuals, please consult your local Mitsubishi representative. Incorrect output or malfunction due to a communication failure may result in an accident.

● When connecting an external device with a CPU module or intelligent function module to modify data of a running programmable controller, configure an interlock circuit in the program to ensure that the entire system will always operate safely. For other forms of control (such as program modification, parameter change, forced output, or operating status change) of a running programmable controller, read the relevant manuals carefully and ensure that the operation is safe before proceeding. Improper operation may damage machines or cause accidents. When a Safety CPU is used, data cannot be modified while the Safety CPU is in SAFETY MODE.

[Design Precautions]

⚠WARNING

- Especially, when a remote programmable controller is controlled by an external device, immediate action cannot be taken if a problem occurs in the programmable controller due to a communication failure. To prevent this, configure an interlock circuit in the program, and determine corrective actions to be taken between the external device and CPU module in case of a communication failure.
- Do not write any data to the "system area" and "write-protect area" of the buffer memory in the module. Also, do not use any "use prohibited" signals as an output signal from the CPU module to each module. Doing so may cause malfunction of the programmable controller system. For the "system area", "write-protect area", and the "use prohibited" signals, refer to the user's manual for the module used. For areas used for safety communications, they are protected from being written by users, and thus safety communications failure caused by data writing does not occur.
- If a communication cable is disconnected, the network may be unstable, resulting in a communication failure of multiple stations. Configure an interlock circuit in the program to ensure that the entire system will always operate safely even if communications fail. Incorrect output or malfunction due to a communication failure may result in an accident. When safety communications are used, an interlock by the safety station interlock function protects the system from an incorrect output or malfunction.
- Configure safety circuits external to the programmable controller to ensure that the entire system operates safely even when a fault occurs in the external power supply or the programmable controller. Failure to do so may result in an accident due to an incorrect output or malfunction.
  (1) Machine homing is controlled by two kinds of data: a homing direction and a homing speed. Deceleration starts when the proximity dog signal turns on. If an incorrect homing direction is set, motion control may continue without deceleration. To prevent machine damage caused by this, configure an interlock circuit external to the programmable controller.
  (2) When the module detects an error, the motion slows down and stops or the motion rapidly stops, depending on the stop group setting in parameter. Set the parameter to meet the specifications of a positioning control system. In addition, set the homing parameter and positioning data within the specified setting range.
  (3) Outputs may remain on or off, or become undefined due to a failure of a component such as an insulation element and transistor in an output circuit, where the module cannot detect any error. In a system that the incorrect output could cause a serious accident, configure an external circuit for monitoring output signals.
- If safety standards (ex., robot safety rules, etc.,) apply to the system using the module, drive unit and servomotor, make sure that the safety standards are satisfied.
- Construct a safety circuit externally of the module or drive unit if the abnormal operation of the module or drive unit differs from the safety directive operation in the system.

[Design Precautions]

⚠CAUTION

- Do not install the control lines or communication cables together with the main circuit lines or power cables. Keep a distance of 100 mm or more between them. Failure to do so may result in malfunction due to noise.
- During control of an inductive load such as a lamp, heater, or solenoid valve, a large current (approximately ten times greater than normal) may flow when the output is turned from off to on. Therefore, use a module that has a sufficient current rating.
- After the CPU module is powered on or is reset, the time taken to enter the RUN status varies depending on the system configuration, parameter settings, and/or program size. Design circuits so that the entire system will always operate safely, regardless of the time.
- Do not power off the programmable controller or reset the CPU module while the settings are being written. Doing so will make the data in the flash ROM and SD memory card undefined. The values need to be set in the buffer memory and written to the flash ROM and SD memory card again. Doing so also may cause malfunction or failure of the module.

[Security Precautions]

⚠WARNING

- To maintain the security (confidentiality, integrity, and availability) of the programmable controller and the system against unauthorized access, denial-of-service (DoS) attacks, computer viruses, and other cyberattacks from external devices via the network, take appropriate measures such as firewalls, virtual private networks (VPNs), and antivirus solutions.

[Installation Precautions]

⚠WARNING

- Shut off the external power supply (all phases) used in the system before mounting or removing the module. Failure to do so may result in electric shock or cause the module to fail or malfunction.

[Installation Precautions]

⚠CAUTION

- Use the programmable controller in an environment that meets the general specifications in the Safety Guidelines included with the base unit. Failure to do so may result in electric shock, fire, malfunction, or damage to or deterioration of the product.
- To mount a module, place the concave part(s) located at the bottom onto the guide(s) of the base unit, and push in the module, and fix it with screw(s). Incorrect interconnection may cause malfunction, failure, or drop of the module.
- To mount a module with no module fixing hook, place the concave part(s) located at the bottom onto the guide(s) of the base unit, push in the module, and fix it with screw(s). Incorrect interconnection may cause malfunction, failure, or drop of the module.
- Tighten the screws within the specified torque range. Undertightening can cause drop of the screw, short circuit, or malfunction. Overtightening can damage the screw and/or module, resulting in drop, short circuit, or malfunction.
- When using an extension cable, connect it to the extension cable connector of the base unit securely. Check the connection for looseness. Poor contact may cause malfunction.
- When using an SD memory card, fully insert it into the SD memory card slot. Check that it is inserted completely. Poor contact may cause malfunction.
- Securely insert an extended SRAM cassette or a battery-less option cassette into the cassette connector of the CPU module. After insertion, close the cassette cover and check that the cassette is inserted completely. Poor contact may cause malfunction.
- Do not directly touch any conductive parts and electronic components of the module, SD memory card, extended SRAM cassette, battery-less option cassette, or connector. Doing so can cause malfunction or failure of the module.

[Wiring Precautions]

⚠WARNING

- Shut off the external power supply (all phases) used in the system before installation and wiring. Failure to do so may result in electric shock or cause the module to fail or malfunction.
- After installation and wiring, attach a blank cover module (RG60) to each empty slot and an included extension connector protective cover to the unused extension cable connector before powering on the system for operation. Failure to do so may result in electric shock.

[Wiring Precautions]

⚠CAUTION

- Individually ground the FG and LG terminals of the programmable controller with a ground resistance of 100 ohms or less. Failure to do so may result in electric shock or malfunction.
- Use applicable solderless terminals and tighten them within the specified torque range. If any spade solderless terminal is used, it may be disconnected when the terminal screw comes loose, resulting in failure.
- Check the rated voltage and signal layout before wiring to the module, and connect the cables correctly. Connecting a power supply with a different voltage rating or incorrect wiring may cause fire or failure.
- Connectors for external devices must be crimped or pressed with the tool specified by the manufacturer, or must be correctly soldered. Incomplete connections may cause short circuit, fire, or malfunction.
- Securely connect the connector to the module. Poor contact may cause malfunction.
- Do not install the control lines or communication cables together with the main circuit lines or power cables. Keep a distance of 100 mm or more between them. Failure to do so may result in malfunction due to noise.
- Place the cables in a duct or clamp them. If not, dangling cables may swing or inadvertently be pulled, resulting in malfunction or damage to modules or cables. In addition, the weight of the cables may put stress on modules in an environment of strong vibrations and shocks. Do not clamp the extension cables with the jacket stripped. Doing so may change the characteristics of the cables, resulting in malfunction.
- Check the interface type and correctly connect the cable. Incorrect wiring (connecting the cable to an incorrect interface) may cause failure of the module and external device.
- Tighten the terminal screws or connector screws within the specified torque range. Undertightening can cause drop of the screw, short circuit, fire, or malfunction. Overtightening can damage the screw and/or module, resulting in drop, short circuit, fire, or malfunction.
- When disconnecting the cable from the module, do not pull the cable by the cable part. For the cable with connector, hold the connector part of the cable. For the cable connected to the terminal block, loosen the terminal screw. Pulling the cable connected to the module may result in malfunction or damage to the module or cable.
- Prevent foreign matter such as dust or wire chips from entering the module. Such foreign matter can cause a fire, failure, or malfunction.
- A protective film is attached to the top of the module to prevent foreign matter, such as wire chips, from entering the module during wiring. Do not remove the film during wiring. Remove it for heat dissipation before system operation.
- Programmable controllers must be installed in control panels. Connect the main power supply to the power supply module in the control panel through a relay terminal block. Wiring and replacement of a power supply module must be performed by qualified maintenance personnel with knowledge of protection against electric shock. For wiring, refer to the MELSEC iQ-R Module Configuration Manual.
- For Ethernet cables to be used in the system, select the ones that meet the specifications in this manual. If not, normal data transmission is not guaranteed.

[Startup and Maintenance Precautions]

⚠WARNING

- Do not touch any terminal while power is on. Doing so will cause electric shock or malfunction.
- Correctly connect the battery connector. Do not charge, disassemble, heat, short-circuit, solder, or throw the battery into the fire. Also, do not expose it to liquid or strong shock. Doing so will cause the battery to produce heat, explode, ignite, or leak, resulting in injury and fire.
- Shut off the external power supply (all phases) used in the system before cleaning the module or retightening the terminal screws, connector screws, or module fixing screws. Failure to do so may result in electric shock.

[Startup and Maintenance Precautions]

⚠CAUTION

- When connecting an external device with a CPU module or intelligent function module to modify data of a running programmable controller, configure an interlock circuit in the program to ensure that the entire system will always operate safely. For other forms of control (such as program modification, parameter change, forced output, or operating status change) of a running programmable controller, read the relevant manuals carefully and ensure that the operation is safe before proceeding. Improper operation may damage machines or cause accidents.
- Especially, when a remote programmable controller is controlled by an external device, immediate action cannot be taken if a problem occurs in the programmable controller due to a communication failure. To prevent this, configure an interlock circuit in the program, and determine corrective actions to be taken between the external device and CPU module in case of a communication failure.
- Do not disassemble or modify the modules. Doing so may cause failure, malfunction, injury, or a fire.
- Use any radio communication device such as a cellular phone or PHS (Personal Handy-phone System) more than 25 cm away in all directions from the programmable controller. Failure to do so may cause malfunction.
- Shut off the external power supply (all phases) used in the system before mounting or removing the module. Failure to do so may cause the module to fail or malfunction.
- Tighten the screws within the specified torque range. Undertightening can cause drop of the component or wire, short circuit, or malfunction. Overtightening can damage the screw and/or module, resulting in drop, short circuit, or malfunction.
- After the first use of the product, do not perform each of the following operations more than 50 times (IEC 61131-2/JIS B 3502 compliant). Exceeding the limit may cause malfunction.
  - Mounting/removing the module to/from the base unit
  - Inserting/removing the extended SRAM cassette or battery-less option cassette to/from the CPU module
  - Mounting/removing the terminal block to/from the module.
- After the first use of the product, do not insert/remove the SD memory card to/from the CPU module more than 500 times. Exceeding the limit may cause malfunction.

[Startup and Maintenance Precautions]

⚠CAUTION

- Do not touch the metal terminals on the back side of the SD memory card. Doing so may cause malfunction or failure of the module.
- Do not touch the integrated circuits on the circuit board of an extended SRAM cassette or a battery-less option cassette. Doing so may cause malfunction or failure of the module.
- Do not drop or apply shock to the battery to be installed in the module. Doing so may damage the battery, causing the battery fluid to leak inside the battery. If the battery is dropped or any shock is applied to it, dispose of it without using.
- Startup and maintenance of a control panel must be performed by qualified maintenance personnel with knowledge of protection against electric shock. Lock the control panel so that only qualified maintenance personnel can operate it.
- Before handling the module, touch a conducting object such as a grounded metal to discharge the static electricity from the human body. Failure to do so may cause the module to fail or malfunction.
- Before testing the operation, set a low speed value for the speed limit parameter so that the operation can be stopped immediately upon occurrence of a hazardous condition.
- Confirm and adjust the program and each parameter before operation. Unpredictable movements may occur depending on the machine.
- When using the absolute position system function, on starting up, and when the module or absolute position motor has been replaced, always perform a homing.
- Before starting the operation, confirm the brake function.
- Do not perform a megger test (insulation resistance measurement) during inspection.
- After maintenance and inspections are completed, confirm that the position detection of the absolute position detection function is correct.
- Lock the control panel and prevent access to those who are not certified to handle or install electric equipment.

[Operating Precautions]

⚠CAUTION

- When changing data and operating status, and modifying program of the running programmable controller from an external device such as a personal computer connected to an intelligent function module, read relevant manuals carefully and ensure the safety before operation. Incorrect change or modification may cause system malfunction, damage to the machines, or accidents.
- Do not power off the programmable controller or reset the CPU module while the setting values in the buffer memory are being written to the flash ROM in the module. Doing so will make the data in the flash ROM and SD memory card undefined. The values need to be set in the buffer memory and written to the flash ROM and SD memory card again. Doing so can cause malfunction or failure of the module.
- Note that when the reference axis speed is specified for interpolation operation, the speed of the partner axis (2nd, 3rd, or 4th axis) may exceed the speed limit value.
- Do not go near the machine during test operations or during operations such as teaching. Doing so may lead to injuries.

[Disposal Precautions]

⚠CAUTION

- When disposing of this product, treat it as industrial waste.
- When disposing of batteries, separate them from other wastes according to the local regulations. For details on battery regulations in EU member states, refer to the MELSEC iQ-R Module Configuration Manual.

[Transportation Precautions]

⚠CAUTION

- When transporting lithium batteries, follow the transportation regulations. For details on the regulated models, refer to the MELSEC iQ-R Module Configuration Manual.
- The halogens (such as fluorine, chlorine, bromine, and iodine), which are contained in a fumigant used for disinfection and pest control of wood packaging materials, may cause failure of the product. Prevent the entry of fumigant residues into the product or consider other methods (such as heat treatment) instead of fumigation. The disinfection and pest control measures must be applied to unprocessed raw wood.

REVISIONS

Revision Date	Manual No.	Description
Oct. 2020	L(NA)03191ENG-A	First edition

INTRODUCTION

Please read this manual carefully so that equipment is used to its optimum.

CONTENTS

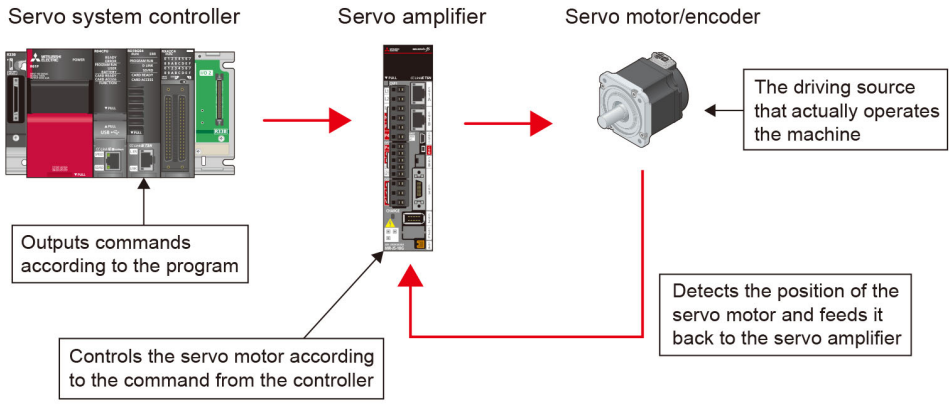
MEMO

1. OVERVIEW

This document is intended for first-time users of Mitsubishi Electric's servo system controller RD78G(H) to learn about necessary system components, software, parameter settings, and programming methods to configure a servo system.

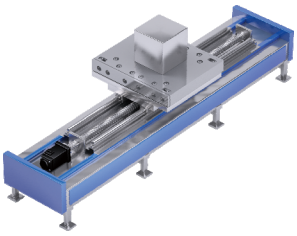
1.1 What Is A Servo System?

A servo system controls an object following the input command values, such as the position, direction, attitude, etc. of the controlled object.
The Mitsubishi Electric's servo system, as shown below, controls a servo motor by transmitting position and speed commands from a controller (a programmable controller and a servo system controller) to a servo amplifier.
An encoder is mounted on the servo motor to detect the position and speed. The detected data is fed back to the servo amplifier so that the deviation between the position/speed commands and the actual position/speed is minimized.

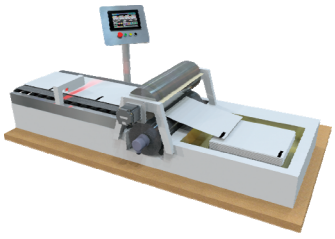


1.2 Application Examples of Servo Systems

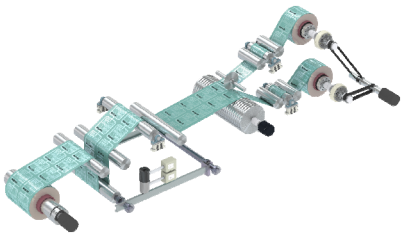
A servo system is used in a wide range of machines such as ones that require high-accuracy positioning, ones that require accurate synchronization such as packing machines and printing machines, and ones that require tension control (torque control) and speed control such as unwinding/winding machines, etc.



[A machine that requires position control]



[A machine that requires synchronous control]



[A machine that requires torque control]

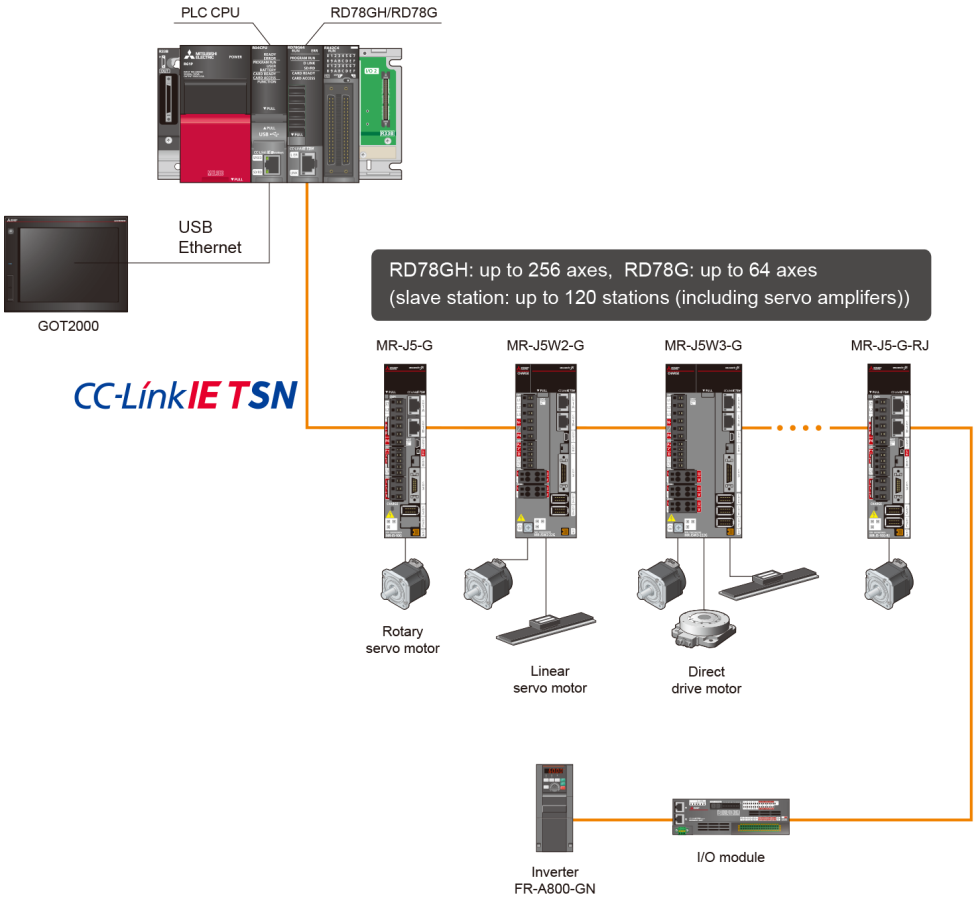
1.3 Role of Servo System Controllers

The servo system controller outputs position/speed commands to servo amplifiers and generates commands to execute interpolation and synchronous control using multiple axes.
In addition, the controller changes the position/speed commands according to the value of the sensors and other devices controlled by a programmable controller.

1.4 Features of MELSEC iQ-R Series Motion Module RD78G(H)

MELSEC iQ-R series Motion module RD78G(H) is a servo system controller supporting CC-Link IE TSN that enables mixing of real-time control communication required for motion control and non-real time information communication utilized by the IT systems. By supporting the network, the Motion module can flexibly connect various devices including servo amplifiers, I/O modules, and high-speed counters.

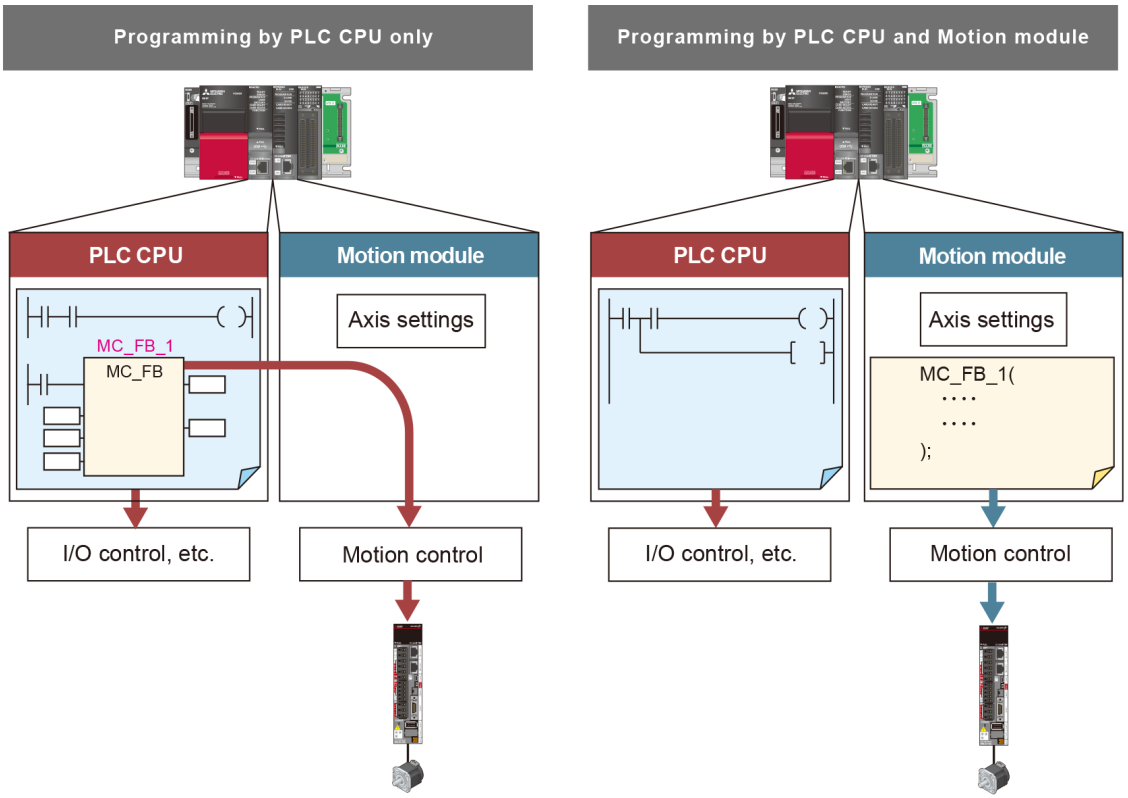
Main features of RD78G(H)
- High-speed processing with multi-core processor
- Up to 256 control axes
- Programming with the PLCopen® Motion Control FB (function block) for positioning control, etc.



1.5 Programming Methods

There are two programming methods for a servo system using RD78G(H): programming by a PLC CPU only and programming by a PLC CPU and the Motion module.
Selectable programming languages vary depending on the controller:

- Motion module : structured text language (ST)
- PLC CPU : ladder diagram (Ladder), function block diagram/ladder diagram (FBD/LD), sequential function chart (SFC), and structured text language (ST)
  (These languages are compliant with IEC 61131-3.)



(1) Programming by a PLC CPU only

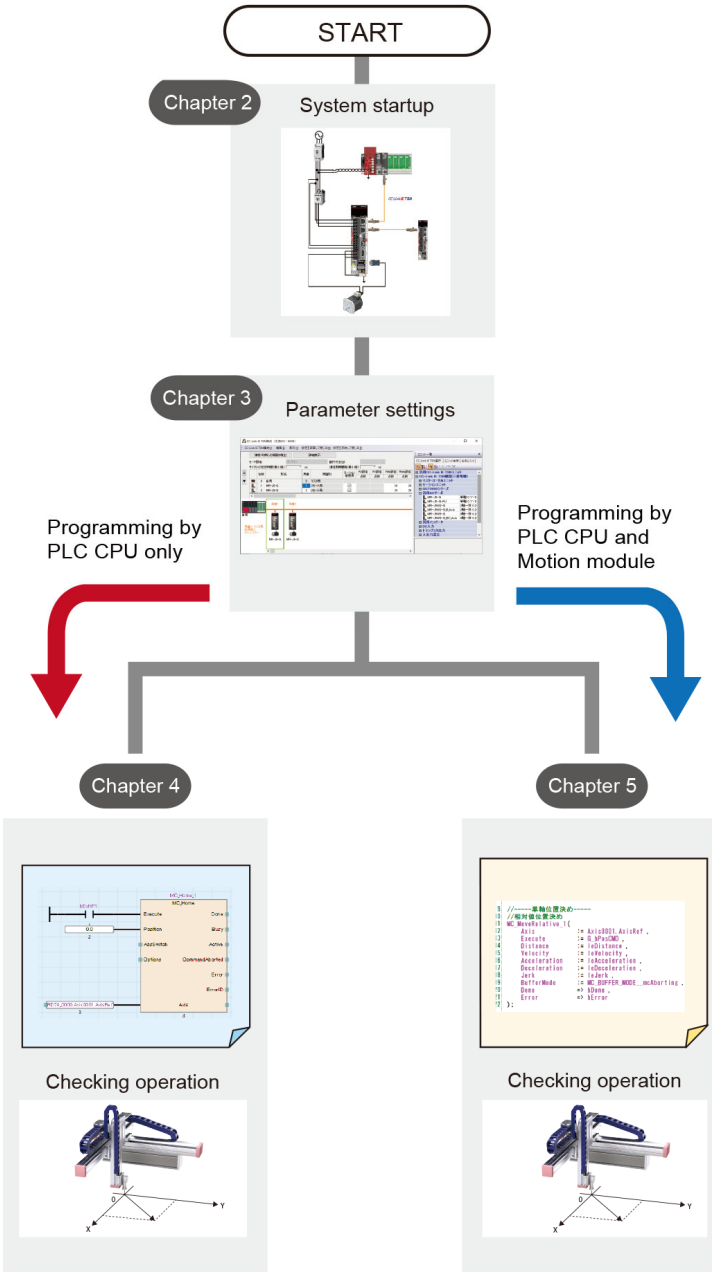
- Advantages : Easy program management by programming with only one CPU.
- Disadvantages : Longer scan time due to processing by one CPU.

(2) Programming by a PLC CPU and the Motion module

- Advantages : Reduced scan time because motion control is programmed to the Motion module, not affecting other processing in a sequence program.
- Disadvantages : Necessary to manage programs of both the PLC CPU and the Motion module.

1.6 Organization of This Document

This document describes procedures from startup to debugging by taking a two-axis XY table as an example.
The programming part consists of two chapters: programming by a PLC CPU only (Chapter 4), and programming by a PLC CPU and the Motion module (Chapter 5).



System startup
- System components preparation
- Firmware/software preparation
- Profile registration
- Motion module FB library registration
- Module installation
- Wiring and cable connection
- Rotary switch setting

Parameter settings
- Creating a project
- Network configuration
- Parameter settings
- Motion control setting
- Axes group setting
- Labels
- Public labels

Programming
- JOG, homing
- Positioning
- Linear interpolation control
- Synchronous control

Checking operation
- Axis monitor
- Program watch
- Program monitor
- Event history

1.7 Relevant Manuals

(1) Motion module

Manual title	Manual No.
MELSEC iQ-R Motion Module User's Manual (Startup)	IB-0300406ENG
MELSEC iQ-R Motion Module User's Manual (Application)	IB-0300411ENG
MELSEC iQ-R Motion Module User's Manual (Network)	IB-0300426ENG
MELSEC iQ-R Programming Manual (Motion Control Function Blocks)	IB-0300533ENG
MELSEC iQ-R Programming Manual (Motion Module Instructions, Standard Functions/Function Blocks)	IB-0300431ENG

(2) Programmable controller

Manual title	Manual No.
MELSEC iQ-R CPU Module User's Manual (Startup)	SH-081263ENG
MELSEC iQ-R CPU Module User's Manual (Application)	SH-081264ENG
MELSEC iQ-R Ethernet/CC-Link IE User's Manual (Startup)	SH-081256ENG
MELSEC iQ-R Ethernet/CC-Link IE User's Manual (Application)	SH-081257ENG
MELSEC iQ-R Programming Manual (Program Design)	SH-081265ENG
MELSEC iQ-R Structured Text (ST) Programming Guide Book	SH-081483ENG
MELSEC iQ-R Programming Manual (CPU Module Instructions, Standard Functions/Function Blocks)	SH-081266ENG

(3) Servo amplifier

Manual title	Manual No.
MR-J5 User's Manual (Hardware)	SH-030298ENG
MR-J5-G/MR-J5W-G User's Manual (Introduction)	SH-030294ENG
MR-J5-G/MR-J5W-G User's Manual (Parameters)	SH-030308ENG
MR-J5 User's Manual (Function)	SH-030300ENG
MR-J5 User's Manual (Communication Function)	SH-030302ENG
MR-J5 User's Manual (Trouble Shooting)	SH-030312ENG
MR-J5 User's Manual (Object Dictionary)	SH-030304ENG

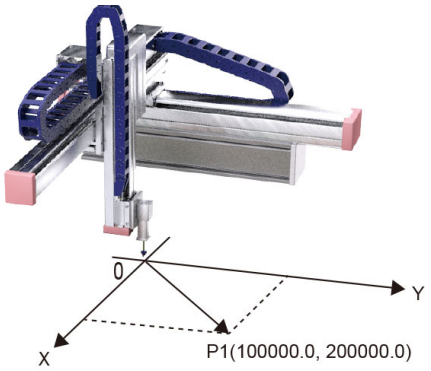
MEMO

2. SYSTEM STARTUP

2.1 System Overview

The following shows an example of a two-axis machine that uses ball screws.

(1) Machine



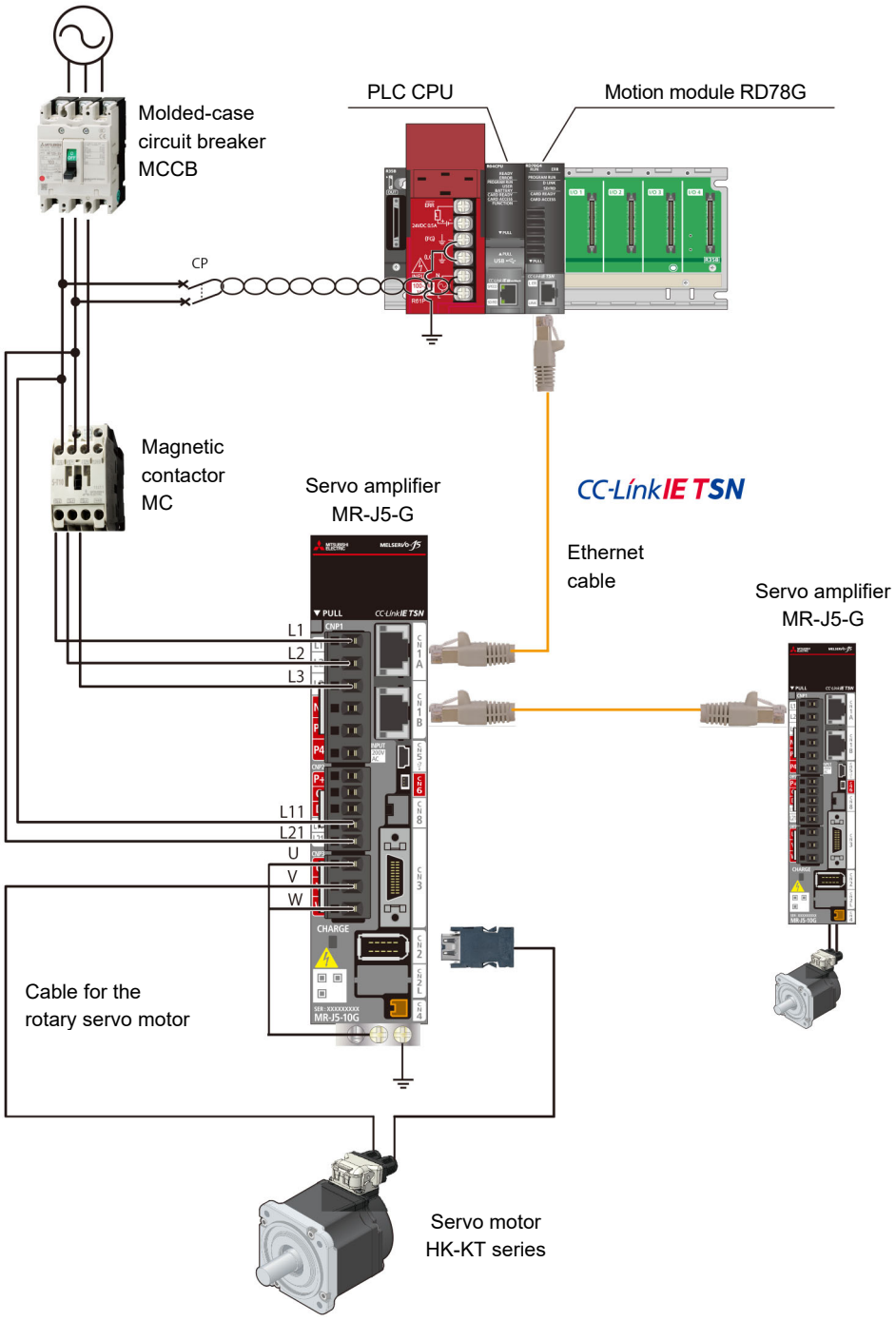
[unit: µm]

(2) Specifications

Lead of the ball screw (PB)	: 10.0 [mm]
Reduction ratio (NL/NM)	: 1/2 (Load side [NL]/Motor side [NM]) When the servo motor rotates twice, the ball screw of the load side rotates once.
Encoder resolution	: 26 bits (67108864 [pulse])

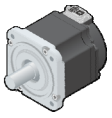
2.2 System Configuration

The following shows a system configuration example using a Motion module (RD78G), a servo amplifier (MR-J5-G), and a servo motor (HK-KT series).



2.3 System Components Preparation

This document describes a project using the following components.
Prepare modules, devices, cables, and software according to the user's system.

Motion module	Engineering environment	
RD78G4 	MELSOFT GX Works3 	
Servo amplifier	**Servo motor**	
MR-J5-10G 	HK-KT13W 	
Main base unit	**Power supply module**	**CPU module**
R35B 	R61P 	R04CPU 
Cable for the rotary servo motor	**Ethernet cable**	**USB cable**
MR-AEP2CBL2M-A1-L 	Category 5e or higher (double shielded/STP) straight cable 	MR-J3USBCBL3M 
Molded-case circuit breaker (MCCB)	**Magnetic contactor (MC)**	**Circuit protector (CP)**
		

2.4 Firmware/Software Installation

2.4.1 Firmware/software preparation

Contents of this document are based on the specifications of the firmware, software, and engineering environment listed in the following tables.
The screen displays may differ depending on the version of the firmware/software to be used.
Please contact your local sales office for the latest version.

[Firmware/software]

Module	Description	Module model	Version
PLC CPU module	Firmware update information file	R00CPU/R01CPU/R02CPU	14 or later
		R04CPU/R08CPU/R16CPU/R32CPU/R120CPU (Note-1)	46 or later
		R□ENCPU (Note-1)	46 or later
Motion module	Network boot software	RD78G/RD78GH	04 or later
	Boot software		04 or later
	Basic system software		09 or later
Servo amplifier	Firmware for CC-Link IE TSN-compatible servo amplifier MR-J5-G		A7 or later

(Note-1): This version can be installed to a controller with ver. 23 or later.

[Engineering environment]

Item	Description	Version
MELSOFT GX Works3	PLC engineering software	1.065T or later
Motion control setting	Software to set and monitor the Motion module	1.011M or later
MELSOFT MR Configurator2	Servo amplifier setup software	1.105K or later
MELSOFT GX LogViewer	An easy-to-use tool to display and analyze large-capacity data collected by the logging function of a module	1.106K or later

[Profile]

Item	Description	Version
Profile for MR-J5-G	Profiles for MELSERVO-J5 series CC-Link IE TSN-compatible servo amplifier	02 or later

[FB library] *Use it when programming with a PLC CPU.

Item	Description	Version
MELSEC iQ-R Motion module FB library	An FB library for the systems using the MELSEC iQ-R series Motion module	01B or later

2.4.2 MELSOFT GX Works3 installation

(1) Check before installation
- Log on to the personal computer as an administrator.
- Close all running applications before installation. If the product is installed while other applications are running, it may not operate normally.

(2) Installation procedure

1) Insert the MELSOFT GX Works3 DVD-ROM to the DVD-ROM drive, and double-click "setup.exe" in the Disk1 folder on the DVD-ROM.

2) Select or enter the necessary information by following the on-screen instructions. (Note-1)

(Note-1): The product ID is written on the "License certificate" included with the product.
Enter the 12-digit number divided into 3 and 9 digits.

[POINTS]

Motion control setting and GX LogViewer can be installed from the MELSOFT GX Works3 DVD, but they are not included in MELSOFT GX Works3 update module.
When updating MELSOFT GX Works3 by using the update module, also update Motion control setting and GX LogViewer separately.
Please contact your local sales office for the update module.

2.4.3 Profile registration

Register the profiles for the devices such as a servo amplifier and a remote I/O module to be used. (The profiles for MR-J5-G/MR-J5W_-G servo amplifiers are automatically registered when MELSOFT GX Works3 is installed.)

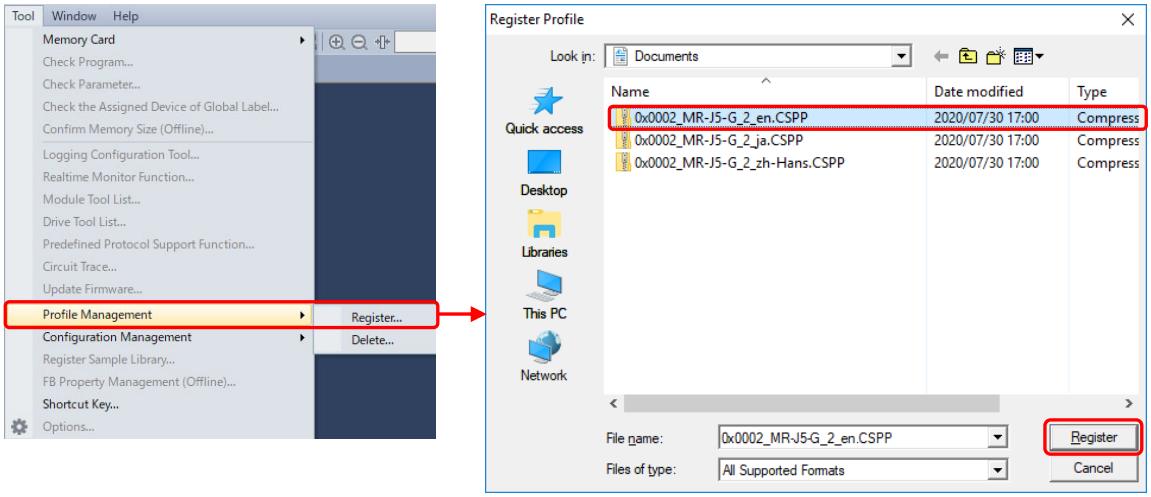
(1) Downloading the profiles

Please contact your local sales office for downloading the profiles.

(2) Installing the profiles

After unzipping the downloaded file in any location, register a CSP+ file (zip file).

1) Start MELSOFT GX Works3, click [Tool] → [Profile Management] → [Register], and display the "Register Profile" screen.
Select the file of the language to be used from the unzipped profile, and click the [Register] button.



2.4.4 MELSOFT iQ-R Series Motion Module FB library

To create programs with the PLC CPU, register the FB library with the following procedure.

(1) Downloading the Motion module FB

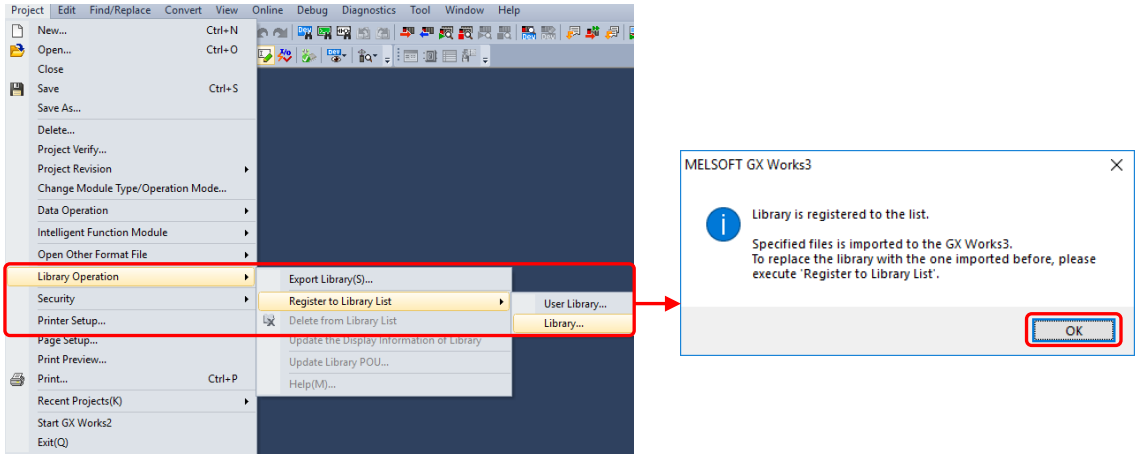
Please contact your local sales office for downloading the Motion module FBs.

(2) Registration procedure of the Motion module FB library

1) Start MELSOFT GX Works3.

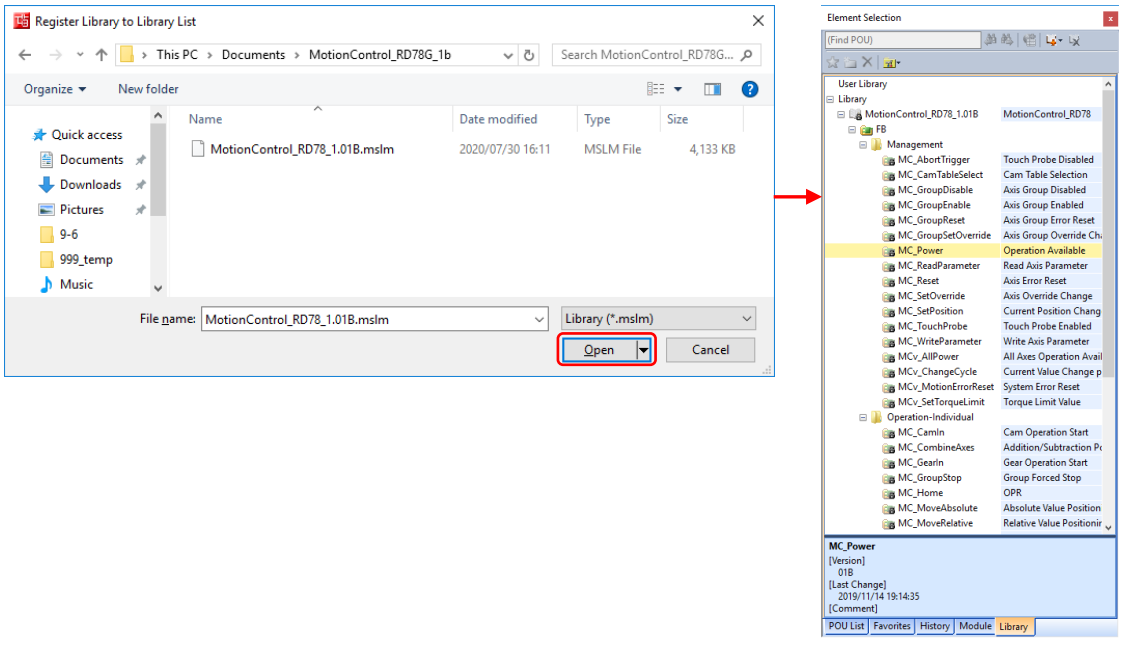
2) Open any project.

Select [Project] → [Library Operation] → [Register to Library List] → [Library], and then click the [OK] button on the confirmation screen.



3) On the "Register Library to Library List" screen, select [MotionControl_****.mslm], and click the [Open] button.

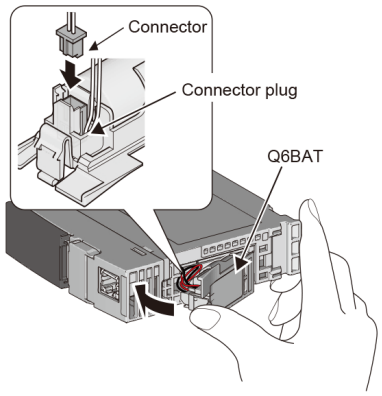
Make sure that the Motion module FB is registered in the [Library] tab of the element selection window.



2.5 Module Installation

(1) Installing a battery

The Q6BAT installed in the CPU module is shipped with the connector disconnected.
Connect the connector with the following procedure.



1) Open the battery cover located on the bottom of the CPU module.
2) Check that the Q6BAT is correctly installed.
3) Check the direction of the connector attached to the Q6BAT, and insert it into the connector pins on the cover. Firmly push the connector all the way.
4) Close the bottom cover.

(2) Installing the modules

Install each module to the main base unit.

2.6 Wiring and Cable Connection

The following shows a wiring and cable connection example for the Motion module and the servo amplifier.
The listed wire size is applicable when the MR-J5-10G servo amplifier is used.
If the servo amplifiers with other capacities are used, refer to the relevant user's manual.

(1) Wiring the power supply module

The following shows an applicable size of the power and grounding wires for the power supply module. To reduce noise such as lightning surge, connect an isolation transformer.

Item	Applicable wire size
Power wires	AWG 18 to AWG 14
Grounding wires	AWG 18 to AWG 14

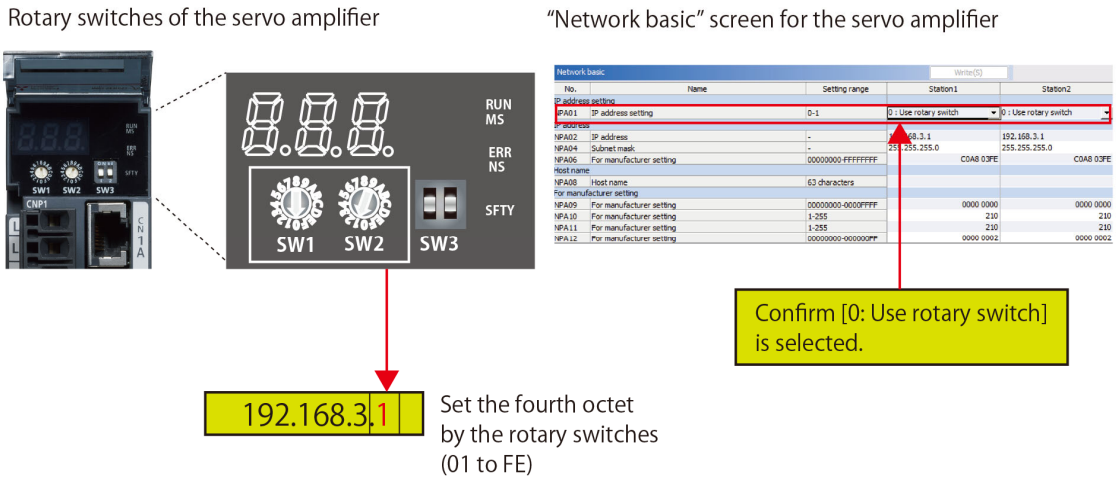
(2) Wiring the power supply of the servo amplifier

Wire the control circuit power supply (L11 and L21) and the main circuit power supply (L1, L2, and L3) of the servo amplifier.

Item	Applicable wire size
Control circuit power supply (L11 and L21)	1.25 mm^2 to 2 mm^2 (AWG 16 to AWG 14)
Main circuit power supply (L1, L2, and L3)	2 mm^2 (AWG 14)
Grounding wires	AWG 18 to AWG 14

(3) Rotary switch setting of the servo amplifier

The settings of the rotary switches (SW1 and SW2) of the servo amplifier correspond to the fourth octet of the IP address in the "CC-Link IE TSN Configuration" window.



(4) Connecting the cables

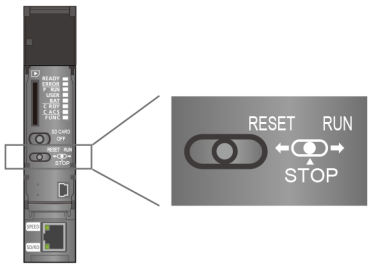
Connect the Ethernet cable and the cable for the rotary servo motor.
Connect the USB cable between the personal computer and the CPU module.

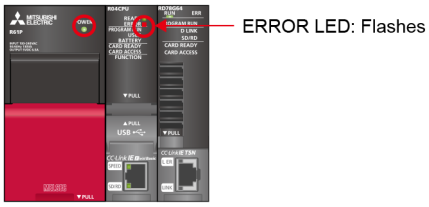
(5) Power-on of the system

1) Check the wiring of the power supply module.

2) Make sure that the PLC CPU module is in the STOP status.

3) Turn ON the power supply.





Confirm that the LEDs of the modules are ON.

- Power supply module: LED (green) ON
- PLC CPU module: READY LED (green) ON

  The ERROR LED flashes red when parameters and programs are not written to the CPU module, but no immediate error is occurring.

  After the parameters and programs are written and the power supply is cycled, the ERROR LED turns OFF.
- Motion module: RUN LED (green) ON

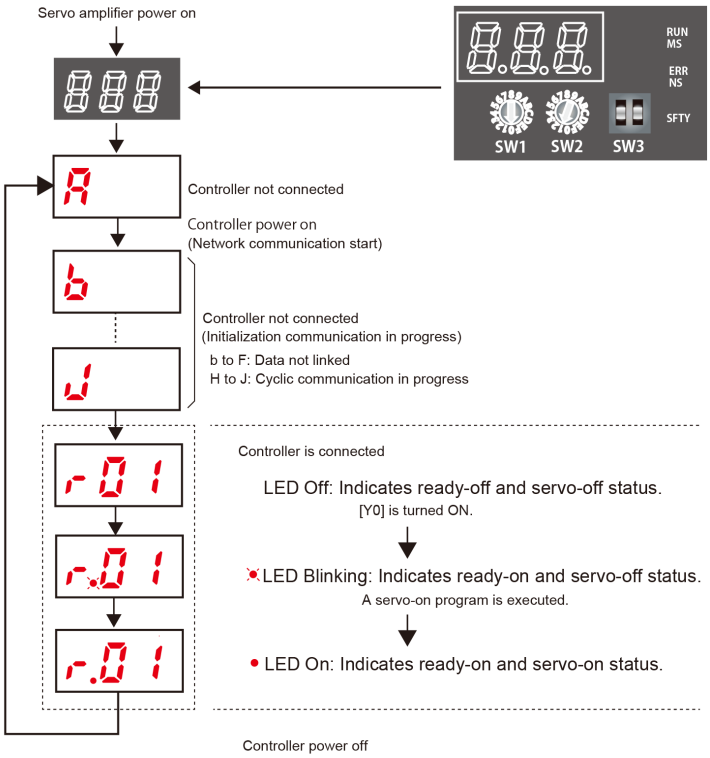
  When the communication with the slave station is performed normally after the parameters and programs are written and the power supply is cycled, the ERR LED turns off.

(6) Power-on of the servo amplifier

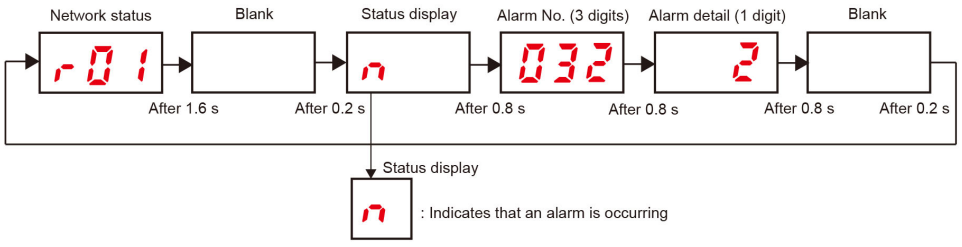
Check the wiring of the servo amplifier, and turn ON the control circuit power supply of the servo amplifier.

The communication status of the servo amplifier and the Motion module can be checked on the servo amplifier display.

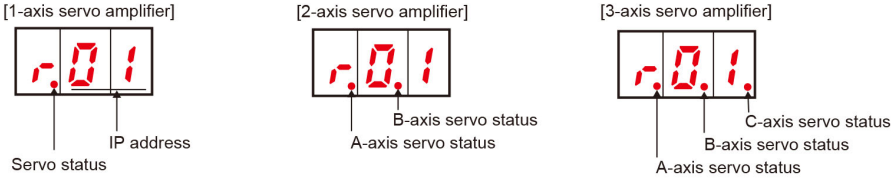
(a) The 7-segment LED display of the servo amplifier (normal status)



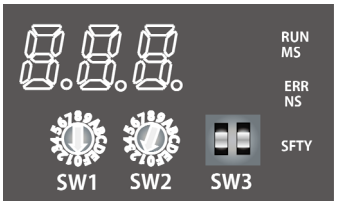
(b) Alarm occurrence of a 1-axis servo amplifier



(c) 7-segment LED display during a network connection



(7) Status LEDs

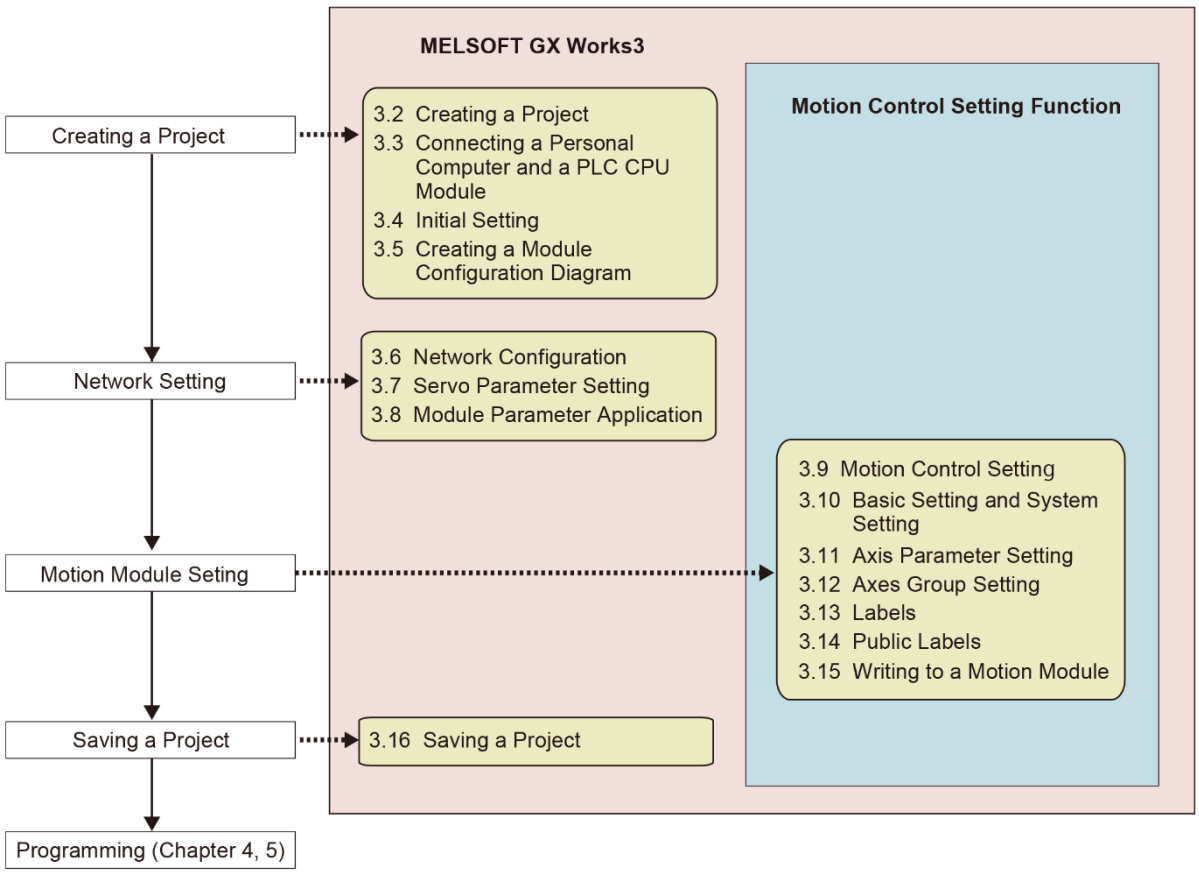


Name	LED color	Description
RUN MS	Green	OFF: Indicates that an alarm is occurring. ON: Indicates that the servo amplifier is ON.
ERR NS	Red	OFF: Indicates that no alarm or warning is occurring. Blinking: Indicates that a warning is occurring. ON: Indicates that an alarm is occurring.
SFTY	Green	OFF: Indicates that the functional safety cannot be activated. ON: Indicates that the functional safety can be activated.

3. PARAMETER SETTING

3.1 Parameter Setting Procedure

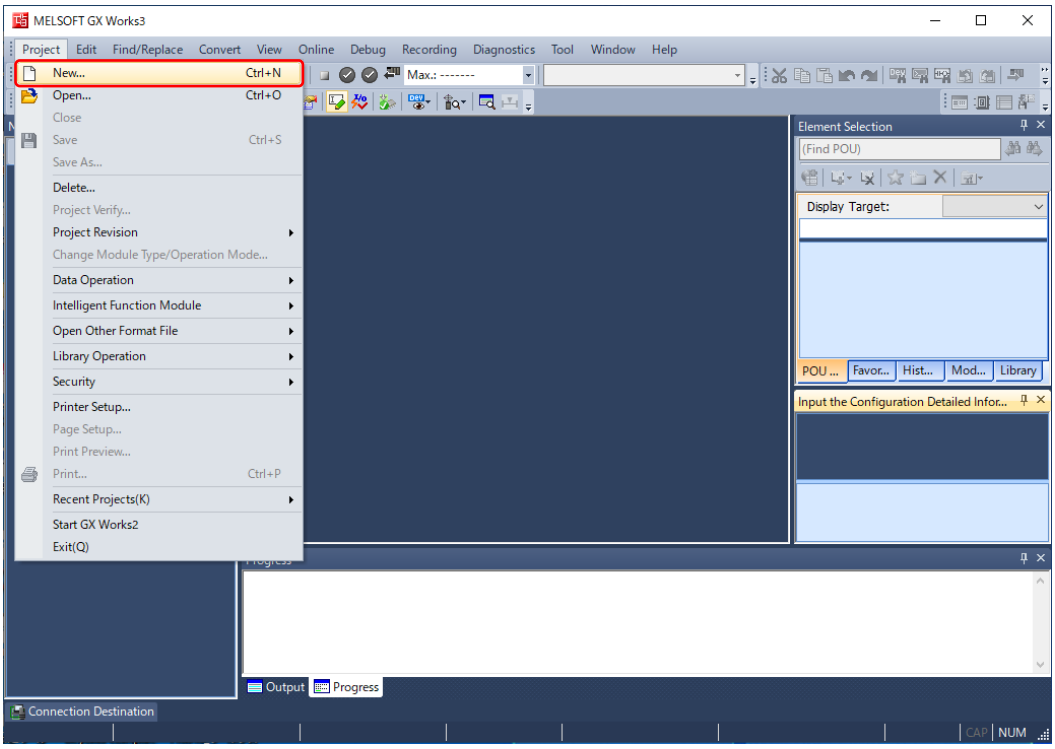
The following diagram shows the flow of the parameter setting procedure.



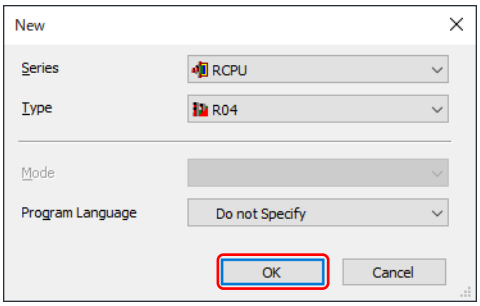
3.2 Creating a Project

Create a new project.

1) Start MELSOFT GX Works3, and select [Project] → [NEW].

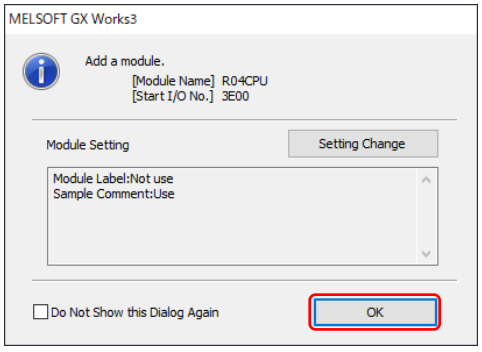


2) Select the series, the type, and the program language, and click the [OK] button.



[Setting example]
Series: RCPU
Type: Select the PLC CPU to be used
Program language: Select the language to be used

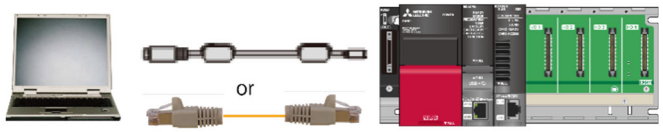
3) Read the displayed message, and click the [OK] button.



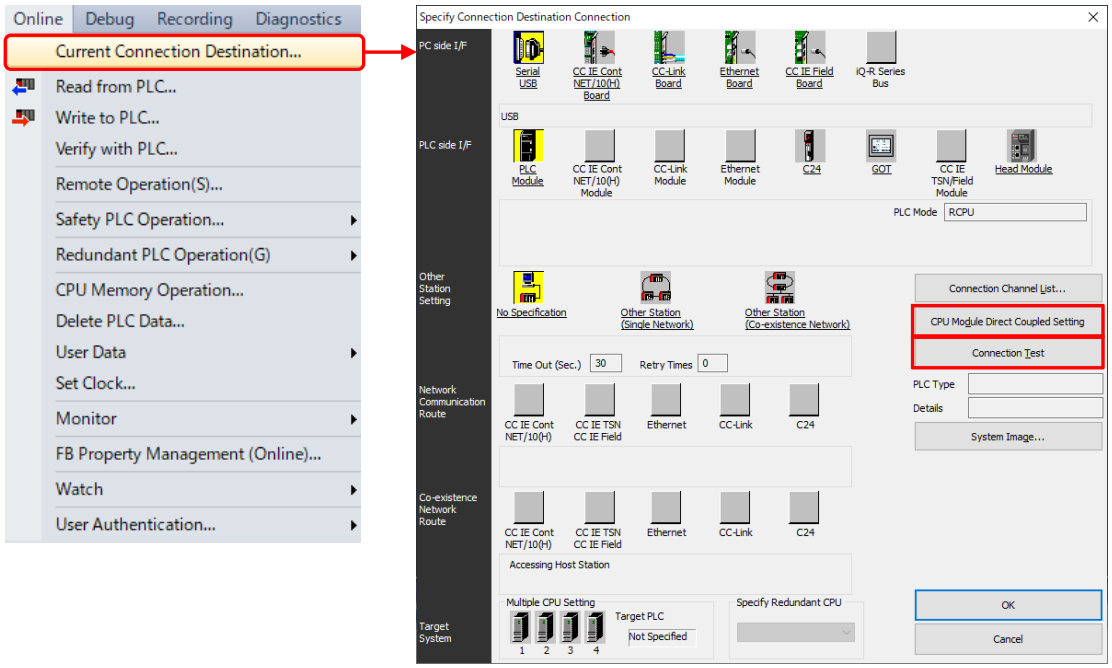
3.3 Connecting a Personal Computer and a PLC CPU Module

Check the connection between a personal computer and a PLC CPU module.

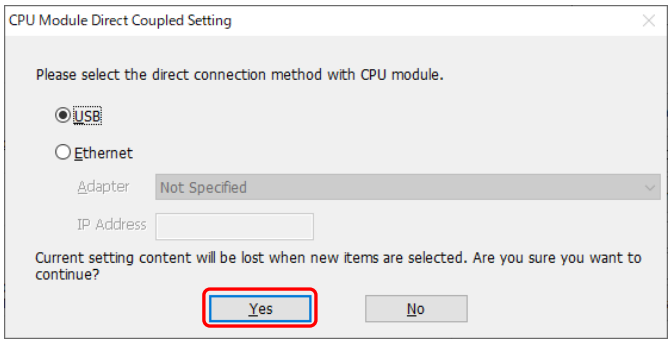
1) Connect a personal computer and a PLC CPU module with a USB or Ethernet cable.



2) Select [Online] → [Current Connection Destination] to open the "Specify Connection Destination" screen.
Click the [CPU Module Direct Coupled Setting] button, and set the connection destination for the personal computer.

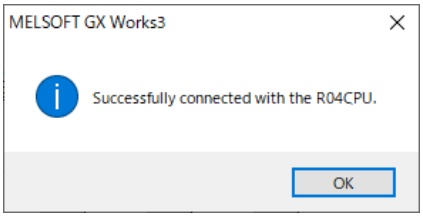


3) Select the connection method to be used, and click the [Yes] button to return to the "Specify Connection Destination" screen.



When selecting "Ethernet", specify the adapter.

4) Click the [Connection Test] button on the "Specify Connection Destination" screen to perform a connection test with the personal computer.

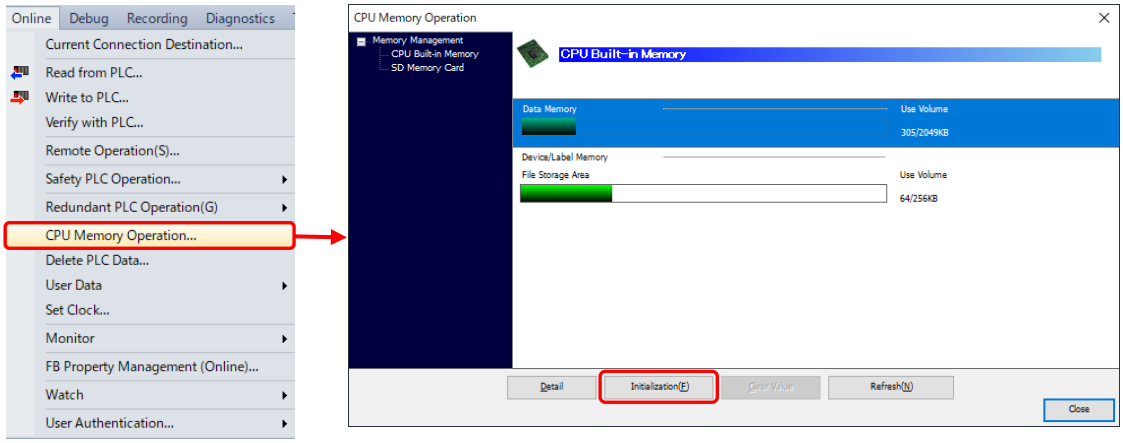


3.4 Initial Setting

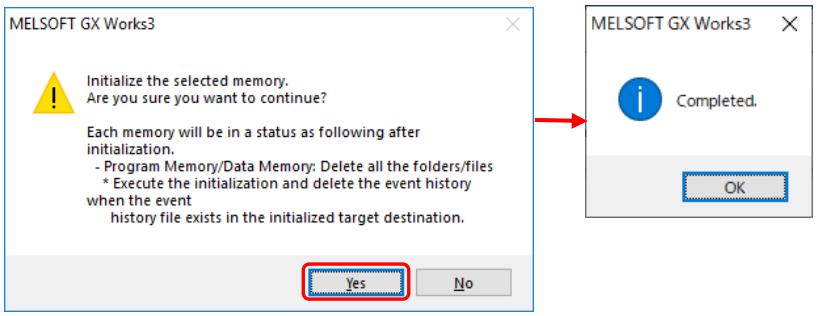
3.4.1 Initializing a PLC CPU module

The battery in the PLC CPU module is shipped with the connector disconnected.
Therefore, the data in the memory is unstable. Initialize the PLC CPU module to clear the memory.

1) Select [Online] → [CPU Memory Operation]. Click the [Initialization] button on the [CPU Memory Operation] screen.



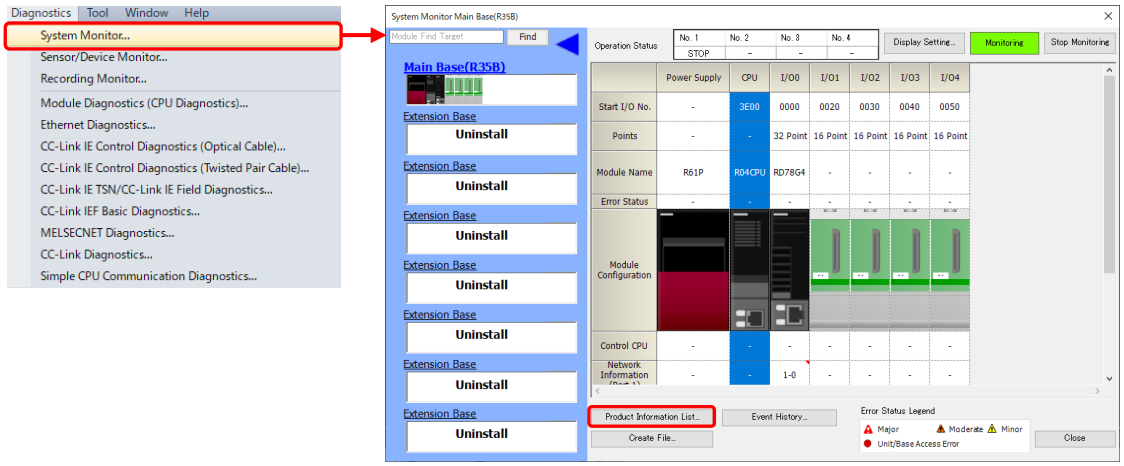
2) Click the [Yes] button on the confirmation screen to start initialization.



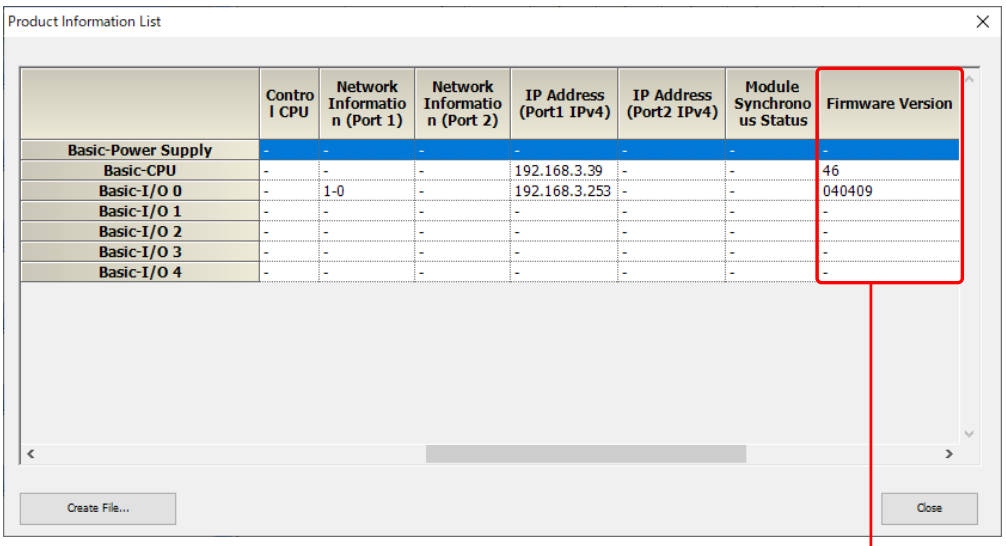
3.4.2 Checking firmware version

Check the firmware version of the PLC CPU module and the Motion module.

1) Select [Diagnostics] → [System Monitor] to open the "System Monitor Main Base (R35B)" screen. Click the [Product Information List] button to open the "Product Information List" screen.



2) Check the firmware version.



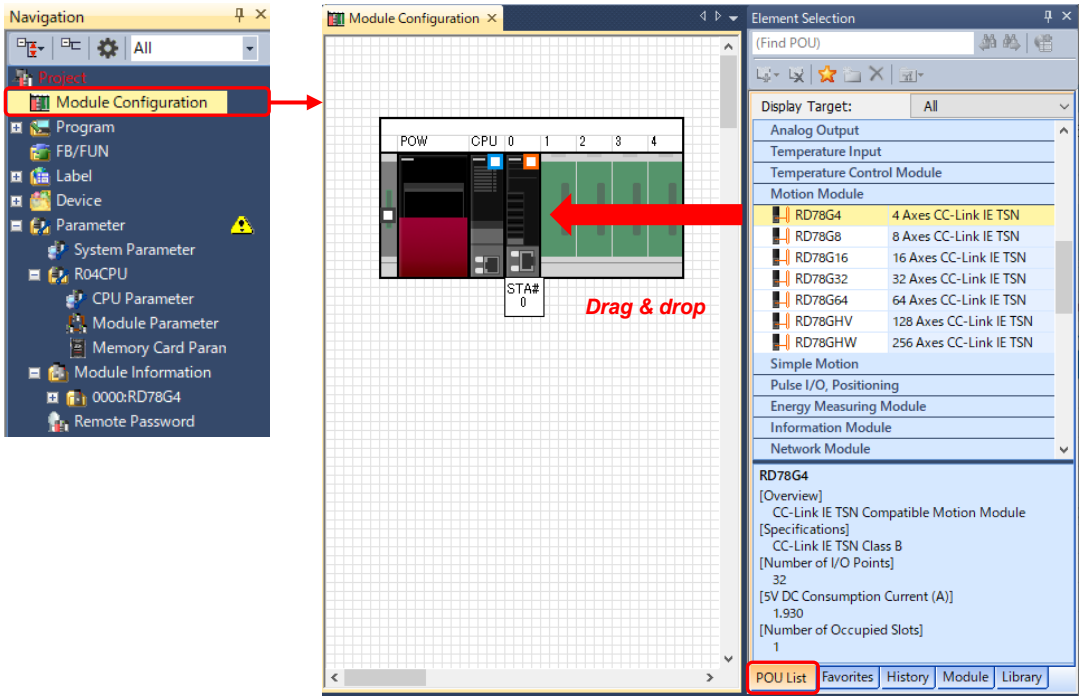
[Motion module version designation]

040409
- Basic system software
- Boot software
- Nertwork boot software

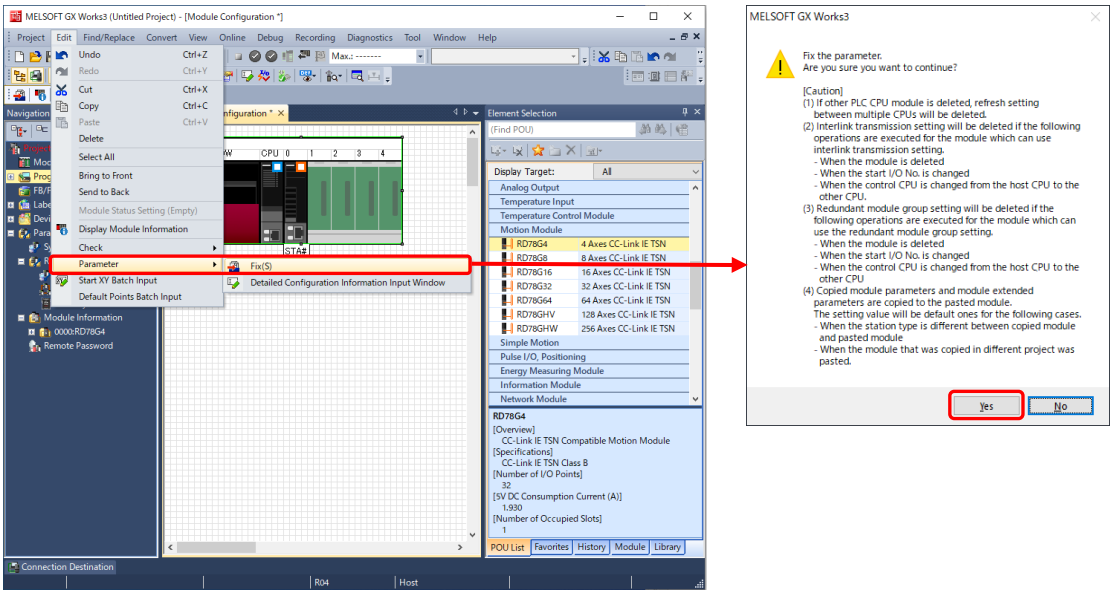
3.5 Creating a Module Configuration Diagram

Create a module configuration diagram.

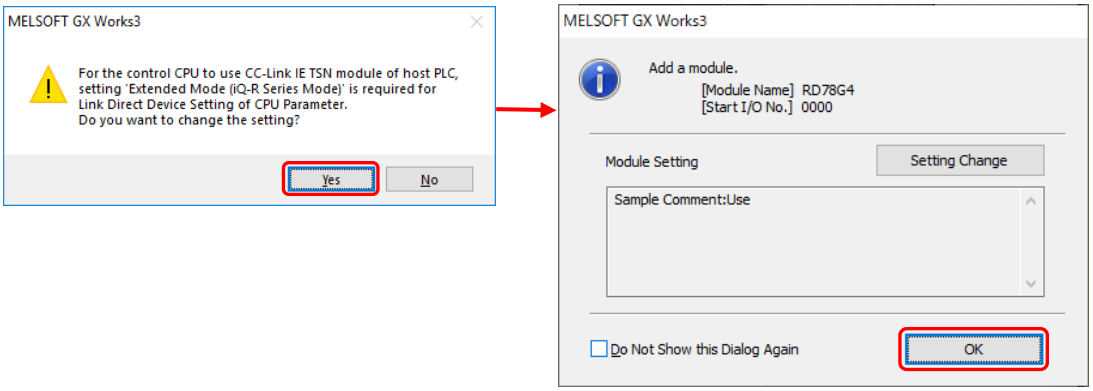
1) Double-click "Module Configuration" in the navigation window to open the "Module Configuration" window. Click the [POU List] tab in the element selection window. Select the modules to be used (Main base unit, Power supply module, PLC CPU module, and Motion module), and drag and drop them to the module configuration diagram.



2) Select [Edit] → [Parameter] → [Fix], and click the [Yes] button on the confirmation screen.



3) Click the [Yes] button on the confirmation screen. Read the displayed message, and click the [OK] button to fix the parameters (initial values) for each module.

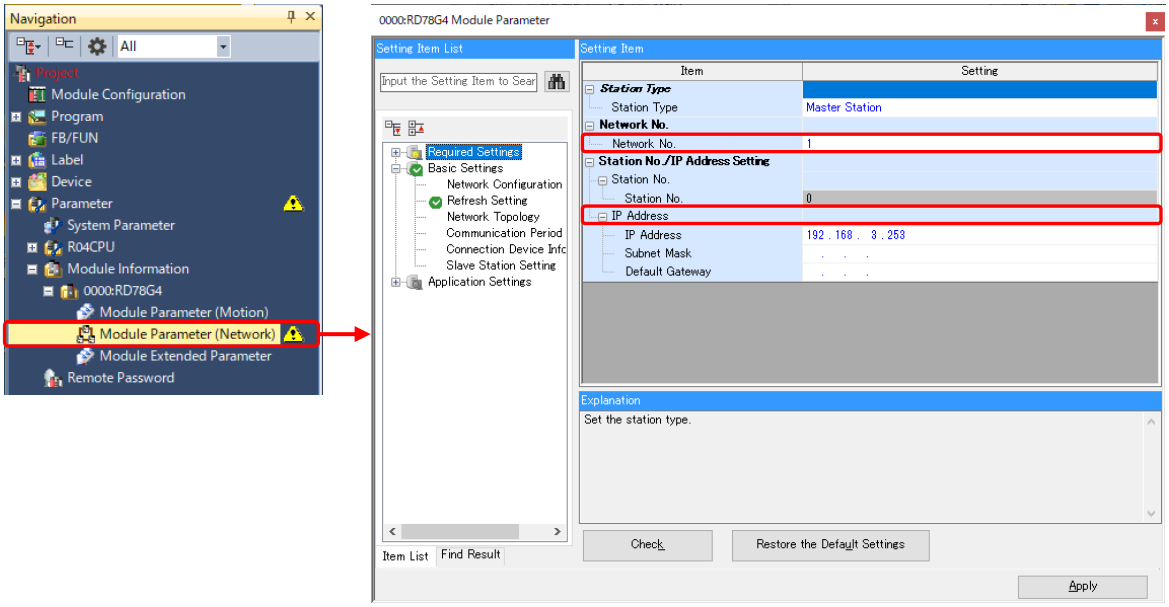


3.6 Network Configuration

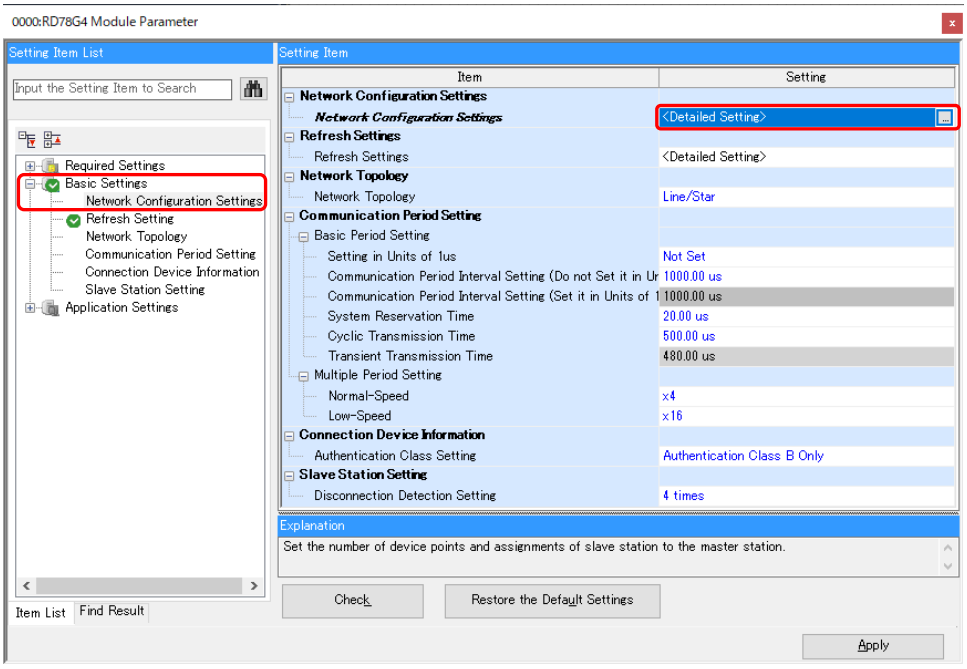
Set slave devices, such as servo amplifiers, to be connected to CC-Link IE TSN.

3.6.1 Setting a network configuration

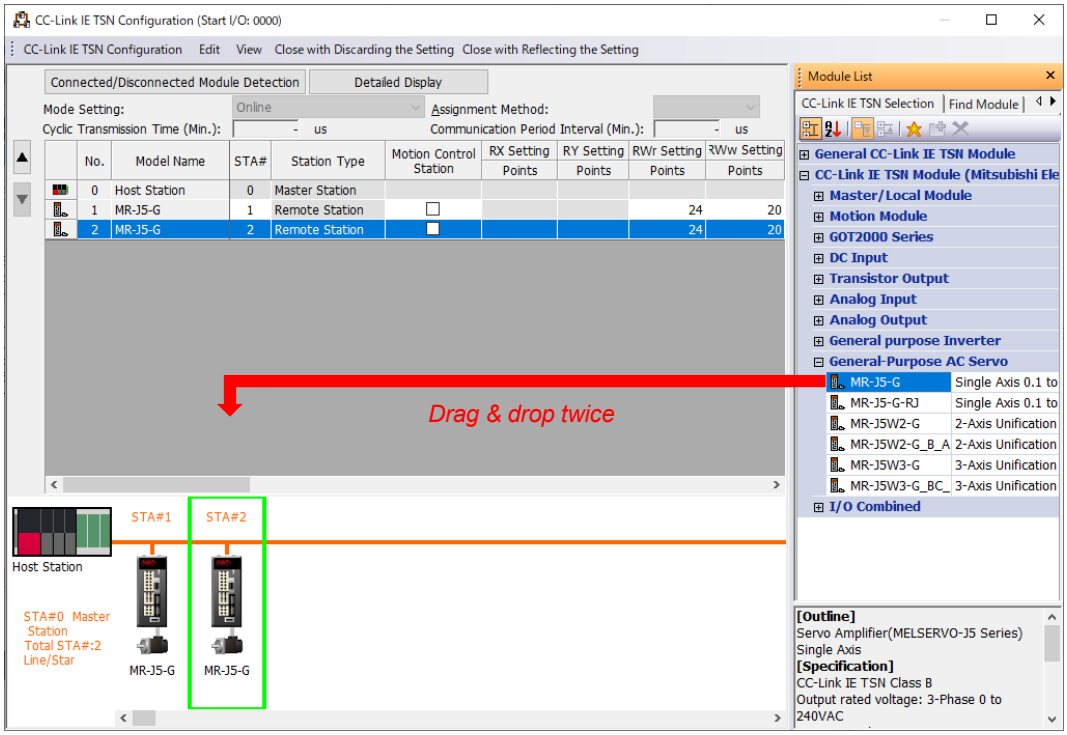
1) Double-click "Module Parameter (Network)" of the Motion module in the navigation window. Set "Required Settings" → "Network No." and "IP Address" in the parameter editor (Module Parameter).
The example in this document uses the initial value. (Network No.: 1, IP Address: 192.168.3.253)



2) Double-click "Basic Settings" → "Network Configuration Settings" → "<Detailed Setting>" in the parameter editor (Module Parameter) to open the "CC-Link IE TSN Configuration" window.

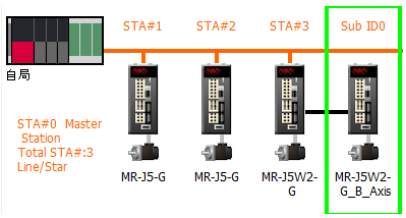


3) Select the devices to be used in the module list, and drag and drop them.
The initial values are displayed in the station No. and the I/O points. The IP address is assigned automatically based on the master station setting, the master station No., and the order the module is dropped.
The setting example in this document uses two units of the servo amplifiers. Drag and drop MR-J5-G twice.



(Note): When adding devices which are not listed in the module list, register the profiles. (Refer to Section 2.4.3.)

[POINTS]

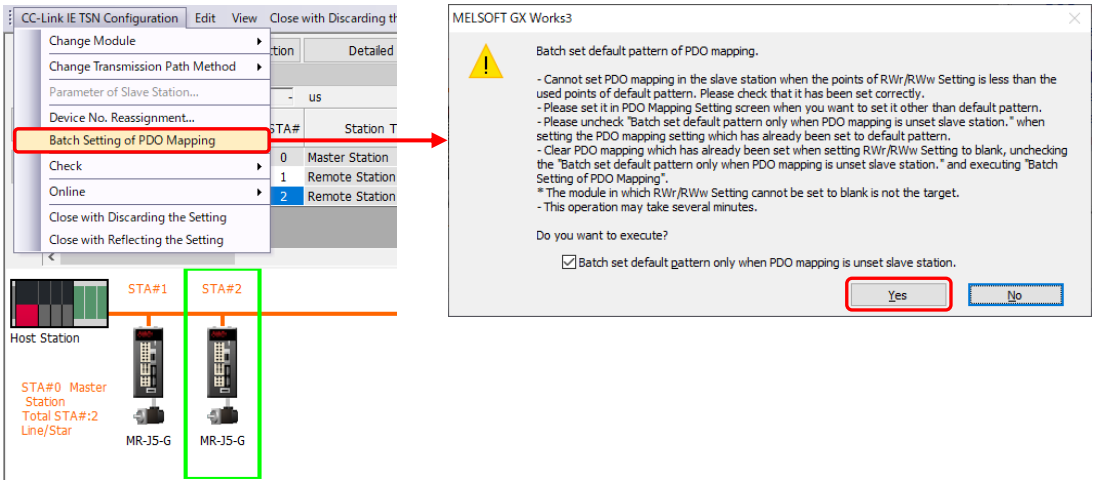


When the user drags and drops a multi-axis servo amplifier (MR-J5W_-G), the B-axis and C-axis are set together with the A-axis.

3.6.2 PDO mapping

Set default data of the PDO mapping in a batch.

1) Select [CC-Link IE TSN Configuration] → [Batch Setting of PDO Mapping]. Click the [Yes] button on the confirmation screen to set the default pattern of the PDO mapping in a batch.



[POINTS]

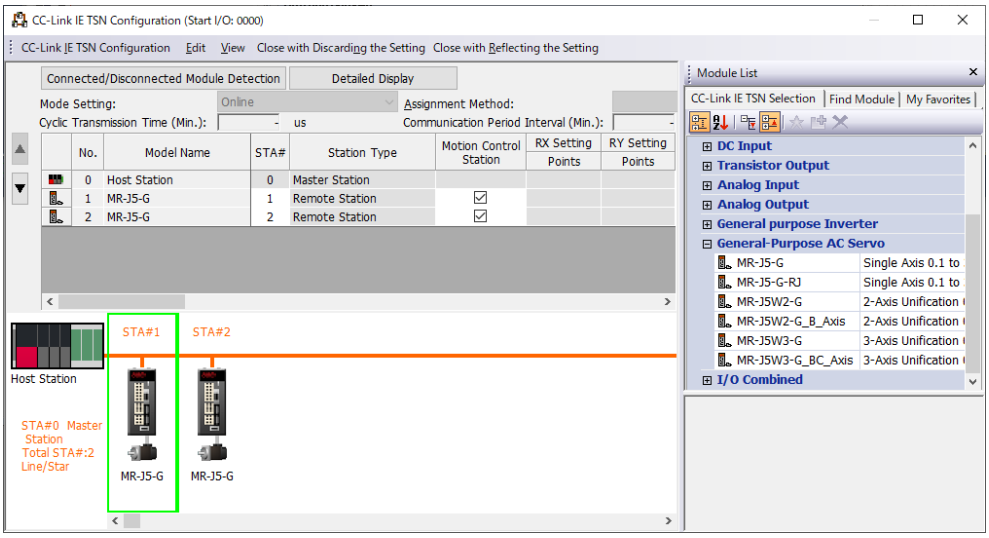
PDO is an abbreviation for Process Data Object.

The PDO communication is equivalent to the conventional CC-Link cyclic communication.

The PDO mapping performs mapping (associating) of the data (object) which is sent/received between the controller and the slaves in cyclic communication (PDO communication).

3.6.3 Setting a motion control station

Select the checkboxes of "Motion Control Station" of the modules controlled by the Motion module such as servo amplifiers and I/O modules.



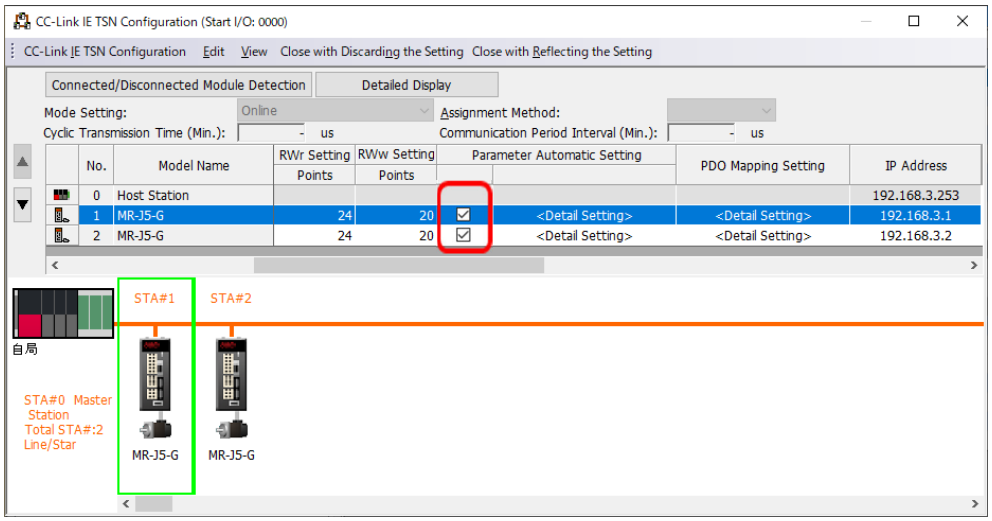
3.7 Servo Parameter Setting

There are two methods for writing parameters to servo amplifiers.

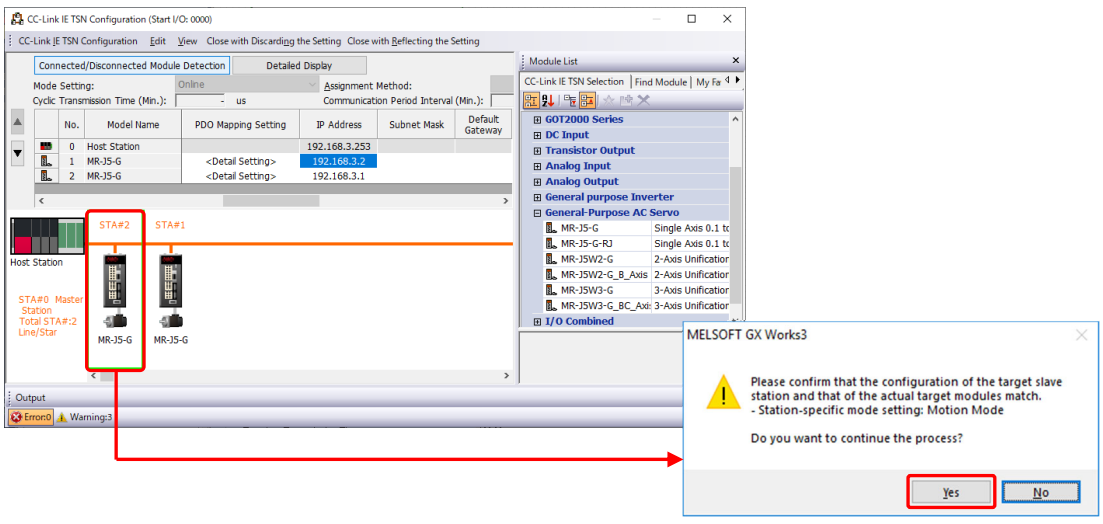
Writing method	Advantage	Disadvantage
Writing to a PLC CPU by MELSOFT GX Works3	• Batch-management of parameters • Setting parameters without connecting servo amplifiers	• Taking time at the initial communication
Writing to servo amplifiers by MR Configurator2	• Quick startup of servo amplifiers since transmission of parameters is not necessary at power-on	• Necessary to write parameters to each servo amplifier

The following shows the procedure for writing servo parameters to a PLC CPU by MELSOFT GX Works3.

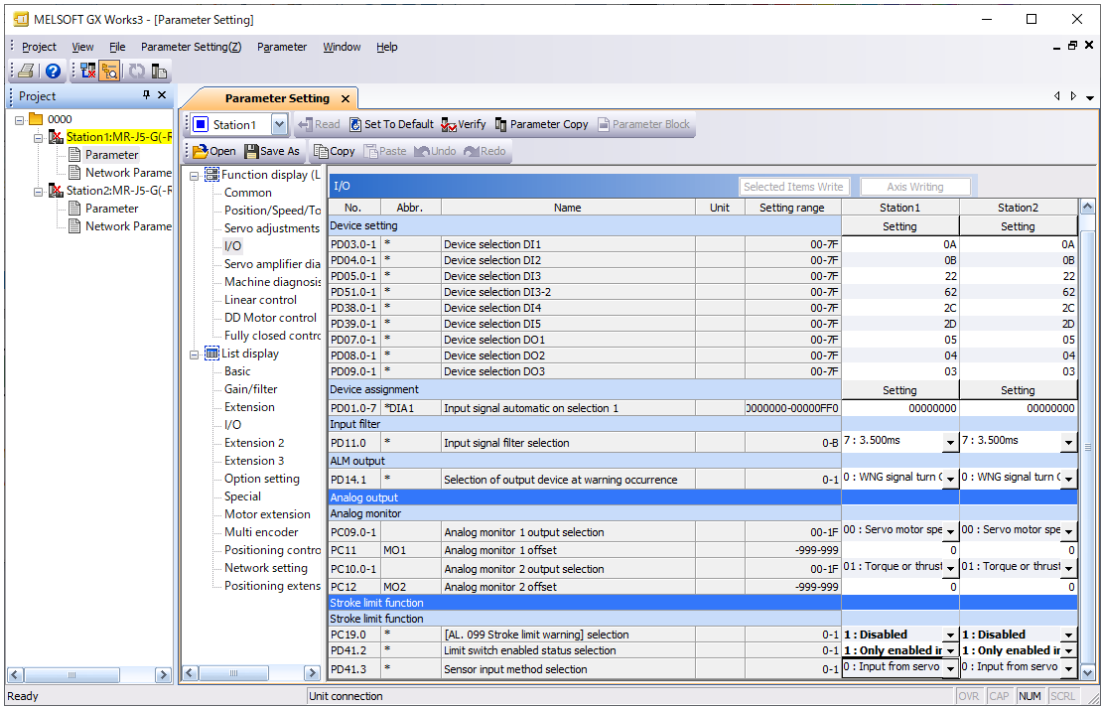
1) Select the checkboxes of "Parameter Automatic Setting" in the "CC-Link IE TSN Configuration" window to write parameters to the slave stations from the master station at the initial communication.



2) Double-click the illustration of the servo amplifier to be set, and click the [Yes] button on the confirmation screen.



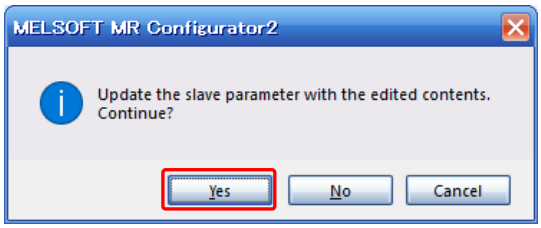
3) Set servo parameters on the "Parameter Setting" screen.



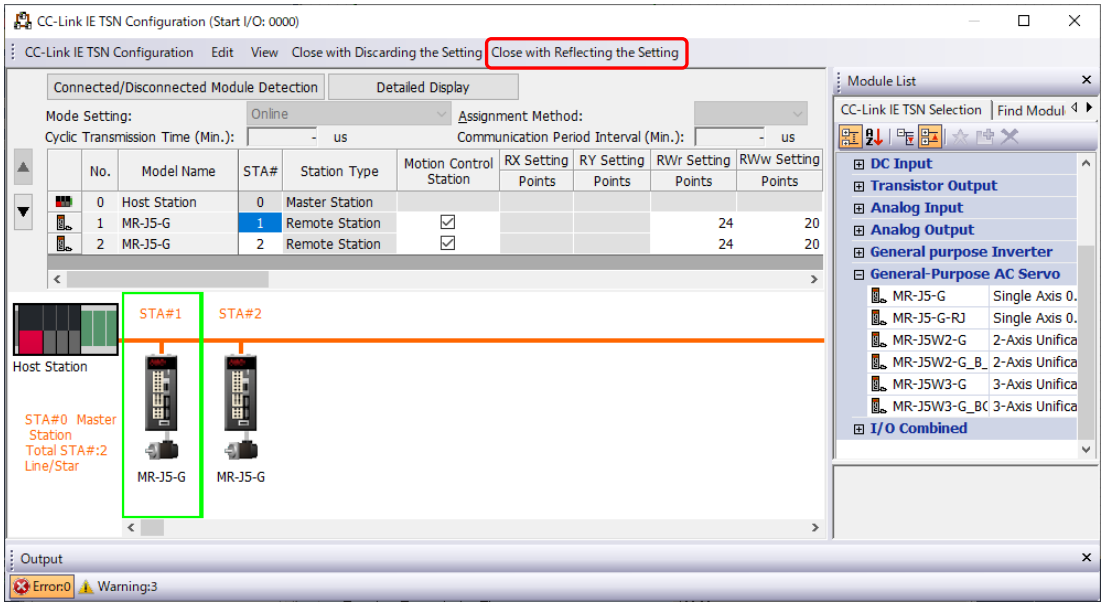
The example in this document changes the following parameters.

Operation	No.	Item	Setting value
[Common] → [Basic] → [Forced stop]	PA04.2	Servo forced stop selection	0: Enabled → 1: Disabled
[I/O] → [Stroke limit function] → [Stroke limit function]	PC19.0	[AL. 099 Stroke limit warning] selection	0: Enabled → 1: Disabled
	PD41.2	Limit switch enabled status selection	0: Always enabled → 1: Only enabled in home position return mode

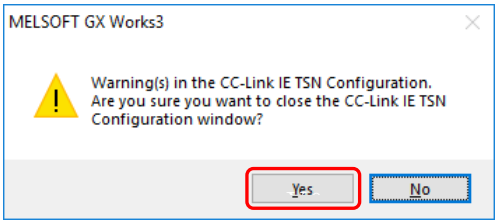
4) Click the [×] button at the upper right in the window.
Click the [Yes] button on the confirmation screen to update the parameters.



5) Click the [Close with Reflecting the Setting] button to reflect the data.

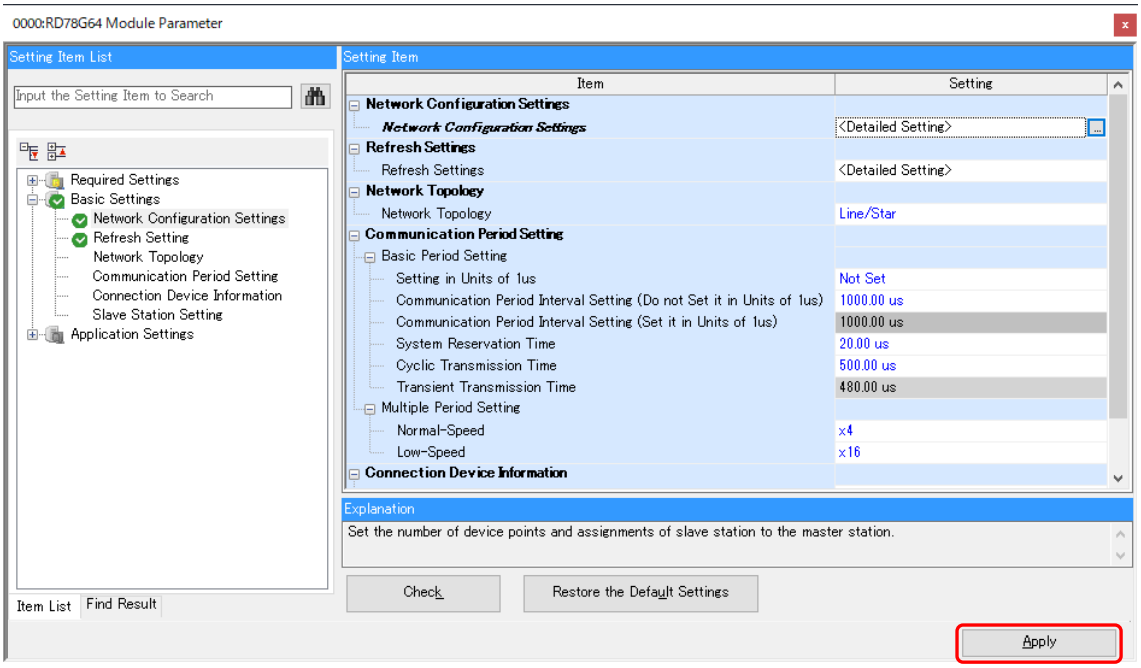


Although the warning window appears depending on the version, click the [Yes] button.



3.8 Module Parameter Application

Click the [Apply] button in the parameter editor (Module Parameter) to reflect the parameters of the Motion module.



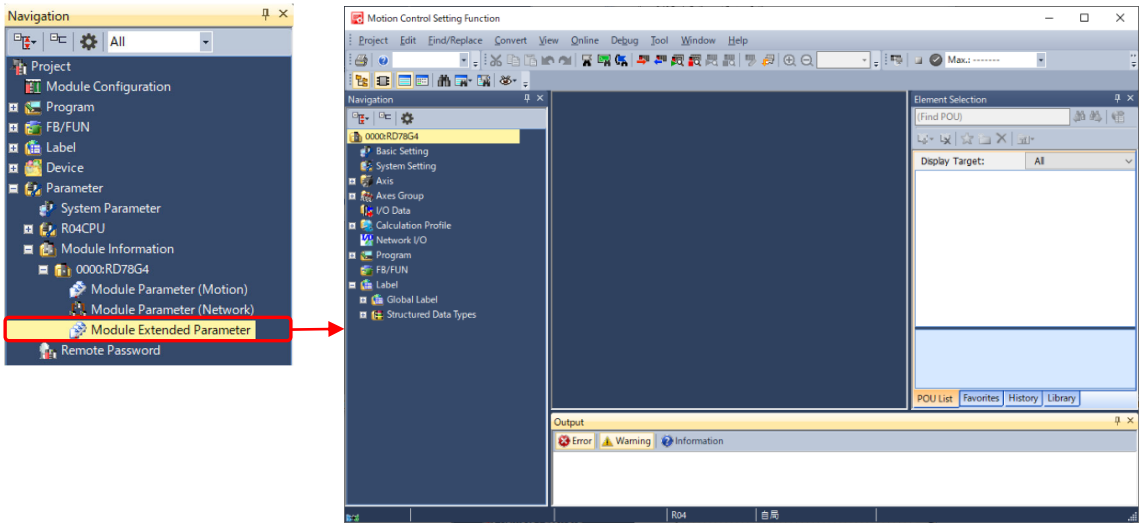
[POINTS]

Note that changes in the parameters are not reflected unless clicking the [Apply] button.

3.9 Motion Control Setting

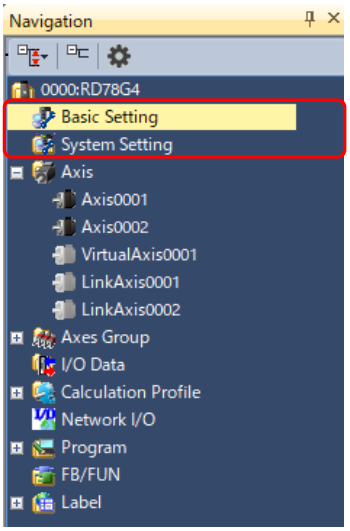
Set parameters and create programs for the Motion module on the motion control setting screen.

Double-click "Module Extended Parameter" of the Motion module in the navigation window to open the motion control setting function.
The set items are saved as a project of MELSOFT GX Works3.



3.10 Basic Setting and System Setting

In the basic setting, set each parameter for the programs. In the system setting, set the all axes forced stop and each parameter for the add-ons.
Change the parameters as necessary. (The example in this document uses the initial values.)



3.11 Axis Parameter Setting

3.11.1 Setting a station address and an axis type

Axes are classified into five types as follows.

- Real drive axis : Outputs commands using a servo amplifier connected to CC-Link IE TSN.
- Real encoder axis : Generates the current position from the output pulses of the synchronous encoder which is connected to a servo amplifier on CC-Link IE TSN.
- Virtual drive axis : Virtually generates commands.
- Virtual encoder axis : Virtually generates the current position from the variables.
- Virtual linked axis : Virtually connects FBs of the single axis synchronous control.

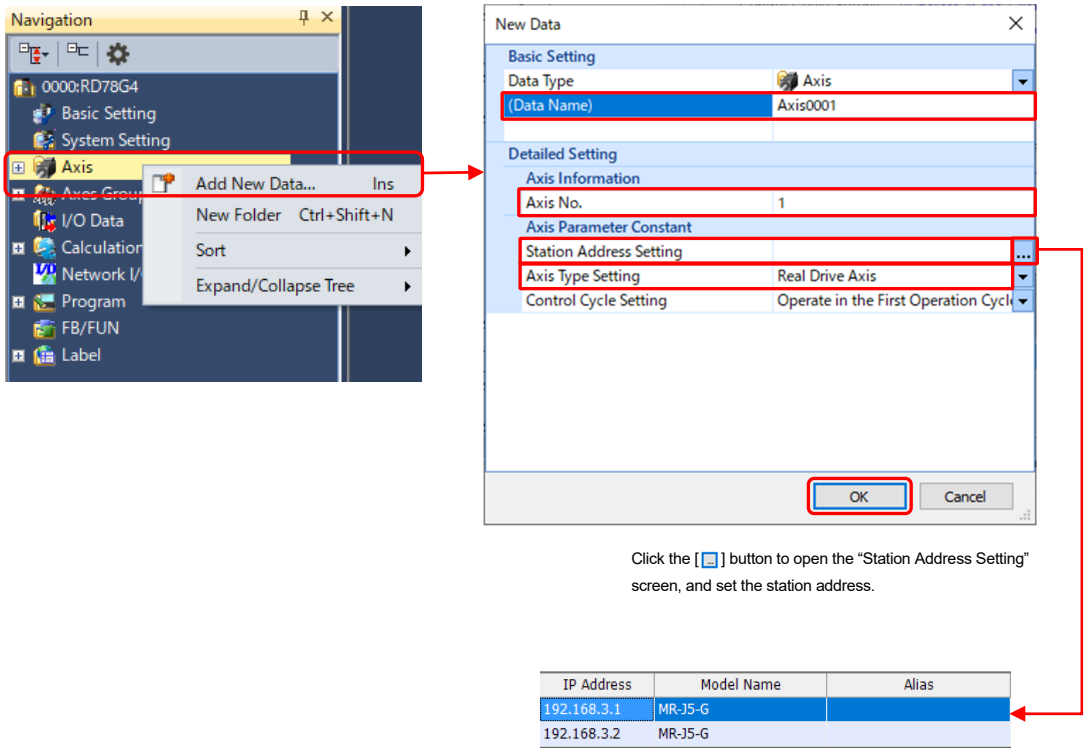
The example in this document registers two real drive axes, one virtual drive axis, and two virtual linked axes.

1) Real drive axis

Set the station address to link the axis information in the axis parameter to the servo amplifier set in the network configuration.

Right-click "Axis" in the navigation window of the motion control setting function, and select [Add New Data].

Enter the data name, the axis No., the station address setting, and the axis type setting as the following table shows, and click the [OK] button.

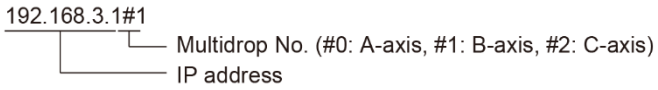


Setting item	First axis	Second axis
Data Name	Axis0001	Axis0002
Axis No.	1	2
Station Address Setting	192.168.3.1	192.168.3.2
Axis Type Setting	Real Drive Axis	Real Drive Axis

[POINTS]

For the station address setting, use the IP address set in the CC-Link IE TSN configuration window.
A multi-drop No. is shown when a station address of the second or the third axis of a multi-axis servo amplifier is specified.

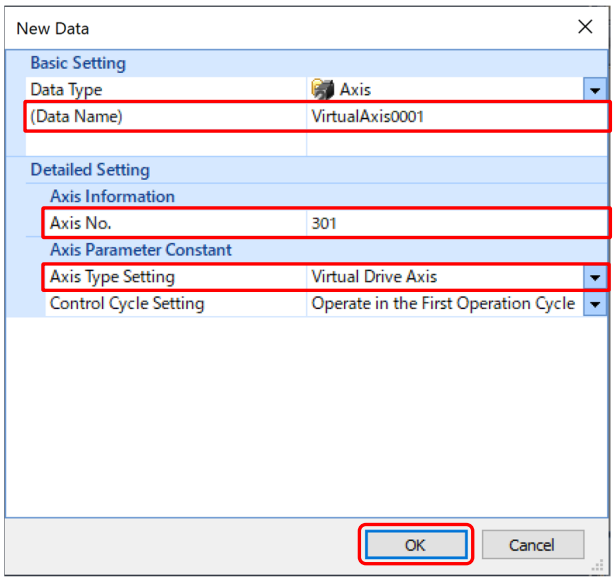
Example: For B-axis of MR-J5-W3 with an IP address of 192.168.3.1



2) Virtual drive axis

Right-click "Axis" in the navigation window of the motion control setting function, and select [Add New Data].

Enter the data name, the axis No., and the axis type setting as the following table shows, and click the [OK] button.

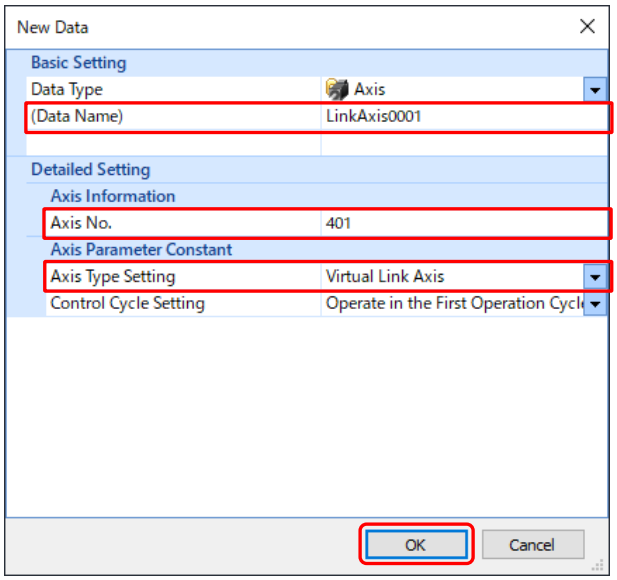


Setting item	First axis
Date Name	VirtualAxis0001
Axis No.	301
Axis Type Setting	Virtual Drive Axis

3) Virtual linked axis

Right-click "Axis" in the navigation window of the motion control setting function, and select [Add New Data].

Enter the data name, the axis No., and the axis type setting as the following table shows, and click the [OK] button.

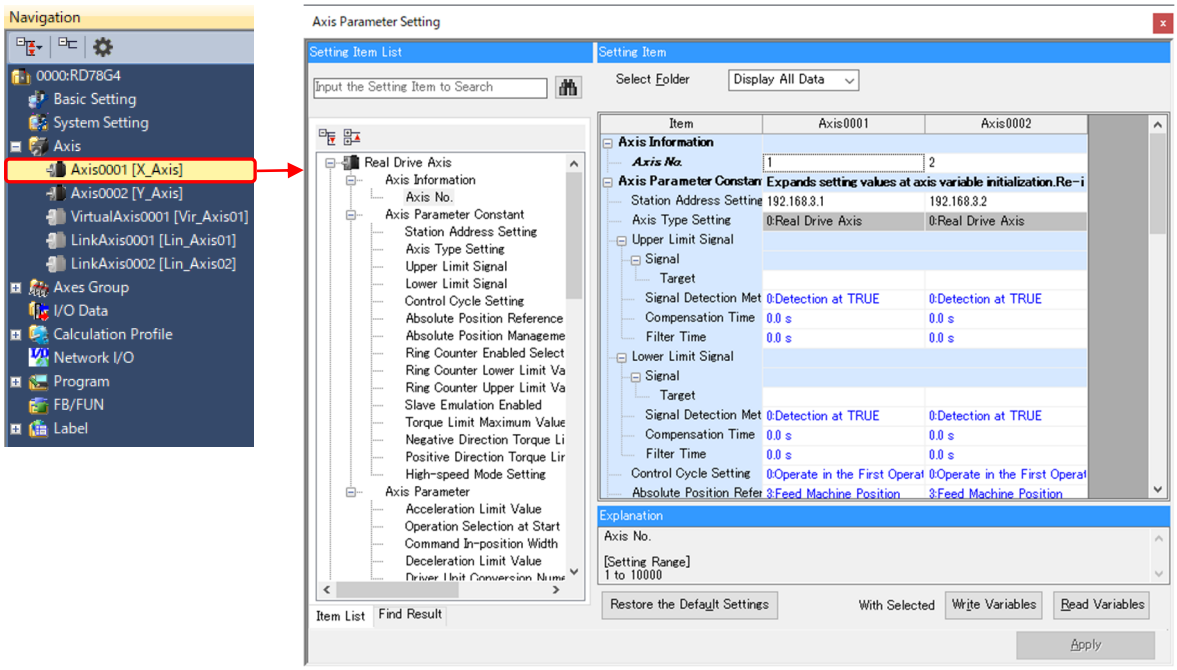


Setting item	First axis	Second axis
Data Name	LinkAxis0001	LinkAxis0002
Axis No.	401	402
Axis Type Setting	Virtual Link Axis	Virtual Link Axis

3.11.2 Setting each item

1) Real drive axis

Double-click "Axis" →"Axis0001" in the navigation window to open the "Axis Parameter Setting" window.



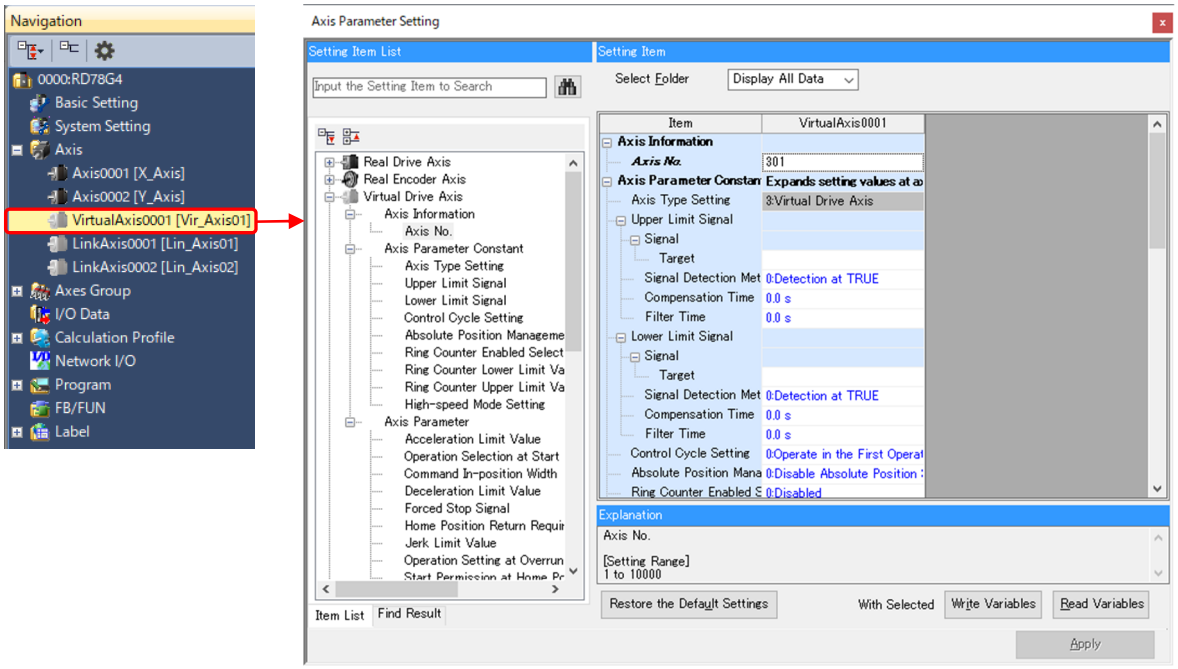
The setting example in this document is shown below.

Item		Description	Axis0001 setting value	Axis0002 setting value
Axis Parameter Constant	Station Address Setting	Set the IP address of the driver.	192.168.3.1	192.168.3.2
	Upper Limit Signal Lower Limit Signal	Set the hardware stroke limit switches at the upper/lower limit of the movable range by assigning external signals.	Initial value (Refer to Appendix 2.)	
Axis Parameter	Driver Unit Conversion Numerator	Convert units of the target position and the feedback position between the controller and the driver. (Refer to Section 3.11.3 for the setting methods.)	67108864	
	Driver Unit Conversion Denominator		5000	
	Stop Signal → Signal → Target	Set the stop signal.	[VAR]G_bStopSignalX (Note-1)	[VAR]G_bStopSignalX (Note-1)
	Forced Stop Signal, Stop Signal	Set the forced stop signal and the stop signal.	Initial value	
	Start Permission at Home Position Return Uncompleted	Set whether axis start is allowed or not for when the homing request is true.	0: Not allowed	
	Software Stroke Limit Upper Limit Value/Lower Limit Value/Target	Set the range and the target of the software stroke limit	Initial value	
	Position Command Unit	Set the position command unit and the speed command unit used for the motion control. The units used in this document are as follows: [um] for the position command unit and [um/s] for the speed command unit.	um(Note-2)	
	Speed Command Unit		U/s	

(Note-1): The stop signal is registered as a global label. Refer to Section 4.3.2 or Section 5.3.2 for setting the global labels.
(Note-2): "um" indicates micrometer.

2) Virtual drive axis

Double-click "Axis" → "VirtualAxis0001" in the navigation window to open the "Axis Parameter Setting" window.

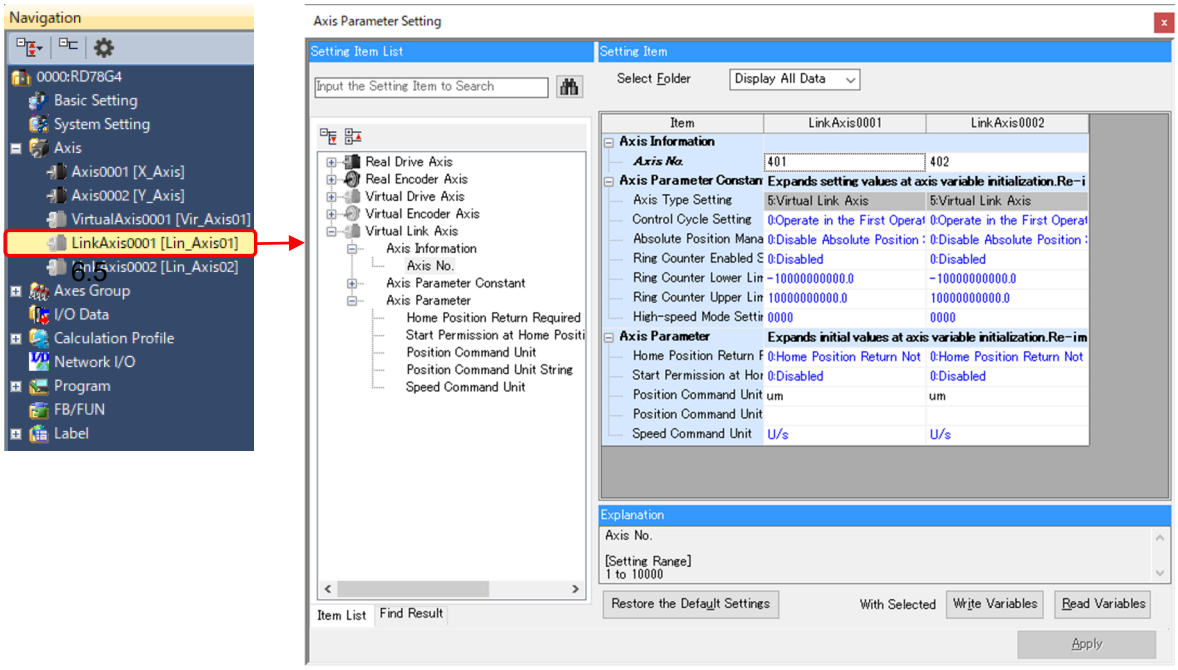


The setting example in this document is shown below.

Item		Description	Setting value
Axis Parameter	Position Command Unit	Set the position command unit and the speed command unit used for the motion control. In this example, the same unit as that of the real drive axis is set for the virtual drive axis ([um] for the position command unit and [um/s] for the speed command unit).	um
	Speed Command Unit		U/s

3) Virtual linked axis

Double-click "Axis" → "LinkAxis0001" in the navigation window to open the "Axis Parameter Setting" window.



The setting example in this document is shown below.

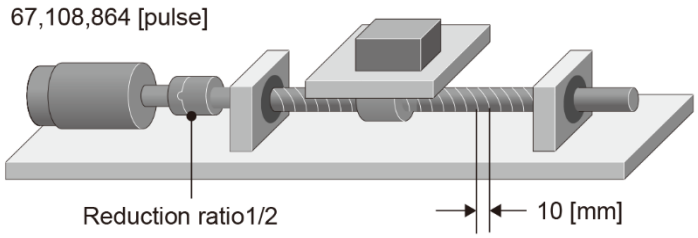
Item		Description	Setting value
Axis Parameter	Position Command Unit	Set the position command unit and the speed command unit used for the motion control. In this example, the same unit as that of the real drive axis is set for the virtual linked axes ([um] for the position command unit and [um/s] for the speed command unit).	um
	Speed Command Unit		U/s

3.11.3 Driver unit conversion (electronic gear)

A setting example of driver unit conversion numerator/driver unit conversion denominator is shown below.

(1) About an electronic gear

[Ball screw]



Servo motor encoder resolution: 67,108,864 [pulse] (26 bits)
Lead of the ball screw: 10000 [um]
Reduction ratio: 1/2 (Load side [NL]/Motor side [NM])
When the servo motor rotates twice, the ball screw of the load side rotates once.

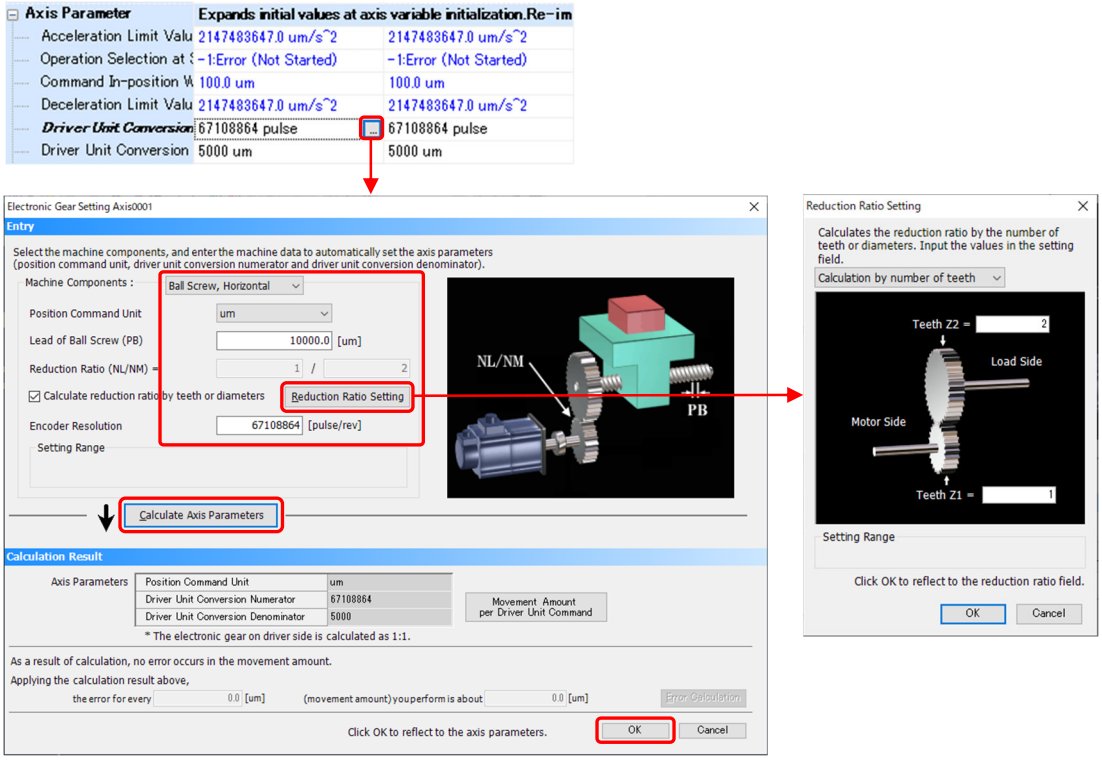
$$\frac{\textit{Driver unit conversion numerator}}{\textit{Driver unit conversion denominator}} = \frac{\textit{Number of encoder pulses}}{\textit{Movement amount} \times \textit{Reduction ratio}} = \frac{67108864}{10000 \times 1/2} = \frac{67108864}{5000}$$

Driver unit conversion numerator = Number of pulses per rotation 67,108,864
Driver unit conversion denominator = Movement amount per rotation 5000

(2) Electronic gear setting

The procedure of the electronic gear setting is shown below.

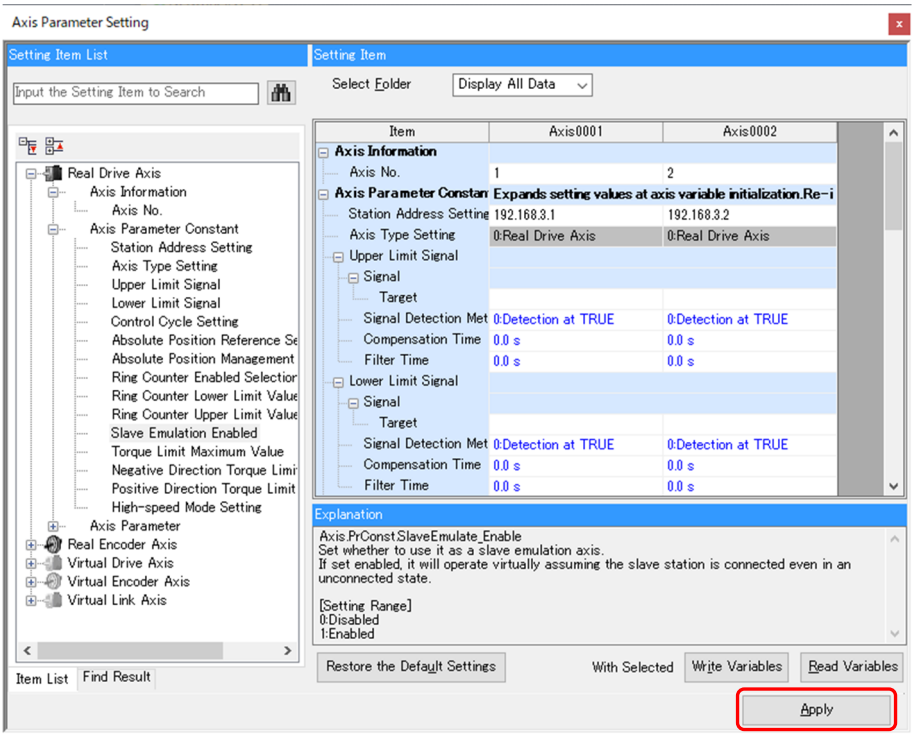
1) Click the [...] button in the "Axis Parameter" → "Driver Unit Conversion Numerator" or "Driver Unit Conversion Denominator" to open the "Electronic Gear Setting" screen.



2) Enter data of the machine components. Click the [Reduction Ratio Setting] button to open the "Reduction Ratio Setting" screen.
3) Click the [Calculate Axis Parameters] button.
4) Click the [OK] button to reflect the calculation results to the axis parameter.

3.11.4 Reflecting axis parameters

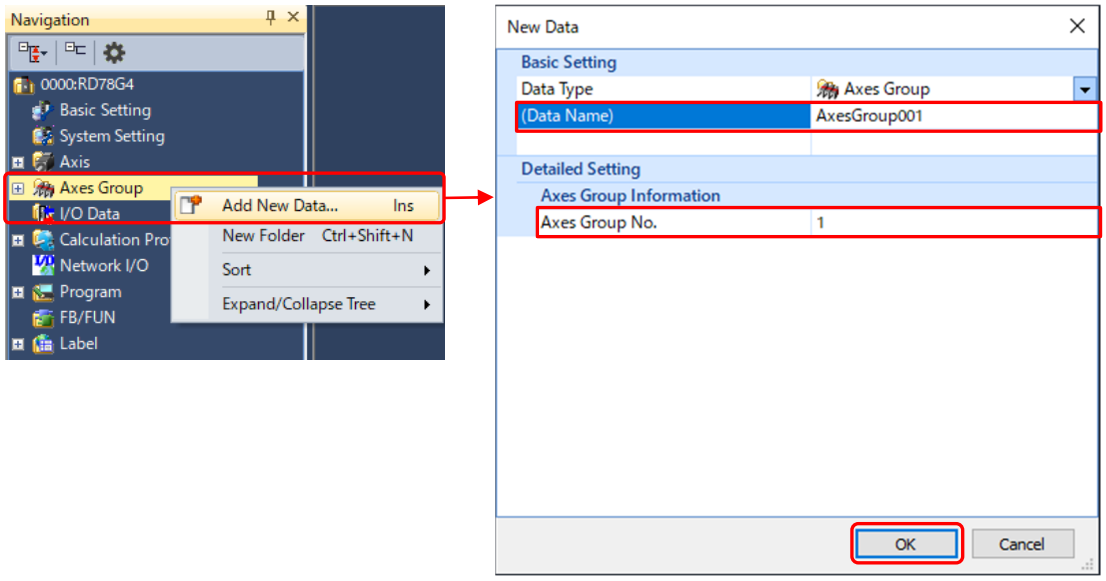
After setting each item, click the [Apply] button to fix the parameters.



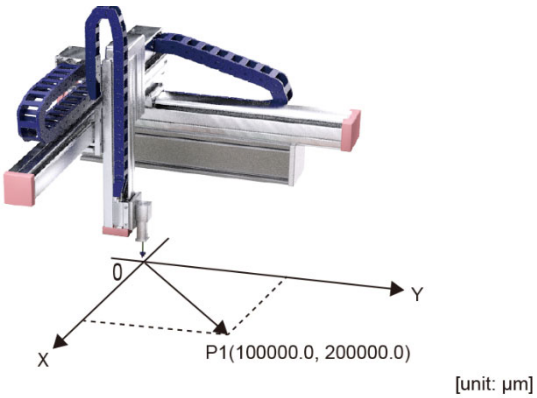
3.12 Axes Group Setting

An axes group is used for multiple axes control such as linear interpolation and circular interpolation. An axes group is not used for synchronous control.

Right-click "Axes Group" in the navigation window of the motion control setting function, and select [Add New Data]. Set the data name and the axes group No., and click the [OK] button.



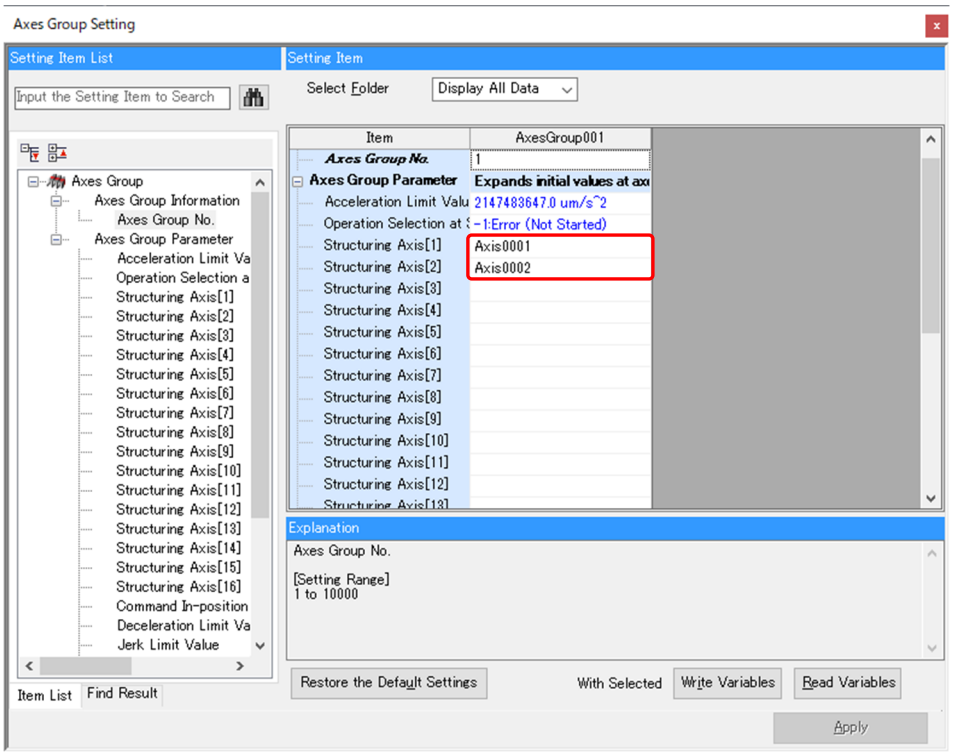
The X axis (Axis0001) and Y axis (Axis0002) of the following two-axis machine example (refer to Section 2.1) are registered in an axes group.



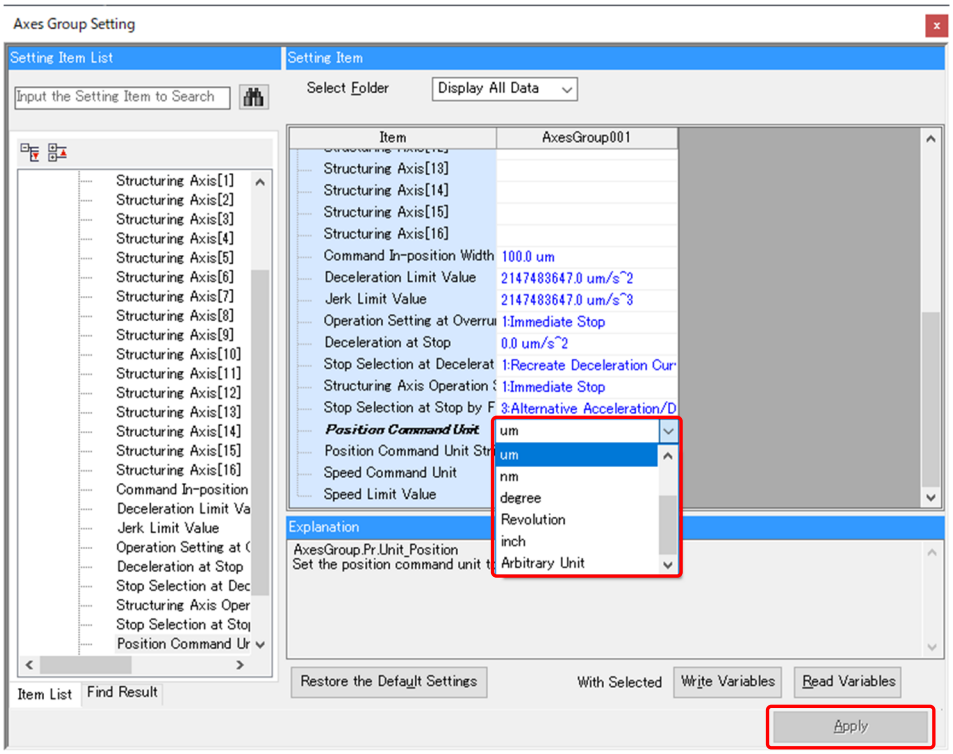
[unit: µm]

The setting example of the axes group is shown below.

[Setting example of axes group structuring axes]



[Setting example of axes group units] (Position command unit: [um], Speed command unit: [U/s])



After setting each item, click the [Apply] button to fix the axes group setting.

3.13 Labels

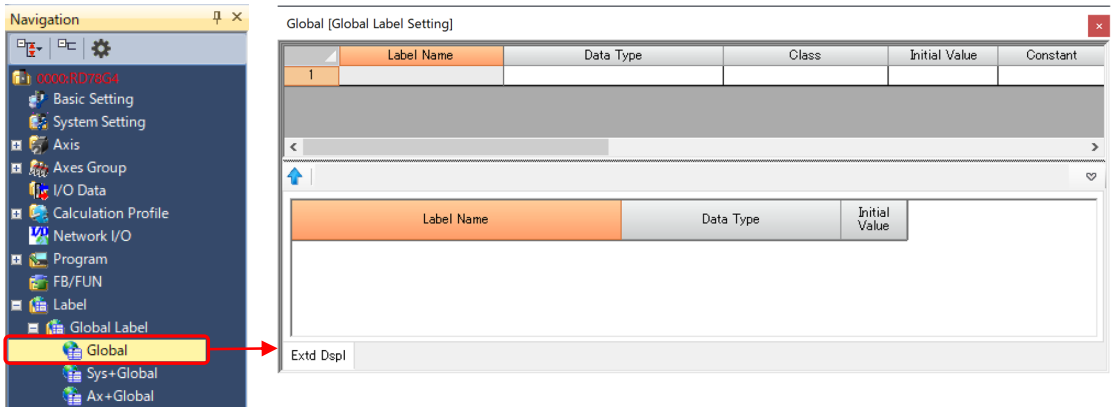
Labels are classified into local labels and global labels.
Select whether the label is registered as a local label or a global label according to your application.

Label	Description
Local label	• Labels that can be used in only one program
Global label	• Labels that can be used in all programs • Labels that can be set to public labels (Refer to Section 3.14.)

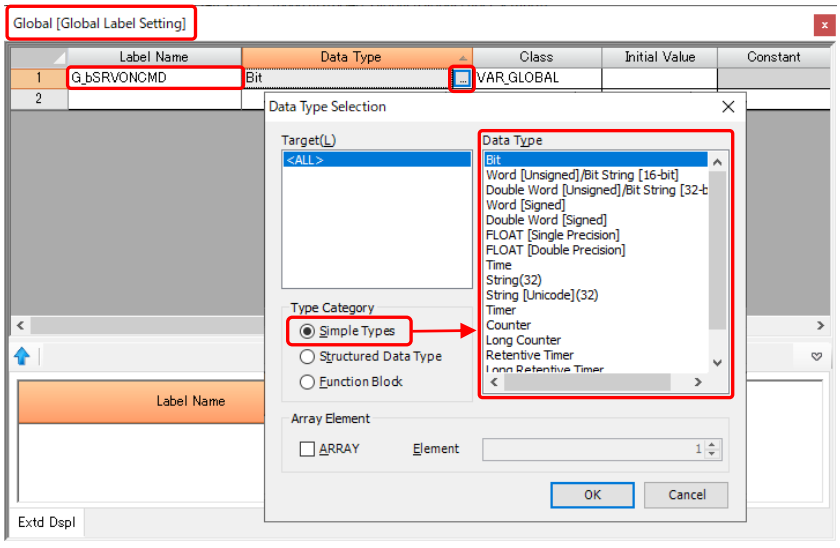
3.13.1 Creating a label

This section describes how to create a label in the motion control setting function.
Refer to Section 4.3.2 or Section 5.3.2 for registration examples of labels.

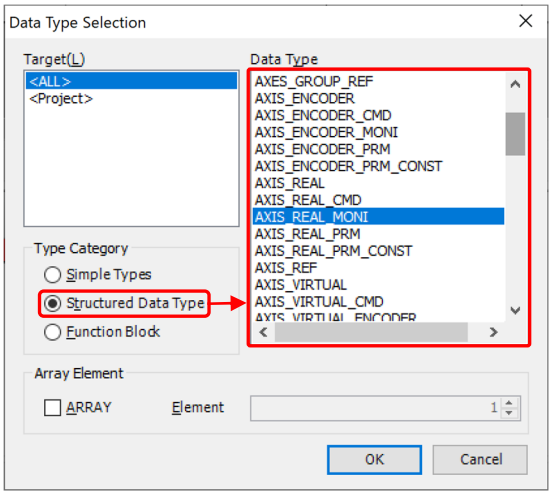
1) Double-click "Label" → "Global Label" → "Global" in the navigation window of the motion control setting function to open the "Global [Global Label Setting]" window.

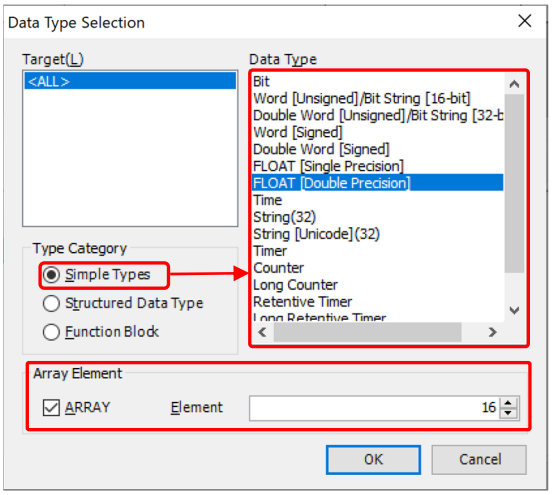


2) Set a global label.
(a) Enter the label name. (Setting example: G_bSRVONCMD)
(b) Select the data type. (Click the [...] button to open the "Data Type Selection" window.)
Select "Simple Types" in the type category, and select the data type.



Set the structured data type variables and the array variables in the same manner as Simple Types, and click the [OK] button.





[Registration example of global labels]

Global labels for the programs used in Chapter 5 are shown below. (Refer to Chapter 5 for details.)

Global [Global Label Setting]

	Label Name	Data Type	Class	Initial	Constant	Comment	Remark	Public Label	Motion Control Attribute
1	G_bSRVONCMD	Bit	VAR_GLOBAL			サーボオンオフ / Servo ON/OFF		Enable	WRITE (=> Motion)
2	G_bHoming1CMD	Bit	VAR_GLOBAL			原点復帰指令 Axis0001 / Homing command Axis0001		Enable	WRITE (=> Motion)
3	G_bHoming2CMD	Bit	VAR_GLOBAL			原点復帰指令 Axis0002 / Homing command Axis0002		Enable	WRITE (=> Motion)
4	G_bHoming3CMD	Bit	VAR_GLOBAL			原点復帰指令 VirtualAxis0001 / Homing command VirtualAxis001		Enable	WRITE (=> Motion)
5	G_bPosCMD	Bit	VAR_GLOBAL			単軸位置決め始動 / Single axis positioning start		Enable	WRITE (=> Motion)
6	G_bContPosCMD	Bit	VAR_GLOBAL			単軸連続位置決め始動 / Single axis continuous positioning start		Enable	WRITE (=> Motion)
7	G_bInterpolationCMD	Bit	VAR_GLOBAL			2軸直線補間制御始動 / 2-axis linear interpolation control start		Enable	WRITE (=> Motion)
8	G_bSyncCMD	Bit	VAR_GLOBAL			同期制御始動 / Synchronous control start		Enable	WRITE (=> Motion)
9	G_bResetCMD	Bit	VAR_GLOBAL			エラーリセット / Error reset		Enable	WRITE (=> Motion)
10	G_bMotionResetCMD	Bit	VAR_GLOBAL			システムエラーリセット / System error reset		Enable	WRITE (=> Motion)
11	G_bJogF1CMD	Bit	VAR_GLOBAL			JOG正転指令 Axis0001 / JOG Positive rotation command Axis0001		Enable	WRITE (=> Motion)
12	G_bJogR1CMD	Bit	VAR_GLOBAL			JOG逆転指令 Axis0001 / JOG Reverse rotation command Axis0001		Enable	WRITE (=> Motion)
13	G_bJogF2CMD	Bit	VAR_GLOBAL			JOG正転指令 Axis0002 / JOG Positive rotation command Axis0002		Enable	WRITE (=> Motion)
14	G_bJogR2CMD	Bit	VAR_GLOBAL			JOG逆転指令 Axis0002 / JOG Reverse rotation command Axis0002		Enable	WRITE (=> Motion)
15	G_leJogVelocity	FLOAT [Double Precision]	VAR_GLOBAL			JOG速度 / JOG Velocity		Enable	WRITE (=> Motion)
16	G_bHoming1Done	Bit	VAR_GLOBAL			原点復帰完了 Axis0001 / Homing complete Axis0001		Enable	READ (Motion =>)
17	G_bHoming2Done	Bit	VAR_GLOBAL			原点復帰完了 Axis0002 / Homing complete Axis0002		Enable	READ (Motion =>)
18	G_bHoming3Done	Bit	VAR_GLOBAL			原点復帰完了 VirtualAxis0001 / Homing complete VirtualAxis0001		Enable	READ (Motion =>)
19	G_bPosDone	Bit	VAR_GLOBAL			単軸位置決め完了 / Single axis positioning complete		Enable	READ (Motion =>)
20	G_bContPosDone	Bit	VAR_GLOBAL			単軸連続位置決め完了 / Single axis continuous positioning complete		Enable	READ (Motion =>)
21	G_bInterpolationDone	Bit	VAR_GLOBAL			2軸直線補間制御完了 / 2-axis linear interpolation control complete		Enable	READ (Motion =>)
22	G_bSyncDone	Bit	VAR_GLOBAL			同期制御完了/ Synchronous control complete		Enable	READ (Motion =>)
23	G_bJog1Busy	Bit	VAR_GLOBAL			JOG運転中 Axis0001 / JOG operation in progress Axis0001		Enable	READ (Motion =>)
24	G_bJog2Busy	Bit	VAR_GLOBAL			JOG運転中 Axis0002 / JOG operation in progress Axis0002		Enable	READ (Motion =>)
25	G_bStopSignalX	Bit	VAR_GLOBAL			停止指令 Axis0001 / Stop command Axis0001		Enable	READ (Motion =>)
26	G_bStopSignalY	Bit	VAR_GLOBAL			停止指令 Axis0002 / Stop command Axis0002		Enable	READ (Motion =>)
27									

(Note): Refer to Section 3.14 for details of public labels.

3.14 Public Labels

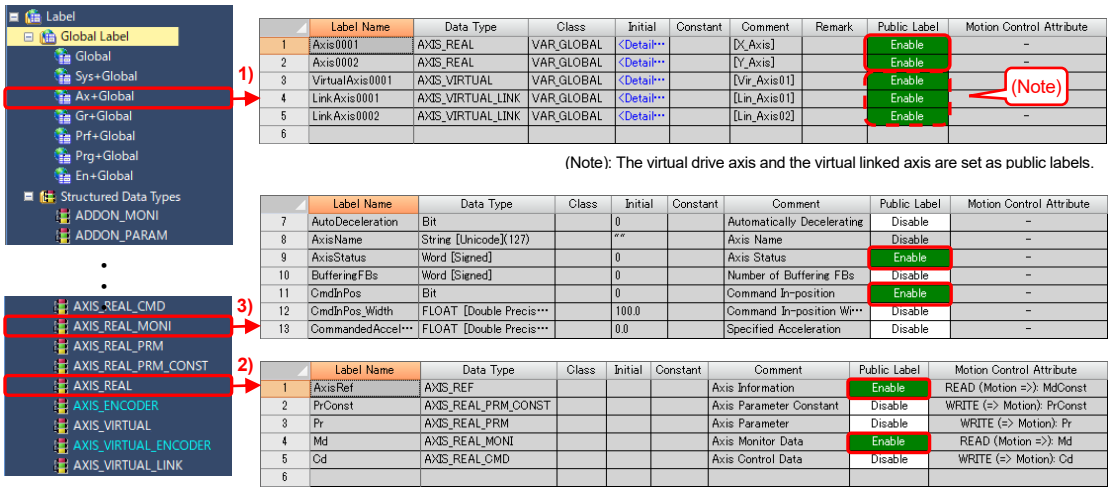
A PLC CPU can read and write global labels and structure members in the Motion modules by setting them as public labels.
Data such as monitor data of position and speed is defined as a variable with a specified name (label) in the Motion module. The data can be used from the PLC CPU by setting the labels as public labels.
This section describes how to register a public label. Refer to Section 4.3.2 or Section 5.3.2 for registration examples of public labels.

3.14.1 Public label registration

The following shows how to register the axis information (AxisRef), the axis status (AxisStatus), and the command in-position (CmdInPos) of the real drive axis as a public label.

[Registering Motion module structures as a public label]

1) Double-click "Label" → "Global Label" → "Ax+Global" in the navigation window of the motion control setting function, and set the "Public Label" column of the target labels to "Enable".
2) Double-click "Structured Data Types" → "AXIS_REAL", and set the "Public Label" column of the AxisRef (axis information) and the Md (axis monitor data) to "Enable".
3) Double-click "Structured Data Types" → "AXIS_REAL_MONI", and set the "Public Label" column of the AxisStatus (axis status) and the CmdInPos (command in-position) to "Enable".



	Label Name	Data Type	Class	Initial	Constant	Comment	Remark	Public Label	Motion Control Attribute
1	Axis0001	AXIS_REAL	VAR_GLOBAL	<Detail…		[X_Axis]		Enable	-
2	Axis0002	AXIS_REAL	VAR_GLOBAL	<Detail…		[Y_Axis]		Enable	-
3	VirtualAxis0001	AXIS_VIRTUAL	VAR_GLOBAL	<Detail…		[Vir_Axis01]		Enable	
4	LinkAxis0001	AXIS_VIRTUAL_LINK	VAR_GLOBAL	<Detail…		[Lin_Axis01]		Enable	
5	LinkAxis0002	AXIS_VIRTUAL_LINK	VAR_GLOBAL	<Detail…		[Lin_Axis02]		Enable	-
6									

	Label Name	Data Type	Class	Initial	Constant	Comment	Public Label	Motion Control Attribute
7	AutoDeceleration	Bit		0		Automatically Decelerating	Disable	-
8	AxisName	String [Unicode](127)		""		Axis Name	Disable	-
9	AxisStatus	Word [Signed]		0		Axis Status	Enable	-
10	BufferingFBs	Word [Signed]		0		Number of Buffering FBs	Disable	-
11	CmdInPos	Bit		0		Command In-position	Enable	-
12	CmdInPos_Width	FLOAT [Double Precis…		100.0		Command In-position Wi…	Disable	-
13	CommandedAccel…	FLOAT [Double Precis…		0.0		Specified Acceleration	Disable	-

	Label Name	Data Type	Class	Initial	Constant	Comment	Public Label	Motion Control Attribute
1	AxisRef	AXIS_REF				Axis Information	Enable	READ (Motion =>): MdConst
2	PrConst	AXIS_REAL_PRM_CONST				Axis Parameter Constant	Disable	WRITE (=> Motion): PrConst
3	Pr	AXIS_REAL_PRM				Axis Parameter	Disable	WRITE (=> Motion): Pr
4	Md	AXIS_REAL_MONI				Axis Monitor Data	Enable	READ (Motion =>): Md
5	Cd	AXIS_REAL_CMD				Axis Control Data	Disable	WRITE (=> Motion): Cd
6								

[Setting user-created global labels as public labels]

A PLC CPU can read and write user-created global labels in the Motion module by setting them as public labels.

The motion control attribute is required to be set for the user-created global labels.

	Label Name	Data Type	Class	Initial	Constant	Comment	Remark	Public Label	Motion Control Attribute
8	G_bSyncCMD	Bit	VAR_GLOBAL			同期制御始動 / Synchronous control start		Enable	WRITE (=> Motion)
9	G_bResetCMD	Bit	VAR_GLOBAL			エラーリセット / Error reset		Enable	WRITE (=> Motion)
10	G_bMotionResetCMD	Bit	VAR_GLOBAL			システムエラーリセット / System error reset		Enable	WRITE (=> Motion)
11	G_bJogF1CMD	Bit	VAR_GLOBAL			JOG正転指令 Axis0001 / JOG Positive rotation command Axis0001		Enable	WRITE (=> Motion)
12	G_bJogR1CMD	Bit	VAR_GLOBAL			JOG逆転指令 Axis0001 / JOG Reverse rotation command Axis0001		Enable	WRITE (=> Motion)
13	G_bJogF2CMD	Bit	VAR_GLOBAL			JOG正転指令 Axis0002 / JOG Positive rotation command Axis0002		Enable	WRITE (=> Motion)
14	G_bJogR2CMD	Bit	VAR_GLOBAL			JOG逆転指令 Axis0002 / JOG Reverse rotation command Axis0002		Enable	WRITE (=> Motion)
15	G_eJogVelocity	FLOAT [Double Precision]	VAR_GLOBAL			JOG速度 / JOG Velocity		Enable	WRITE (=> Motion)
16	G_bHoming1Done	Bit	VAR_GLOBAL			原点復帰完了 Axis0001 / Homing complete Axis0001		Enable	READ (Motion =>)
17	G_bHoming2Done	Bit	VAR_GLOBAL			原点復帰完了 Axis0002 / Homing complete Axis0002		Enable	READ (Motion =>)
18	G_bHoming3Done	Bit	VAR_GLOBAL			原点復帰完了 VirtualAxis0001 / Homing complete VirtualAxis0001		Enable	READ (Motion =>)
19	G_bPosDone	Bit	VAR_GLOBAL			単軸位置決め完了 / Single axis positioning complete		Enable	READ (Motion =>)
20	G_bContPosDone	Bit	VAR_GLOBAL			単軸連続位置決め完了 / Single axis continuous positioning complete		Enable	READ (Motion =>)
21	G_bInterpolationDone	Bit	VAR_GLOBAL			2軸直線補間制御完了 / 2-axis linear interpolation control complete		Enable	READ (Motion =>)
22	G_bSyncDone	Bit	VAR_GLOBAL			同期制御完了 / Synchronous control complete		Enable	READ (Motion =>)
23	G_bJog1Busy	Bit	VAR_GLOBAL			JOG運転中 Axis0001 / JOG operation in progress Axis0001		Enable	READ (Motion =>)

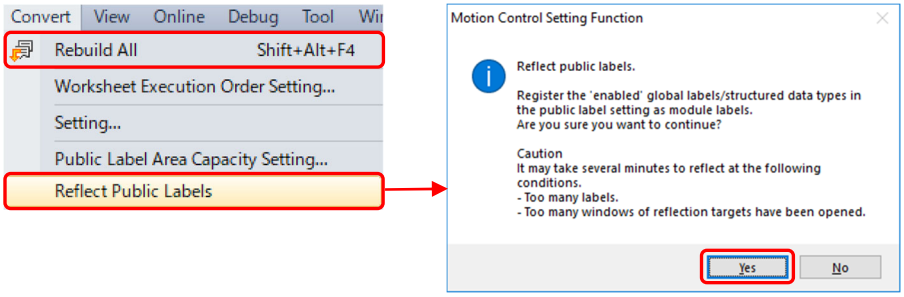
Reading label data in the Motion module: READ (Motion →)
Writing data to labels in the Motion module: WRITE (→ Motion)

3.14.2 Public label reflection

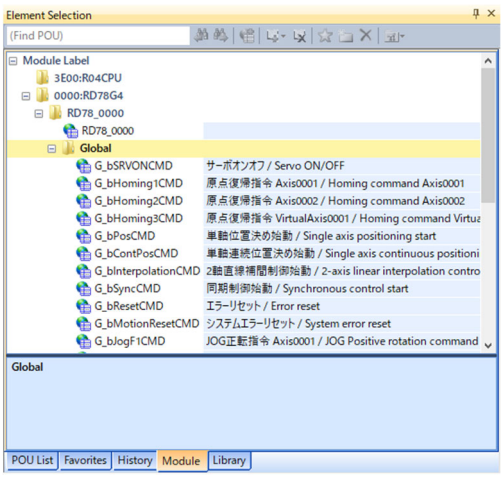
Reflect the registered public labels.

After the labels are reflected, the PLC CPU can use the members registered as public labels.

1) In the motion control setting function, select [Convert] → [Rebuild All] to convert all public labels.
2) Select [Convert] → [Reflect Public Labels], and click the [Yes] button on the confirmation screen.



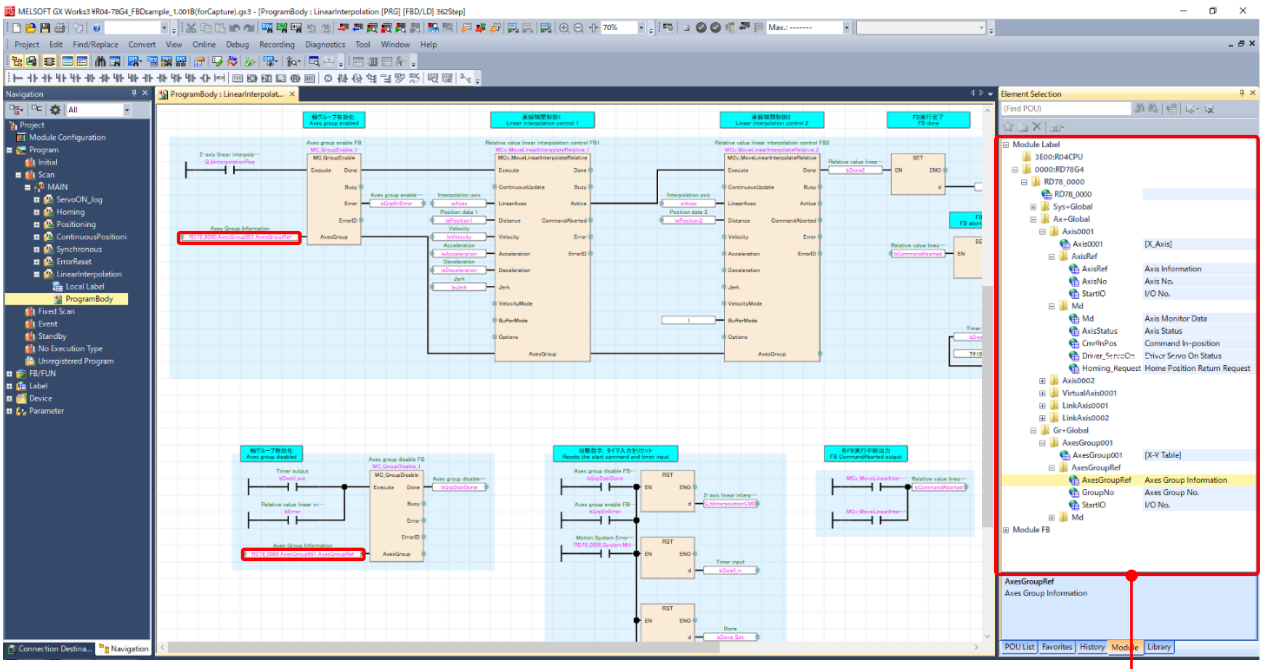
3) Convert all the program in MELSOFT GX Works3 for the PLC CPU to use the public labels.
The reflected public labels are registered in the element selection window of MELSOFT GX Works3



3.14.3 A program example using public labels

A program example using public labels is shown below.

1) Double-click "Module Label" on the [Module] tab in the element selection window to open the public labels. Drag and drop the public labels.



[POINTS]

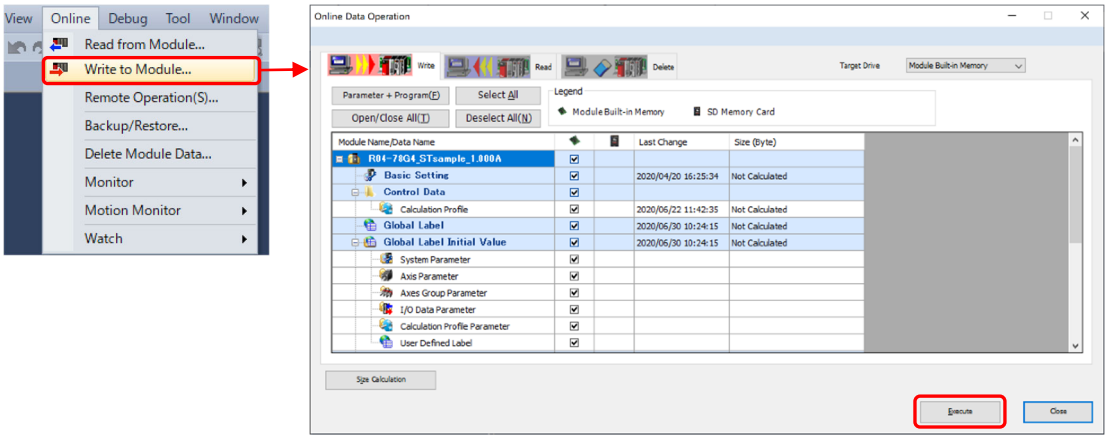
When a public label of a Motion module is used from a PLC CPU, "Module + Module No." is added at the head of the name.

RD78_0000.Axis0001.AxisRef
- Member
- Axis label name
- Module + Module No.

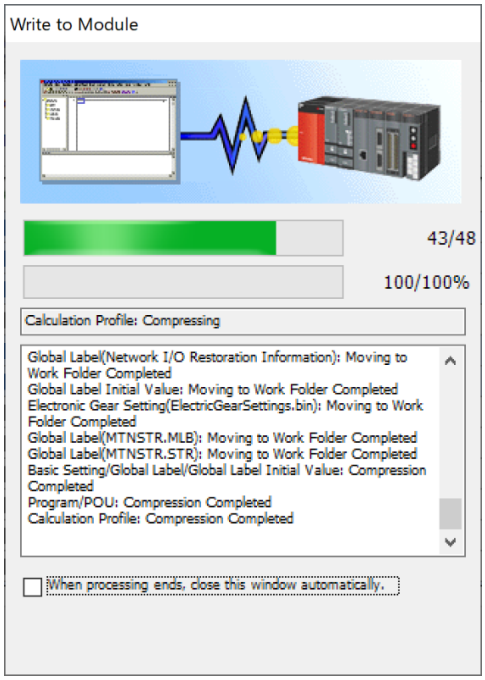
3.15 Writing to a Motion Module

After writing the module parameters to the PLC CPU, reset the PLC CPU, and turn the PLC CPU power ON again to communicate with the Motion module.
Then, write the items set in the motion control setting to the Motion module.

1) Select [Online] → [Write to Module] to open the "Online Data Operation" screen.
Check the items to be written on the screen, and click the [Execute] button.



2) Write the parameters to the Motion module.

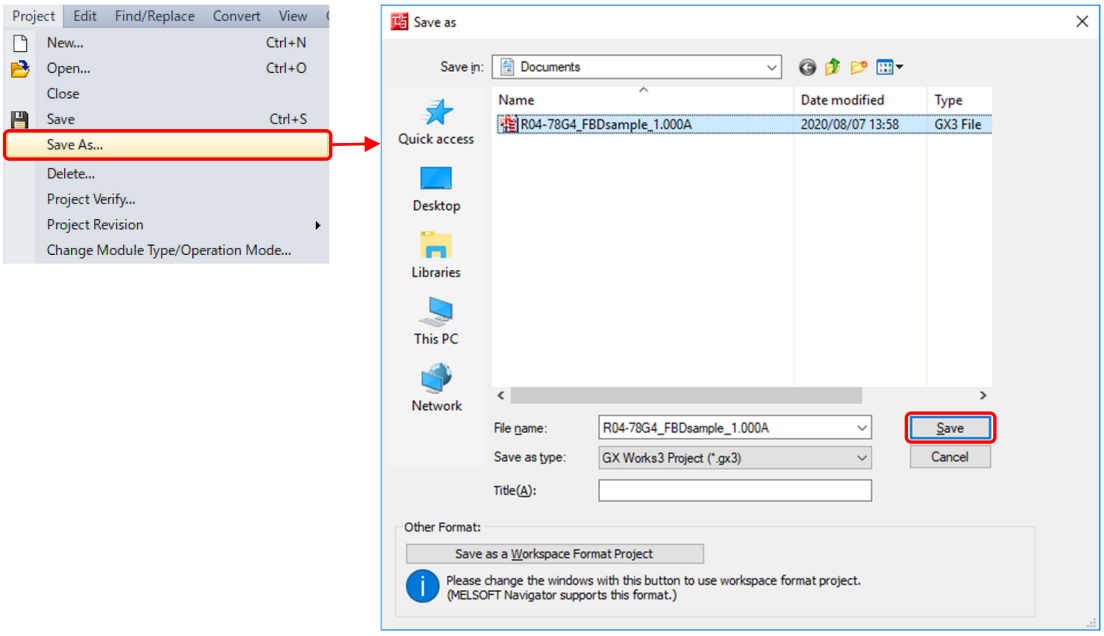


3) Turn the PLC CPU power OFF to ON again, and check that no error occurs.
The error status can be checked with the LED display of the PLC CPU and the Motion module, or the system monitor of MELSOFT GX Works3.

3.16 Saving a Project

Save the created project.

1) Select [Project] → [Save As] on MELSOFT GX Works3 to open the "Save As" screen. Enter a file name, and click the [Save] button.



3.17 Parameter List

The following shows a list of the parameters which are set in this chapter.
Refer to Chapter 4 and Chapter 5 for the public label setting.

[Axis parameter setting]
1) Real Drive Axis

Item	Axis0001	Axis0002
Axis Information		
Axis No.	1	2
Axis Parameter Constant	**Expands setting values at axis variable initialization. Re-i**	
Station Address Setting	192.168.3.1	192.168.3.2
Axis Type Setting	0:Real Drive Axis	0:Real Drive Axis
Upper Limit Signal		
Signal		
Target		
Signal Detection Meth	0:Detection at TRUE	0:Detection at TRUE
Compensation Time	0.0 s	0.0 s
Filter Time	0.0 s	0.0 s
Lower Limit Signal		
Signal		
Target		
Signal Detection Meth	0:Detection at TRUE	0:Detection at TRUE
Compensation Time	0.0 s	0.0 s
Filter Time	0.0 s	0.0 s
Control Cycle Setting	0:Operate in the First Operati	0:Operate in the First Operati
Absolute Position Refere	3:Feed Machine Position	3:Feed Machine Position
Absolute Position Manag	-1:Automatic Setting (Acquire	-1:Automatic Setting (Acquire
Ring Counter Enabled S	0:Disabled	0:Disabled
Ring Counter Lower Lim	-10000000000.0	-10000000000.0
Ring Counter Upper Lim	10000000000.0	10000000000.0
Slave Emulation Enablec	0:Disabled	0:Disabled
Torque Limit Maximum \	1000.0 %	1000.0 %
Negative Direction Torq	300.0 %	300.0 %
Positive Direction Torqu	300.0 %	300.0 %
High-speed Mode Settin	0000	0000
Axis Parameter	**Expands initial values at axis variable**	
Acceleration Limit Value	2147483647.0 um/s^2	21474836
Operation Selection at S	-1:Error (Not Started)	-1:Error (N
Command In-position Wi	100.0 um	100.0 um
Deceleration Limit Value	2147483647.0 um/s^2	21474836
Driver Unit Conversion I	67108864 pulse	67108864
Driver Unit Conversion I	5000 um	5000 um
Forced Stop Signal		
Signal		
Target		
Signal Detection Meth	0:Detection at TRUE	0:Detection at TRUE
Compensation Time	0.0 s	0.0 s
Filter Time	0.0 s	0.0 s
Home Position Return R	1:Home Position Return Requ	1:Home Position Return Requ
Jerk Limit Value	2147483647.0 um/s^3	2147483647.0 um/s^3
Operation Setting at Ove	1:Immediate Stop	1:Immediate Stop
Start Permission at Hom	0:Disabled	0:Disabled
Deceleration at Stop	0.0 um/s^2	0.0 um/s^2
Stop Selection at Decele	1:Recreate Deceleration Curv	1:Recreate Deceleration Curv
Stop Selection at Stop by	3:Alternative Acceleration/De	3:Alternative Acceleration/De
Stop Selection at H/W St	1:Immediate Stop	1:Immediate Stop
Process Selection at Sel	0:Ignore	0:Ignore
Stop Selection at S/W St	1:Immediate Stop	1:Immediate Stop
Driver Command Discar	1:Detection Enabled	1:Detection Enabled
Stop Signal		
Signal		
Target	[VAR]G_bStopSignalX	[VAR]G_bStopSignalY
Signal Detection Meth	0:Detection at TRUE	0:Detection at TRUE
Compensation Time	0.0 s	0.0 s
Filter Time	0.0 s	0.0 s
Software Stroke Limit Lo	-10000000000.0 um	-10000000000.0 um
Software Stroke Limit Ta	-1:Invalid	-1:Invalid
Software Stroke Limit Up	10000000000.0 um	10000000000.0 um
Position Command Unit	um	um
Position Command Unit		
Speed Command Unit	U/s	U/s
Negative Direction Spee	2500000000.0 um/s	2500000000.0 um/s
Operation Setting at Spe	0:Ignore	0:Ignore
Positive Direction Speec	2500000000.0 um/s	2500000000.0 um/s

2) Virtual Drive Axis

Stop Signal	
Signal	
Target	
Signal Detection Method	0:Detection at TRUE
Compensation Time	0.0 s
Filter Time	0.0 s
Software Stroke Limit Lower Value	-10000000000.0 um
Software Stroke Limit Target	-1:Invalid
Software Stroke Limit Upper Value	10000000000.0 um
Position Command Unit	um
Position Command Unit String	
Speed Command Unit	U/s
Negative Direction Speed Limit Value	2500000000.0
Operation Setting at Speed Limit Valu	0:Ignore
Positive Direction Speed Limit Value	2500000000.0

Item	VirtualAxis0001
Axis Information	
Axis No.	301
Axis Parameter Constant	**Expands setting values at axis var**
Axis Type Setting	3:Virtual Drive Axis
Upper Limit Signal	
Signal	
Target	
Signal Detection Method	0:Detection at TRUE
Compensation Time	0.0 s
Filter Time	0.0 s
Lower Limit Signal	
Signal	
Target	
Signal Detection Method	0:Detection at TRUE
Compensation Time	0.0 s
Filter Time	0.0 s
Control Cycle Setting	0:Operate in the First Operation Cyc
Absolute Position Management Setting	0:Disable Absolute Position System
Ring Counter Enabled Selection	0:Disabled
Ring Counter Lower Limit Value	-10000000000.0
Ring Counter Upper Limit Value	10000000000.0
High-speed Mode Setting	0000
Axis Parameter	**Expands initial values at axis varia**
Acceleration Limit Value	2147483647.0 um/s^2
Operation Selection at Start Accelerat	-1:Error (Not Started)
Command In-position Width	100.0 um
Deceleration Limit Value	2147483647.0 um/s^2
Forced Stop Signal	
Signal	
Target	
Signal Detection Method	0:Detection at TRUE
Compensation Time	0.0 s
Filter Time	0.0 s
Home Position Return Required or No	1:Home Position Return Required
Jerk Limit Value	2147483647.0 um/s^3
Operation Setting at Overrun	1:Immediate Stop
Start Permission at Home Position Re	0:Disabled
Deceleration at Stop	0.0 um/s^2
Stop Selection at Deceleration to Stop	1:Recreate Deceleration Curve
Stop Selection at Stop by Factors	3:Alternative Acceleration/Decelerati
Stop Selection at H/W Stroke Limit Err	1:Immediate Stop
Process Selection at Servo OFF Com	0:Ignore
Stop Selection at S/W Stroke Limit Err	1:Immediate Stop

3) Virtual Linked Axis

Item	LinkAxis0001	LinkAxis0002
Axis Information		
Axis No.	401	402
Axis Parameter Constant	**Expands setting values at axis variable initialization.Re-i**	
Axis Type Setting	5:Virtual Link Axis	5:Virtual Link Axis
Control Cycle Setting	0:Operate in the First Operati	0:Operate in the First Operati
Absolute Position Manag	0:Disable Absolute Position S	0:Disable Absolute Position S
Ring Counter Enabled S	0:Disabled	0:Disabled
Ring Counter Lower Lim	-10000000000.0	-10000000000.0
Ring Counter Upper Lim	10000000000.0	10000000000.0
High-speed Mode Settin	0000	0000
Axis Parameter	**Expands initial values at axis variable initialization.Re-in**	
Home Position Return R	0:Home Position Return Not F	0:Home Position Return Not F
Start Permission at Hom	0:Disabled	0:Disabled
Position Command Unit	um	um
Position Command Unit		
Speed Command Unit	U/s	U/s

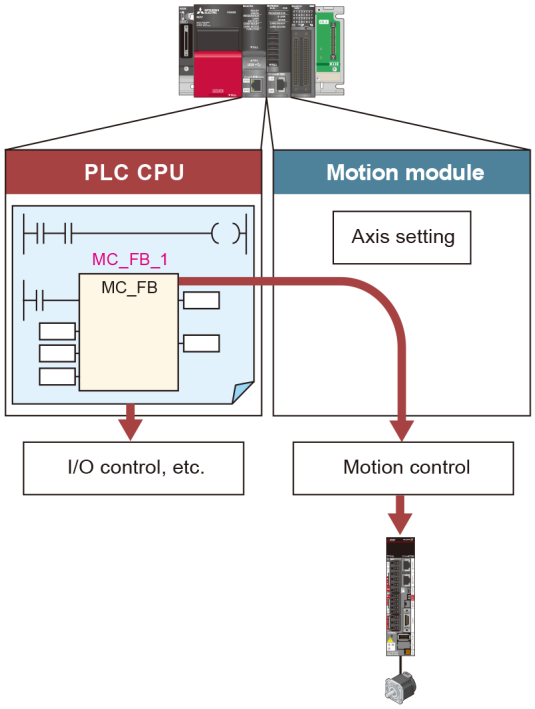
[Axes group setting]

Item	AxesGroup001
Axes Group Information	
Axes Group No.	1
Axes Group Parameter	**Expands initial values at a**
Acceleration Limit Value	2147483647.0 um/s^2
Operation Selection at S	-1:Error (Not Started)
Structuring Axis[1]	Axis0001
Structuring Axis[2]	Axis0002
Structuring Axis[3]	
Structuring Axis[4]	
Structuring Axis[5]	
Structuring Axis[6]	
Structuring Axis[7]	
Structuring Axis[8]	
Structuring Axis[9]	
Structuring Axis[10]	
Structuring Axis[11]	
Structuring Axis[12]	
Structuring Axis[13]	
Structuring Axis[14]	
Structuring Axis[15]	
Structuring Axis[16]	
Command In-position Wi	100.0 um
Deceleration Limit Value	2147483647.0 um/s^2
Jerk Limit Value	2147483647.0 um/s^3
Operation Setting at Ove	1:Immediate Stop
Deceleration at Stop	0.0 um/s^2
Stop Selection at Decele	1:Recreate Deceleration Curv
Structuring Axis Operati	1:Immediate Stop
Stop Selection at Stop by	3:Alternative Acceleration/De
Position Command Unit	um
Position Command Unit	
Speed Command Unit	U/s
Speed Limit Value	2500000000.0 um/s

4. PROGRAMMING BY A PLC CPU ONLY

A PLC CPU is programmed by any of the following languages: FBD/LD, ladder, or ST language. This chapter explains programming with FBD/LD that enables users to easily understand the relationship between FBs.
To obtain the sample program used in this chapter, contact your local sales office.

■ File name: **R04-78G4_FBDsample_*****.gx3** (***** indicates the version)



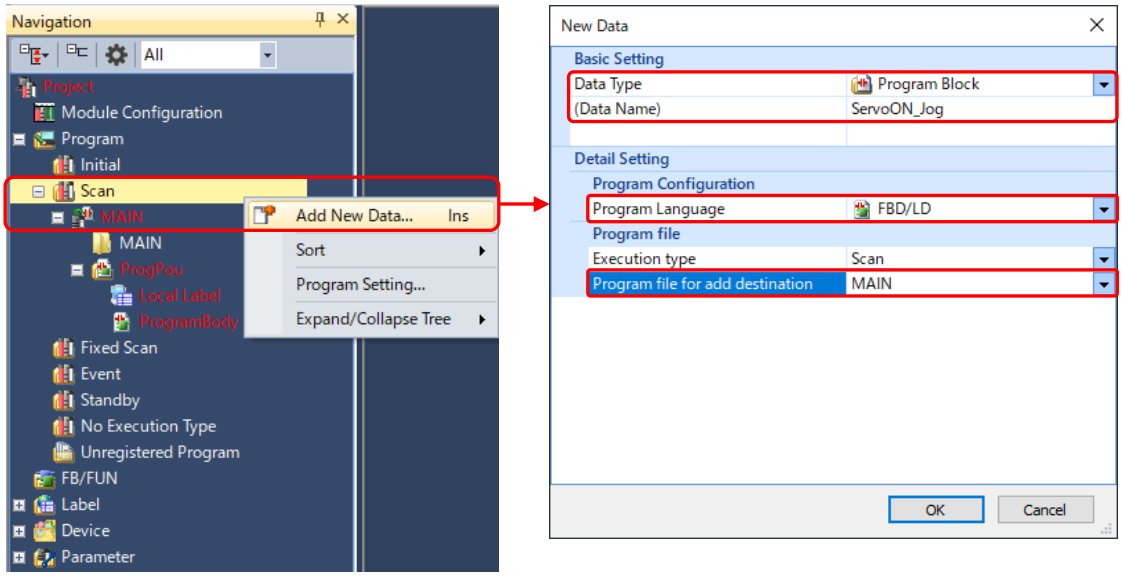
When applying the sample program to an actual system, ensure the applicability and confirm that it will not cause system control problems.
Consider and add interlock conditions according to the user's system.

4.1 Programming Procedure for PLC CPU

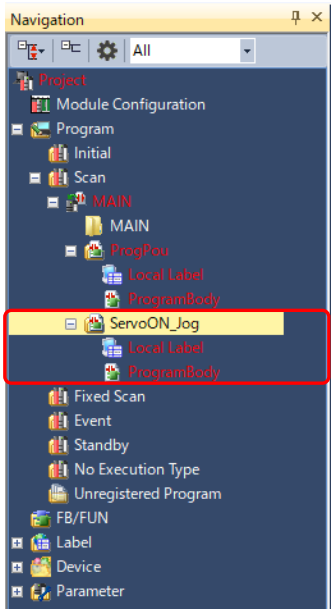
MELSOFT GX Works3 is used to create a program for a PLC CPU.

4.1.1 Creating a program block

1) In the navigation window of MELSOFT GX Works3, right-click an execution type ("Scan" in this example) under "Program", and click "Add New Data".
2) On the "New Data" screen, set the data name, the program language, and the program file for add destination. Only alphanumeric characters can be used to for the data name.



3) A program block is added to the navigation window.



4.1.2 Program execution type

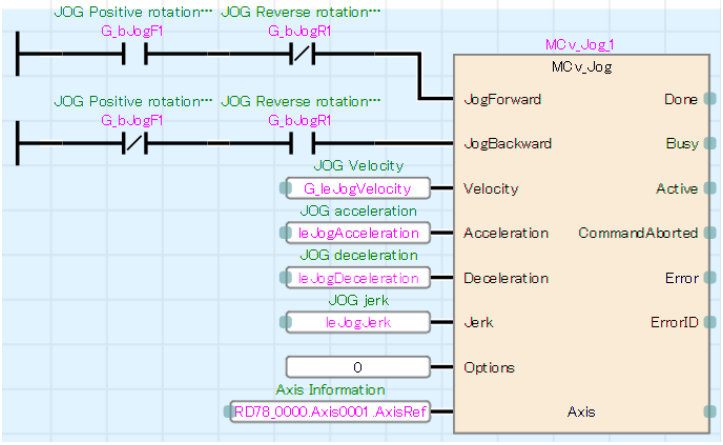
There are following five execution types.

Execution type	Description
Initial	This type of program is executed only once when the CPU module has been powered OFF and ON, or switched from the STOP state to the RUN state.
Scan	This type of program is executed only once per scan. The execution starts from the next scan right after the initial execution type program is executed.
Fixed scan	This is an interrupt program which is executed at a specified time interval. Differently from the normal interrupt program, this type of program does not require the interrupt pointer (I) and the IRET instruction and is executed for each program file unit.
Event	This type of program starts execution when triggered by a specified event.
Standby	This type of program is executed only when its execution is requested.

To set the execution type, right-click on a target program in the navigation window, and select [Register Program] from the shortcut menu, or drag and drop the program onto the target execution type.
The set execution type will be applied to "Program Setting" of "CPU Parameter".

4.1.3 Inputting a FB

Register the Motion module FBs on MELSOFT GX Works3. (Refer to Section 2.4.4.)
This section explains the procedure for setting the following JOG operation FB (MCv_Jog).
The sample program adds some interlock conditions to this program.



(1) Preparing labels

Prepare labels for input/output signals of the FB. This example uses labels for the Velocity, Acceleration, Deceleration, and Jerk inputs of MCv_Jog. When registering a new label, consider whether the label is registered as a global or a local label.

In this example, the labels for JOG command and JOG velocity are registered as a global label, considering the JOG velocity is operated from an external device, such as GOT (HMI).
The labels for JOG acceleration, JOG deceleration, and JOG jerk are as local labels of ServoON_Jog (refer to Section 4.1.1) since these labels are used only within the program.
Refer to Section 4.2 for naming rules of labels.

[Global label]

Global [Global Label Setting]

<Filter> | Show Details(Y) | Display Setting | Check

	Label Name	Data Type		English(Display Target)	Access from External Device
1	G_bJogF1	Bit	...	JOG Positive rotation command Axis0001	
2	G_bJogR1	Bit	...	JOG Reverse rotation command Axis0001	
3	G_leJogVelocity	FLOAT [Double Precision]	...	JOG Velocity	
4			...		

[Local label]

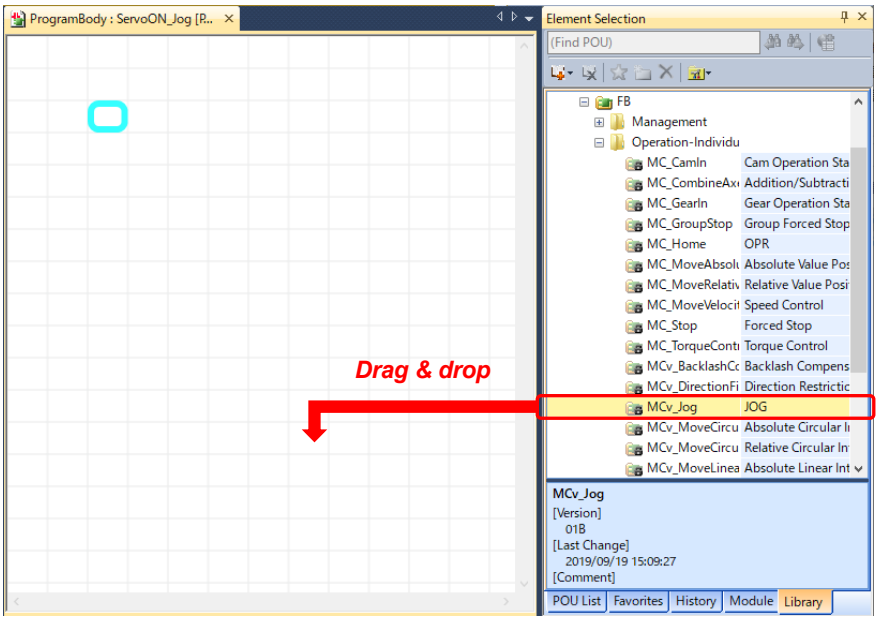
ServoON_Jog [PRG] [Local Labe...

<Filter> | Show Details(Y) | Display Setting | Check

	Label Name	Data Type		English(Display Target)
1	leJogAcceleration	FLOAT [Double Precision]	...	JOG acceleration
2	leJogDeceleration	FLOAT [Double Precision]	...	JOG deceleration
3	leJogJerk	FLOAT [Double Precision]	...	JOG jerk
4			...	

(2) Inputting a FB

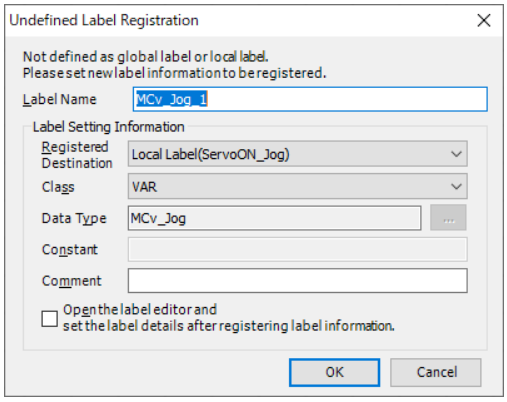
Select "library" → "MotionControl_RD78G_****" → "FB" on the [Library] tab in the element selection window, and you will find the Motion module FB library is registered.
Drag and drop a desired FB to the program editor.



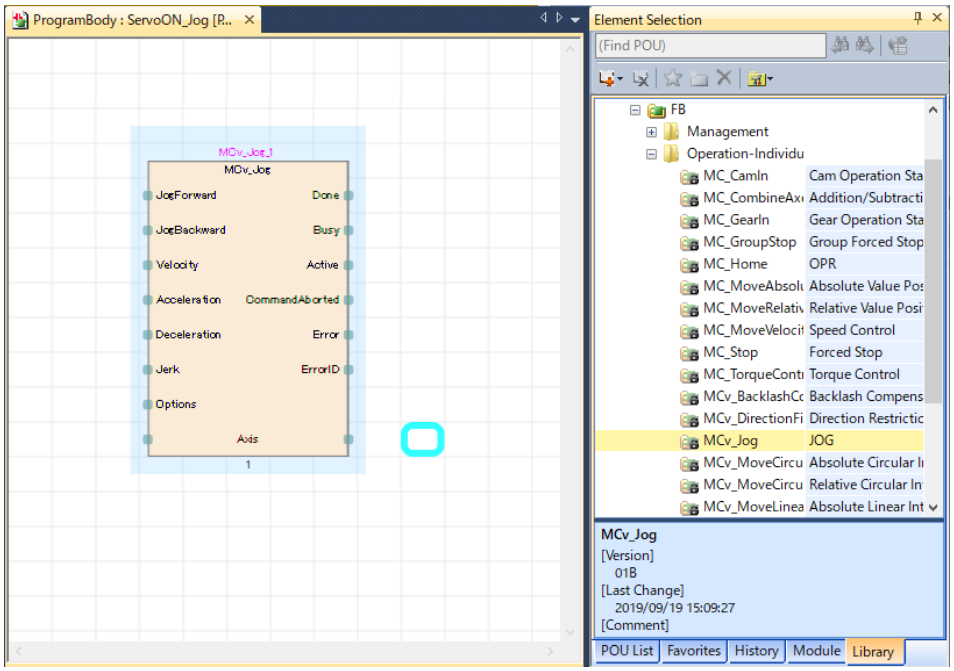
Motion module FBs are categorized in the following three groups.

Group	Description
Management (administrative FB)	A motion control FB that takes an axis or an axes group for the argument and does not change the axis status or the axes group status by execution
Operation-Individual (motion FB)	A motion control FB that takes an axis or an axes group for the argument and changes the axis status or the axes status by execution
StandardFB (general FB)	A motion control FB that does not take an axis or an axes group for the argument

The FB for JOG operation (MCv_Jog) is under the "Operation-Individual" tree.
Drag and drop MCv_Jog to the editor. When an undefined label is entered, the "Undefined Label Registration" screen appears. Enter the label name, the registered destination, and, if necessary, the comment. This example uses the initial settings.



When the setting is completed, click the [OK] button.
The selected FB (MCv_Jog) is displayed on the program editor.

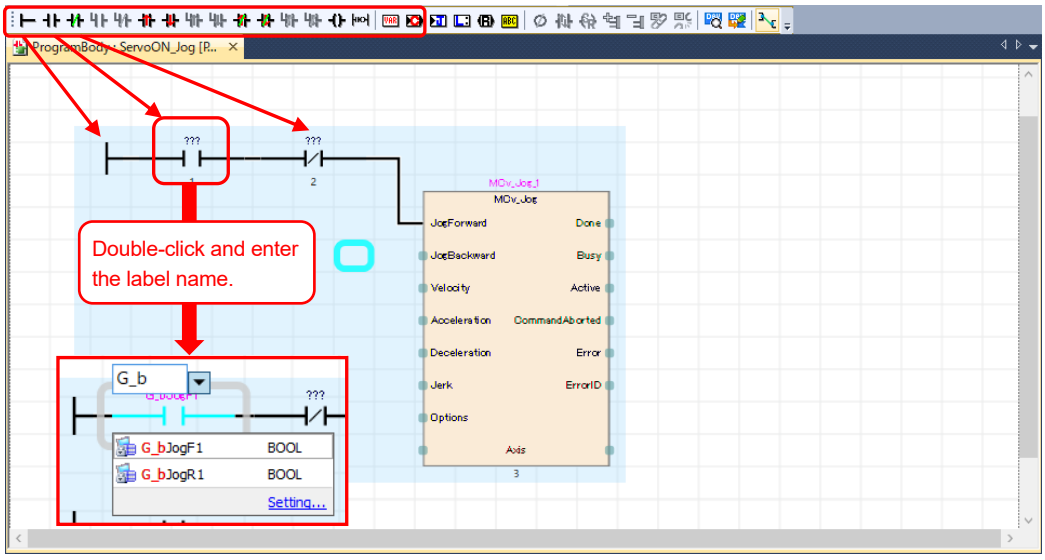


(3) Connecting I/O signals

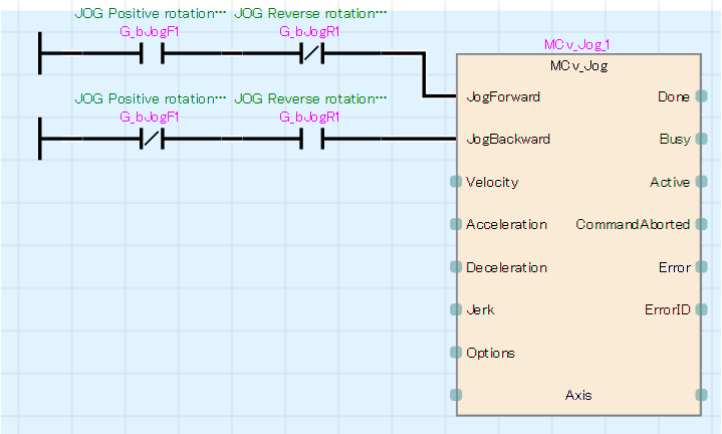
Connect FBD and LD elements to a target input connection point of the FB.
Place an element on the editor, and move it through drag and drop.

(a) LD element

Click where an element is to be placed on the program editor, and then click the element icon on the tool bar to place the element.
When a contact element is placed, it shows "???". Double-click the element, and enter a bit type label or a bit device number. For the label name, the registered global labels are displayed as a suggestion. Connect the LD elements to the input connection point of the FB.



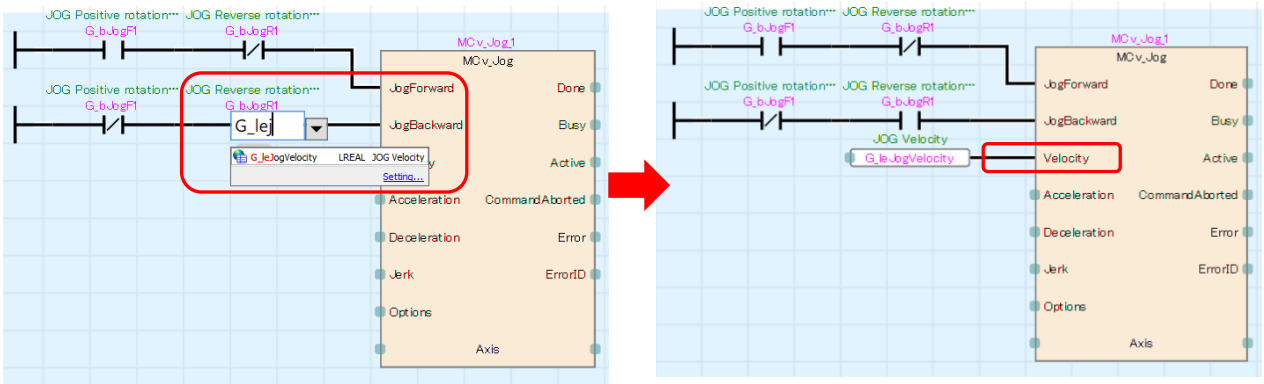
The following shows the diagram that connects the LD elements to the JogForward and JogBackward input connection points.



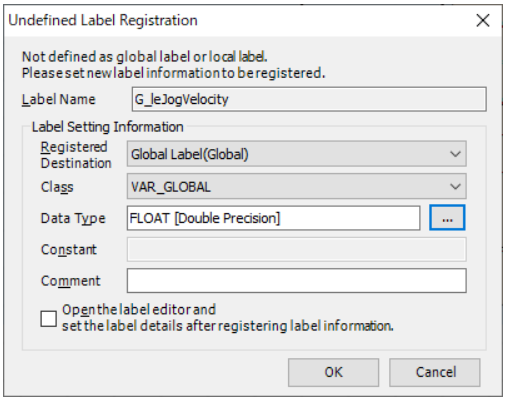
(b) FBD elements

Click where the FBD element is to be placed, and then directly enter the device No. or label name there.

When entering "G_leJogVelocity" that is registered as a global label in (1), click the left side of the Velocity input of MCv_Jog and enter "G_leJogVelocity". The registered global labels are displayed as a suggestion. Connect the FBD element to the input connection point of the FB.



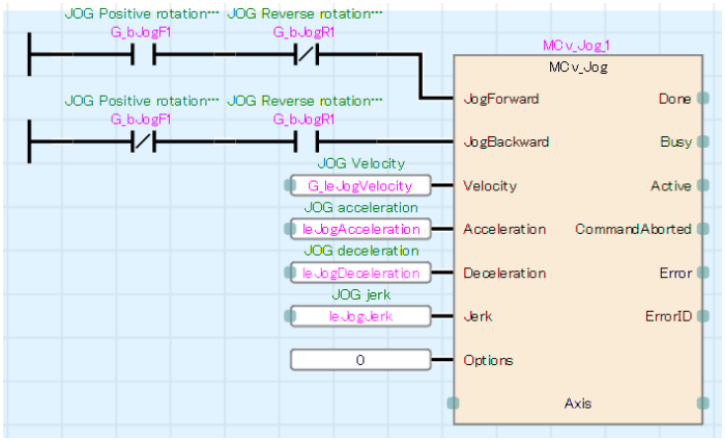
When an undefined label is entered, the "Undefined Label Registration" screen appears. Enter the data type and the registered destination.



Connect the local labels (leJogAcceleration, leJogDeceleration, and leJogJerk) registered in (1) to the corresponding input connection point (Acceleration, Deceleration, and Jerk inputs) of MCv_Jog. For the Options input, connect the constant value of "0".

[NOTES]

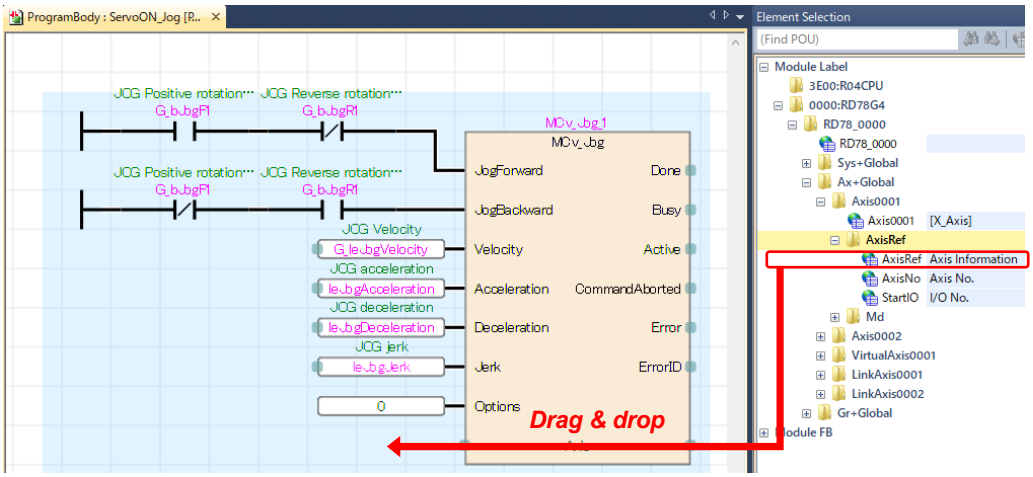
Note that values need to be stored to G_leJogVelocity, leJogAccelertion, leJogDeceleration, and leJogJerk by another program (using EDMOVP instruction, etc.)
(For a setting example, refer to Section 4.6.)



(c) Axis input (public label)

Connect the AxisRef structure registered as a public label to the Axis input connection point. (Refer to Section 3.14.1.)

Click [Module] tab in the element selection window. Select "Module label" → "0000:RD78G4" → "RD78_0000" → "Ax+Global" → "Axis0001" → "AxisRef", and then drag and drop the AxisRef (axis information) to the program editor.



When "RD78G_0000.Axis0001.AxisRef" appears on the editor, connect it to the Axis input connection point of the FB.

[POINTS]

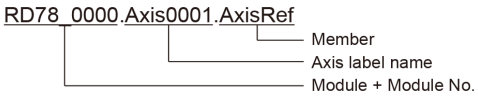
The name of public label "RD78_0000.Axis0001.AxisRef" can be directly entered on the editor.

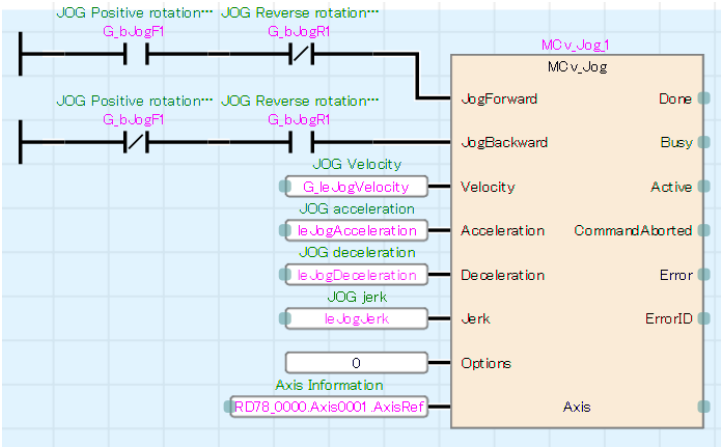
When you enter "RD78", "RD78_0000" is displayed as a suggestion.

When you select "RD78_0000" and enter ".", the axis label name, such as "Axis0001", will be displayed.

When you select "Axis0001" and enter ".", the structure member, such as "AxisRef", will be displayed.

Select "AxisRef", and the input is completed.





The program is completed.

[POINTS]

When using an initial value for FB input signals, the setting can be omitted.

The Options input in this example can be omitted since it sets the initial value of "0".

4.2 Naming Rules of Labels

This document uses the following prefix rules for labels to distinguish their data type.

Data type		Value range	Prefix	
			Local	Global
Bit	BOOL	FALSE(0), TRUE(1)	b	G_b
Word [unsigned] /bit string [16 bits]	WORD	0 to 65535	u	G_u
Double word [unsigned]/ bit string [32 bits]	DWORD	0 to 4294967295	ud	G_ud
Word [signed]	INT	-32468 to 32767	w	G_w
Double word [signed]	DINT	-2147483648 to 2147483647	d	G_d
Single-precision real number	REAL	-2^{128} to -2^{-126}, 0, 2^{-126} to 2^{128}	e	G_e
Double-precision real number	LREAL	-2^{1024} to -2^{-1022}, 0, 2^{-1022} to 2^{1024}	le	G_le
Time	TIME	T#-24d20h31m23s648ms to T#24d20h31m23s647ms	tm	G_tm
Timer	TIMER	TIMER structure S: contact C: coil N: current value	td	G_td

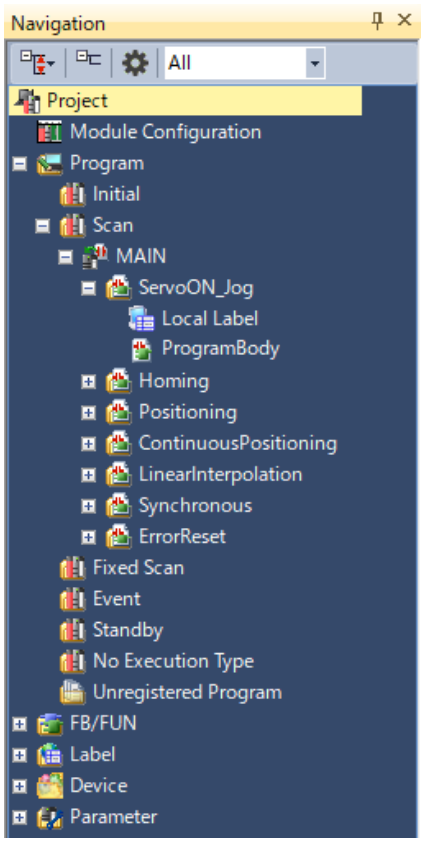
[Examples of local labels]
Bit: bMoveCMD
Double-precision real number: lePosition
Arrays: wAxes[16]
Timer: tdTimer1

[Examples of global labels]
Bit: G_bJogF1
Double-precision real number: G_leVelocity

4.3 Projects

4.3.1 Program names

The following shows the program examples explained in this chapter.



"Scan"→ under "MAIN"

Program name: ServoON_Jog
(PLC READY, all axes servo ON, JOG operation)

Program name: Homing

Program name: Positioning (single axis positioning)

Program name: ContinuousPositioning
(single axis continuous positioning)

Program name: LinearInterpolation
(two-axis linear interpolation control)

Program name: Synchronous (synchronous control)

Program name: ErrorReset (error reset)

4.3.2 Settings for global and public labels

The following shows the setting examples of the global labels in the sample program.
For the local label, refer to each program.

(1) PLC CPU

1)
2)
3)

	Label Name	Data Type	Class	Assign (Device/Label)	Initial Value	Constant	English(Display Target)
1	G_bSRVOFF	Bit	VAR_GLOBAL				Servo OFF
2	G_bJogF1	Bit	VAR_GLOBAL				JOG Positive rotation command Axis0001
3	G_bJogR1	Bit	VAR_GLOBAL				JOG Reverse rotation command Axis0001
4	G_bJogF2	Bit	VAR_GLOBAL				JOG Positive rotation command Axis0002
5	G_bJogR2	Bit	VAR_GLOBAL				JOG Reverse rotation command Axis0002
6	G_bHoming1CMD	Bit	VAR_GLOBAL				Homing command Axis0001
7	G_bHoming2CMD	Bit	VAR_GLOBAL				Homing command Axis0002
8	G_bHoming3CMD	Bit	VAR_GLOBAL				Homing command VirtualAxis0001
9	G_bPosCMD	Bit	VAR_GLOBAL				Single axis positioning start
10	G_bContPosCMD	Bit	VAR_GLOBAL				Single axis continuous positioning start
11	G_bInterpolationCMD	Bit	VAR_GLOBAL				2-axis linear interpolation control start
12	G_bSyncCMD	Bit	VAR_GLOBAL				Synchronous control start
13	G_bErrorReset	Bit	VAR_GLOBAL				Error reset
14	G_bSysErrorReset	Bit	VAR_GLOBAL				System error reset
15	G_leJogVelocity	FLOAT [Double Precision]	VAR_GLOBAL				JOG Velocity
16	G_bHoming1Req	Bit	VAR_GLOBAL				Homing start request Axis0001
17	G_bHoming2Req	Bit	VAR_GLOBAL				Homing start request Axis0002
18	G_bHoming3Req	Bit	VAR_GLOBAL				Homing start request VirtualAxis0001
19	G_bPosReq	Bit	VAR_GLOBAL				Single axis positioning start request
20	G_bContPosReq	Bit	VAR_GLOBAL				Single axis continuous positioning start request
21	G_bInterpolationReq	Bit	VAR_GLOBAL				2-axis linear interpolation control start request
22	G_bSyncReq	Bit	VAR_GLOBAL				Synchronous control start request
23	G_bJog1Busy	Bit	VAR_GLOBAL				JOG operation in progress Axis0001
24	G_bJog2Busy	Bit	VAR_GLOBAL				JOG operation in progress Axis0002
25							

1) These labels are used to execute operation of each program.
2) This double-precision real number label stores the JOG velocity.
   By registering this label as a global label, the JOG velocity can be changed from a GOT (HMI) or other programs.
3) These labels turn ON while the program is being executed.
   They are used as an interlock condition to prevent simultaneous execution of multiple programs.

(2) Motion module

The following shows the global and public label settings of the Motion module.
Set these labels on the motion control setting function of MELSOFT GX Works3. When the settings are finished, reflect the public labels. (Refer to Section 3.14.2.)

[AX+Global]

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Public Label	Motion Control
1	Axis0001	AXIS_REAL	VAR_GLOBAL	<Detailed…		[X_Axis]	Enable	–
2	Axis0002	AXIS_REAL	VAR_GLOBAL	<Detailed…		[Y_Axis]	Enable	–
3	VirtualAxis0001	AXIS_VIRTUAL	VAR_GLOBAL	<Detailed…		[Vir_Axis01]	Enable	–
4	LinkAxis0001	AXIS_VIRTUAL_LINK	VAR_GLOBAL	<Detailed…		[Lin_Axis01]	Enable	–
5	LinkAxis0002	AXIS_VIRTUAL_LINK	VAR_GLOBAL	<Detailed…		[Lin_Axis02]	Enable	–
6								

[Gr+Global]

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Remark	Public Label	Motion Control Attribute
1	AxesGroup001	AXES_GROUP	VAR_GLOBAL	<Detailed Setti…		[X-Y Table]		Enable	–
2									

[Global]

The following are the structures that store data related to the system. Set "Public label" to "Enable".

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Remark	Public Label	Motion Control Attribute
1	G_bStopSignalX	Bit	VAR_GLOBAL			停止指令 Axis0001 / Stop command Axis0001		Enable	WRITE (=> Motion)
2	G_bStopSignalY	Bit	VAR_GLOBAL			停止指令 Axis0002 / Stop command Axis0002		Enable	WRITE (=> Motion)
3									

[Sys+Global]
The following is the structure that stores data related to the system. Set "Public label" to "Enable".

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Remark	Public Label	Motion Control Attribute
1	System	SYSTEM	VAR_GLOBAL	<Detailed Setti…				Enable	-
2									

[AXIS_REAL structure]

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Public Label	Motion Control Attribute
1	AxisRef	AXIS_REF				Axis Information	Enable	READ (Motion =>): MdConst
2	PrConst	AXIS_REAL_PRM_CO…				Axis Parameter Constant	Disable	WRITE (=> Motion): PrConst
3	Pr	AXIS_REAL_PRM				Axis Parameter	Disable	WRITE (=> Motion): Pr
4	Md	AXIS_REAL_MONI				Axis Monitor Data	Enable	READ (Motion =>): Md
5	Cd	AXIS_REAL_CMD				Axis Control Data	Disable	WRITE (=> Motion): Cd
6								

[AXIS_REAL_MONI structure]

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Public Label	Motion Control
1	AccelerationLimit	FLOAT [Double Preci…		0.0		Acceleration Limit Value	Disable	-
2	AccelerationOverride	FLOAT [Double Preci…		1.0		Acceleration Override Coefficient	Disable	-
3	AccelerationZeroBehavior	Word [Signed]		-1		Operation Selection at Start Accel…	Disable	-
4	ActualPosition	FLOAT [Double Preci…		0.0		Feedback Position	Disable	-
5	ActualVelocity	FLOAT [Double Preci…		0.0		Feedback Speed	Disable	-
6	Analyzing	Bit		0		Analyzing	Disable	-
7	AutoDeceleration	Bit		0		Automatically Decelerating	Disable	-
8	AxisName	String [Unicode](127)		""		Axis Name	Disable	-
9	AxisStatus	Word [Signed]		0		Axis Status	Enable	-
10	BufferingFBs	Word [Signed]		0		Number of Buffering FBs	Disable	-
11	CmdInPos	Bit		0		Command In-position	Enable	-
12	CmdInPos_Width	FLOAT [Double Preci…		100.0		Command In-position Width	Disable	-

≈ ≈

26	Driver_Mode	Word [Signed]		0		Driver Control Mode	Disable	-
27	Driver_ReadyOn	Bit		0		Driver Ready On Status	Disable	-
28	Driver_ServoOn	Bit		0		Driver Servo On Status	Enable	-
29	Driver_State	Word [Signed]		0		Driver Status	Disable	-
30	DriverError	Bit		0		Drive Module Error Detection	Disable	-
31	DriverErrorID	Word [Unsigned]/Bit…		0		Drive Module Error Code	Disable	-
32	DriverErrorDetailID	Word [Unsigned]/Bit…		0		Drive Module Error Detail Code	Disable	-
33	Error	Bit		0		Axis Error Detection	Disable	-
34	ErrorID	Word [Unsigned]/Bit…		0		Axis Error Code	Disable	-
35	FeedMachinePosition	FLOAT [Double Preci…		0.0		Feed Machine Position	Disable	-
36	FollowupDisable	Bit		0		Follow-up Disabled	Disable	-
37	ForcedStop_Released	Bit		0		Forced Stop Cancelling	Disable	-
38	ForcedStop_Signal	SIGNAL_SELECT				Forced Stop Signal	Disable	-
39	Homing_Complete	Bit		0		Home Position Return Completed	Disable	-
40	Homing_Request	Bit		1		Home Position Return Request	Enable	-
41	Homing_Required	Bit		0		Home Position Return Required or…	Disable	-

≈ ≈

[AXES_GROUP structure]

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Public Label	Motion Control Attribute
1	AxesGroupRef	AXES_GROUP_REF				Axes Group Information	Enable	READ (Motion =>): MdConst
2	Pr	AXES_GROUP_PRM				Axes Group Parameter	Disable	WRITE (=> Motion): Pr
3	Md	AXES_GROUP_MONI				Axes Group Monitor Data	Enable	READ (Motion =>): Md
4	Cd	AXES_GROUP_CMD				Axes Group Control Data	Disable	WRITE (=> Motion): Cd
5								

[AXES_GROUP_MONI structure]

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Public Label	Motion Control
1	AccelerationLimit	FLOAT [Double Precision]		0.0		Acceleration Limit Value	Disable	–
2	AccelerationOverride	FLOAT [Double Precision]		1.0		Acceleration Override Coefficient	Disable	–
3	AccelerationZeroBehavior	Word [Signed]		–1		Operation Selection at Start Accel…	Disable	–
4	ActualVelocity	FLOAT [Double Precision]		0.0		Feedback Speed	Disable	–
5	Analyzing	Bit		0		Analyzing	Disable	–
6	AutoDeceleration	Bit		0		Automatically Decelerating	Disable	–
7	Axis	AXIS_REF(1..16)				Structuring Axis	Disable	–
8	BufferingFBs	Word [Signed]		0		Number of Buffering FBs	Disable	–
9	CmdInPos	Bit		0		Command In-position	Enable	–
10	CmdInPos_Width	FLOAT [Double Precision]		100.0		Command In-position Width	Disable	–
11	CommandedAcceleration	FLOAT [Double Precision]		0.0		Specified Acceleration	Disable	–
12	CommandedDeceleration	FLOAT [Double Precision]		0.0		Specified Deceleration	Disable	–
13	CommandedJerk	FLOAT [Double Precision]		0.0		Specified Jerk	Disable	–
14	CommandedVelocity	FLOAT [Double Precision]		0.0		Specified Speed	Disable	–
15	DecelerationLimit	FLOAT [Double Precision]		0.0		Deceleration Limit Value	Disable	–
16	Error	Bit		0		Axes Group Error Detection	Disable	–
17	ErrorID	Word [Unsigned]/Bit Stri…		0		Axes Group Error Code	Disable	–
18	GroupName	String [Unicode](127)		""		Axes Group Name	Disable	–
19	GroupStatus	Word [Signed]		0		Axes Group Status	Enable	–
20	InterpolationAxes	Double Word [Unsigned]…		0		Interpolation Axis	Disable	–
21	InVelocity	Bit		0		Target Speed Reached	Disable	–

≈ ≈

[AXIS_VIRTUAL structure]

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Public Label	Motion Control Attribute
1	AxisRef	AXIS_REF				Axis Information	Enable	READ (Motion =>): MdConst
2	PrConst	AXIS_REAL_PRM_CO…				Axis Parameter Constant	Disable	WRITE (=> Motion): PrConst
3	Pr	AXIS_REAL_PRM				Axis Parameter	Disable	WRITE (=> Motion): Pr
4	Md	AXIS_REAL_MONI				Axis Monitor Data	Enable	READ (Motion =>): Md
5	Cd	AXIS_REAL_CMD				Axis Control Data	Disable	WRITE (=> Motion): Cd
6								

[AXIS_VIRTUAL_MONI structure]

	Label Name	Data Type	Class	Initial	Constant	Comment	Public Label	Motion Control
1	AccelerationLimit	FLOAT [Double Precision]		0.0		Acceleration Limit Value	Disable	–
2	AccelerationOverride	FLOAT [Double Precision]		1.0		Acceleration Override Coefficient	Disable	–
3	AccelerationZeroBehavior	Word [Signed]		–1		Operation Selection at Start Accel…	Disable	–
4	Analyzing	Bit		0		Analyzing	Disable	–
5	AutoDeceleration	Bit		0		Automatically Decelerating	Disable	–
6	AxisName	String [Unicode](127)		""		Axis Name	Disable	–
7	AxisStatus	Word [Signed]		0		Axis Status	Enable	–
8	BufferingFBs	Word [Signed]		0		Number of Buffering FBs	Disable	–
9	CmdInPos	Bit		0		Command In-position	Enable	–
10	CmdInPos_Width	FLOAT [Double Precision]		100.0		Command In-position Width	Disable	–
11	CommandedAcceleration	FLOAT [Double Precision]		0.0		Specified Acceleration	Disable	–

≈ ≈

20	ForcedStop_Released	Bit		0		Forced Stop Cancelling	Disable	–
21	ForcedStop_Signal	SIGNAL_SELECT				Forced Stop Signal	Disable	–
22	Homing_Complete	Bit		0		Home Position Return Completed	Disable	–
23	Homing_Request	Bit		1		Home Position Return Request	Enable	–
24	Homing_Required	Bit		0		Home Position Return Required or…	Disable	–

≈ ≈

[AXIS_VIRTUA_LINK structure]

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Public Label	Motion Control Attribute
1	AxisRef	AXIS_REF				Axis Information	Enable	READ (Motion =>): Md…
2	PrConst	AXIS_VIRTUAL_LINK_PRM_CONST				Axis Parameter Constant	Disable	WRITE (=> Motion): Pr…
3	Pr	AXIS_VIRTUAL_LINK_PRM				Axis Parameter	Disable	WRITE (=> Motion): Pr
4	Md	AXIS_VIRTUAL_LINK_MONI				Axis Monitor Data	Enable	READ (Motion =>): Md
5	Cd	AXIS_VIRTUAL_LINK_CMD				Axis Control Data	Disable	WRITE (=> Motion): Cd
6								

[SYS_MONI structure]

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Public Label	Motion Control
1	Addon_AbsSystem	ADDON_MONI				Add-on AbsSystem Monitor	Disable	–
2	Addon_Axis	ADDON_MONI				Add-on Axis Monitor	Disable	–
3	Addon_ExternalSignal	ADDON_MONI				Add-on ExternalSignal Monitor	Disable	–
4	Addon_FileTransfer	ADDON_MONI				Add-on FileTransfer Monitor	Disable	–

≈ ≈

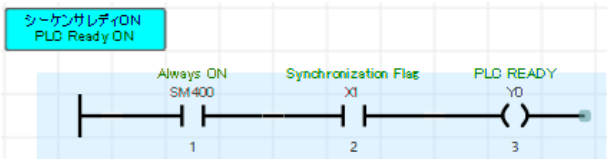
23	BuffermemoryRefreshCycle	CYCLE_MONI				Buffer Memory Refresh Cycle Moni…	Disable	–
24	Environment_UserRootPath	String(127)		''		User Root Path	Disable	–
25	Error	Bit		0		Motion System Error Detection	Enable	–
26	ErrorHistory	Word [Unsigned]…		0		Error History Information	Disable	–
27	ErrorHistory_Latest	Word [Signed]		0		Latest Error History Data No.	Disable	–

4.4 PLC READY (Program Name: ServoON_Jog)

The Motion module uses 32 points each for inputs and outputs to send/receive data to/from a PLC CPU.
Regardless of the programming language, PLC READY [Y0] needs to be turned ON by a PLC CPU to start operation of the Motion module.

(1) Program example

PLC READY flag [Y0] is turned ON when the PLC CPU is powered ON and the synchronization flag [X1] is turned ON.



4.5 Servo ON (Program Name: ServoON_Jog)

This program turns ON the real drive axes connected to the servo system.
The following two FBs are used for executing servo ON.

(1) FBs

Type	Command	Description
MCFB (administrative)	MC_Power	Switches a specified axis to the operation possible status.
	MCv_AllPower	Switches all axes to the operation possible status.

(2) Local labels

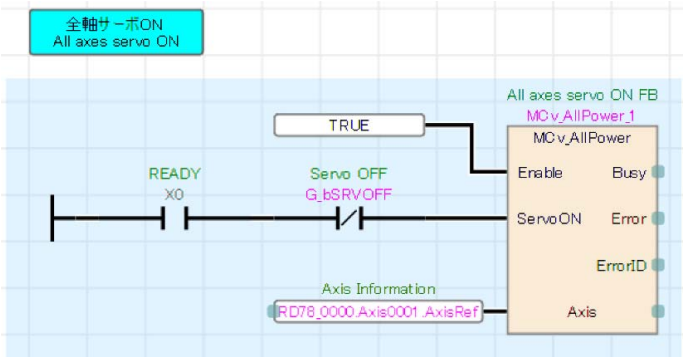
	Label Name	Data Type		English(Display Target)
1	MCv_AllPower_1	MCv_AllPower	...	All axes servo ON FB
2	MCv_Jog_1	MCv_Jog	...	JOG operation FB Axis0001
3	MCv_Jog_2	MCv_Jog	...	JOG operation FB Axis0002
4	leJogAcceleration	FLOAT [Double Precision]	...	JOG acceleration
5	leJogDeceleration	FLOAT [Double Precision]	...	JOG deceleration
6	leJogJerk	FLOAT [Double Precision]	...	JOG jerk
7			...	

The labels for JOG operation

Drag and drop MCv_AllPower to the program editor. The FB (MCv_AllPower_1) is added as a local label. (Refer to Section 4.1.3.)

(3) Program example

When PLC READY [Y0] is turned ON, Ready [X0] is turned ON.
Turning ON of X0 is used as the condition for executing all axes servo ON.
When executing servo OFF, turn ON G_bSRVOFF.
For MCv_AllPower, connect AxisRef type structure to the Axis input connection point.



[POINTS]

The I/O No. of the Motion module must be specified in the AxisRef type structure.
When a user specifies the I/O No. in the PLC CPU program, store the numerical value that deletes the lowest digit of the Motion module I/O No. to the AxisRef.StartIO.
For example, when the I/O No. is "0010", store "1" for AxisRef.StartIO.

4.6 JOG Operation (Program Name: ServoON_Jog)

The servo system outputs commands to a target axis and operates the axis in a specified direction while the JOG positive/reverse rotation commands are ON.

(1) FB

Type	Command	Description
MCFB (motion)	MCv_Jog	Executes a JOG operation according to the target velocity.

(2) Acceleration/deceleration processing function

The acceleration/deceleration function is used to adjust the acceleration/deceleration curve of the motion control according to the user's machine.

(a) Overview

Select the acceleration/deceleration methods from the following.

Method	Description
Acceleration/deceleration specification method (Initial)	Accelerates/decelerates the axis by the acceleration, deceleration, and jerk values specified in the FB.
Acceleration/deceleration time-fixed method	Accelerates/decelerates the axis based on the time specified in the FB regardless of speed.

(b) Setting method

The acceleration/deceleration method can be specified by the Options input of motionFBs, such as MCv_Jog.

Bit	Description
0 to 2	Acceleration/deceleration method setting 0: mcAccDec (acceleration/deceleration specification method) 1: mcFixedTime (acceleration/deceleration time-fixed method)
3 to 15	The function varies depending on the FB.
16 to 31	

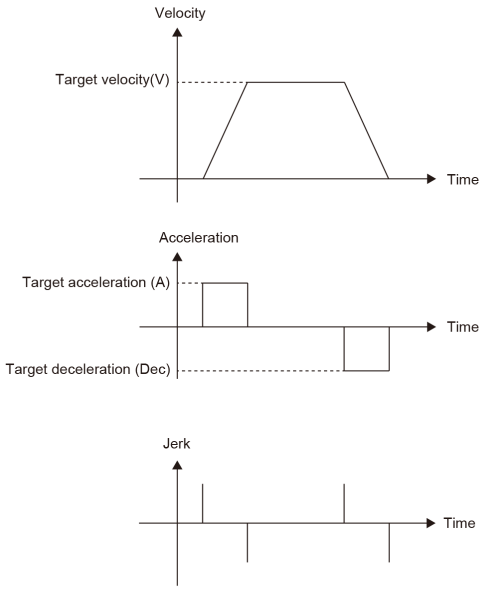
(c) Acceleration/deceleration specification method

Select "0: mcAccDec" in the Options input (using bit 0 to 2 for acceleration/deceleration method setting) of the FB, and set acceleration, deceleration, and jerk.

1) Trapezoidal acceleration/deceleration

When "0.0" is set for the Jerk input of the FB, the operation is referred to as the trapezoidal acceleration/deceleration.

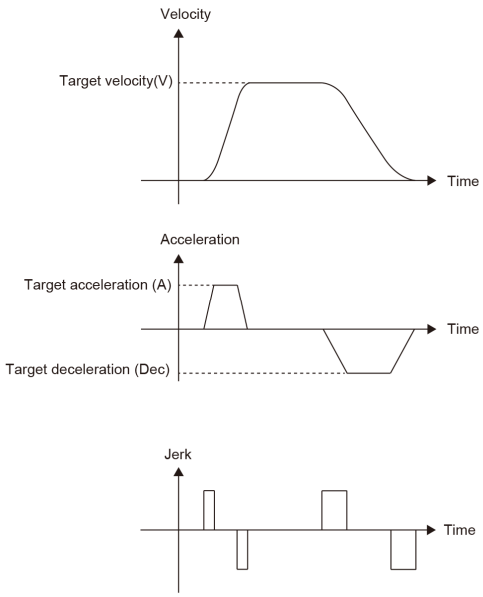
The velocity creates a trapezoidal shape.



2) Jerk acceleration/deceleration

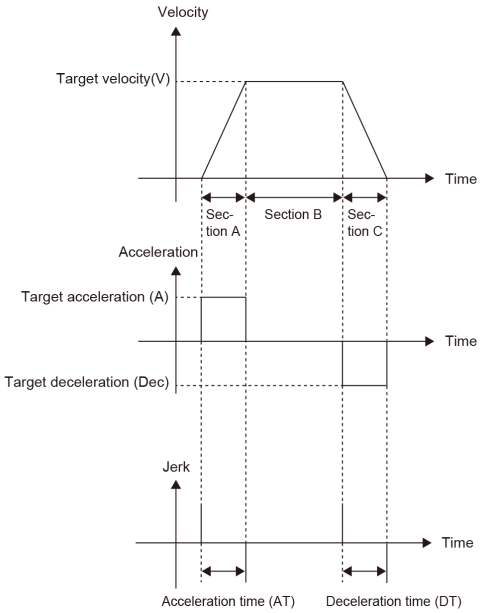
When values other than "0.0" is set for the Jerk input of the FB, the operation is referred to as the jerk acceleration/deceleration.

The velocity creates an S-curve shape.



(d) Acceleration/deceleration time-fixed method

Select "1: mcFixedTime " in the Options input (using bit 0 to 2 for acceleration/deceleration method setting) of the FB, and set the acceleration time in the Acceleration input of the FB. The Deceleration and Jerk inputs are not used.



(e) FB input variables

The following table lists the input variables explained in (b) to (d).

Input variable	Name	Details
Velocity	Target velocity	Specify the target velocity.
Acceleration	Acceleration or acceleration/ deceleration time	Specify the acceleration or the acceleration/deceleration time. • The unit is [U/s^2] when the method is set to "0: mcAccDec". • The unit is [s] when the method is set to "1: mcFixedTime".
Deceleration	Deceleration	Specify the deceleration. • The unit is [U/s^2] when the method is set to "0: mcAccDec". • Not used when the method is set to "1: mcFixedTime"
Jerk	Jerk	Specify the jerk. • The unit is [U/s^3] when the method is set to "0: mcAccDec". • Not used when the method is set to "1: mcFixedTime".
Options	Options	Specify the acceleration/deceleration method using bit 0 to2. 0: mcAccDec Acceleration/deceleration specification method 1: mcFixedTime Acceleration/deceleration time-fixed method

[POINTS]

When the "Acceleration/deceleration specification method" (Options: 0) is selected, the acceleration/deceleration can be calculated using the acceleration/deceleration time as follows.

[Target velocity: V [U/s], acceleration time [s], deceleration time [s]],

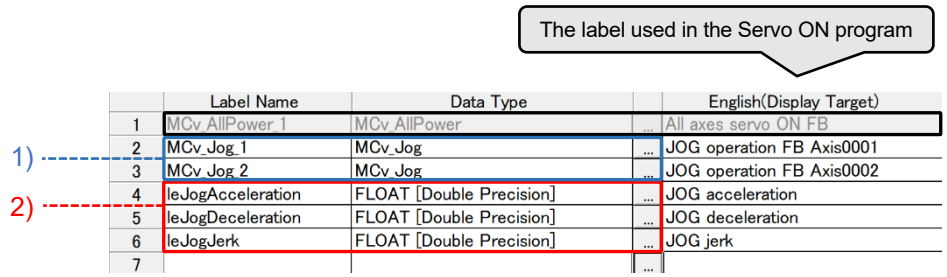
Velocity := (target velocity V);

Acceleration := (target velocity V/acceleration time);

Deceleration := (target velocity V/deceleration time);

Options := (mcAccDec);

(3) Local labels



	Label Name	Data Type		English(Display Target)
1	MCv_AllPower_1	MCv_AllPower	...	All axes servo ON FB
2	MCv_Jog_1	MCv_Jog	...	JOG operation FB Axis0001
3	MCv_Jog_2	MCv_Jog	...	JOG operation FB Axis0002
4	leJogAcceleration	FLOAT [Double Precision]	...	JOG acceleration
5	leJogDeceleration	FLOAT [Double Precision]	...	JOG deceleration
6	leJogJerk	FLOAT [Double Precision]	...	JOG jerk
7			...	

1) These labels are automatically added when the user drags and drops the FB "MCv_Jog" to the program editor.

2) These labels are registered manually.

(4) Program example

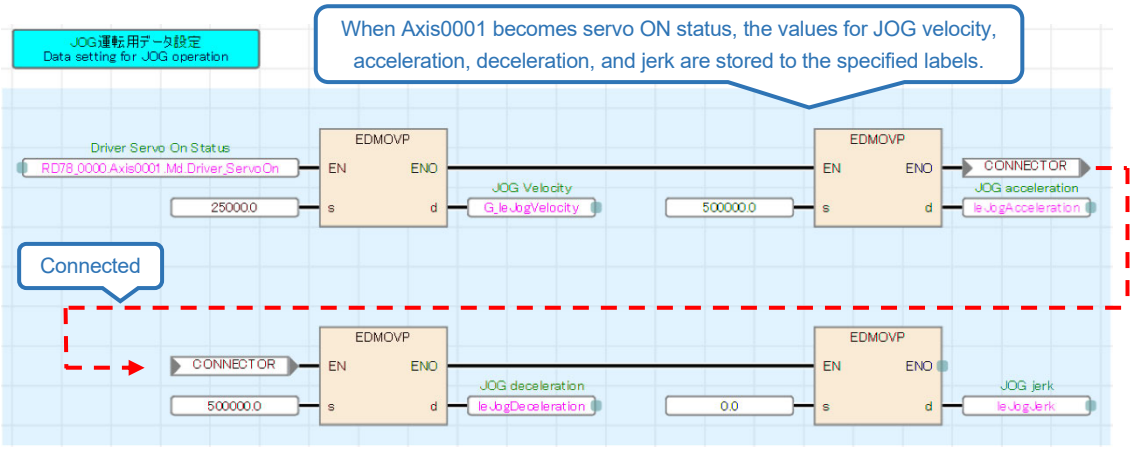
The EDMOVP instruction is used to store values to the JOG velocity, acceleration, deceleration, and jerk.

The labels for the JOG positive/reverse rotation commands are connected to the corresponding command inputs (JogForward and JogBackward inputs) of the FB (MCv_Jog).

The interlock conditions are added to prevent execution of the JOG operation while other programs are running.

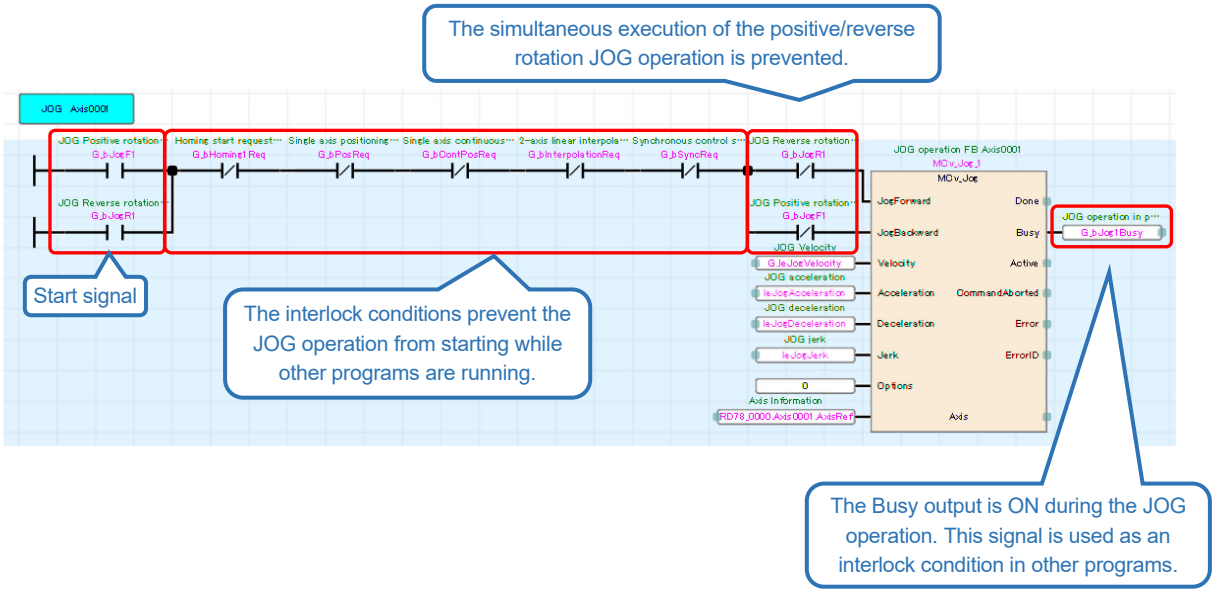
[Start signal]

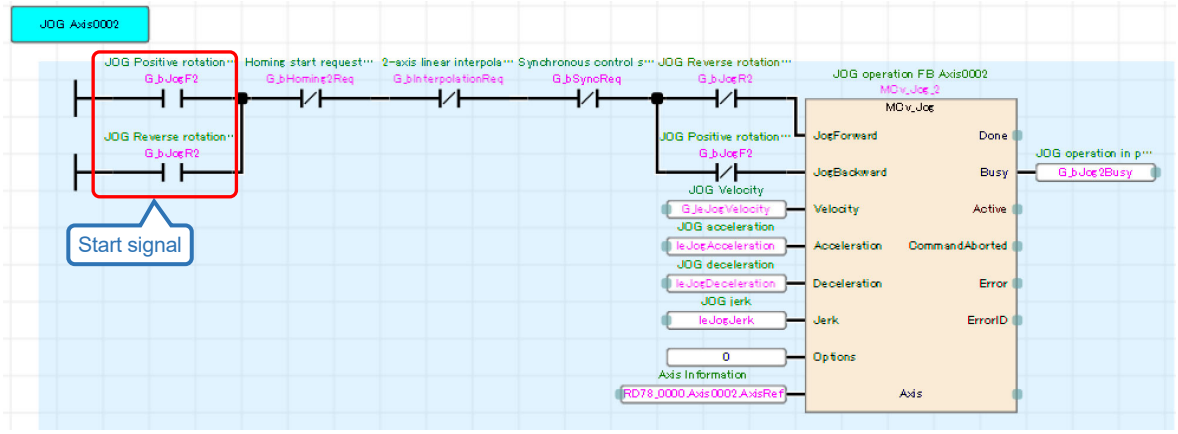
	Axis0001	Axis0002
JOG positive rotation command	G_bJogF1	G_bJogF2
JOG reverse rotation command	G_bJogR1	G_bJogR2



[POINTS]

To enter the EDMOVP instruction, directly enter "EDMOVP" on the editor, or drag and drop "EDMOVP" to the editor (select "APPLICATION INSTRUCTIONS" → "Real number instructions" on the [POU List] tab in the element selection window.)





4.7 Homing (Program Name: Homing)

The homing method is set using the parameters of the servo amplifier.
This section explains the data set method homing.
For the dog type homing, refer to Appendix.

(1) Overview

The data set type homing sets the current position as the home position.

[Time chart of data set type homing (Method37)]

(2) FB

Type	Command	Description
MCFB (motion)	MC_Home	Executes homing of the specified axis.

(3) Local labels

	Label Name	Data Type		English(Display Target)
1	MC_Home_1	MC_Home	...	Homing FB1 Axis0001
2	MC_Home_2	MC_Home	...	Homing FB2 Axis0002
3	MC_Home_3	MC_Home	...	Homing FB3 VirtualAxis0001
4	bHoming1Done	Bit	...	Homing FB1 Done output
5	bHoming1Error	Bit	...	Homing FB1 Error output
6	bHoming2Done	Bit	...	Homing FB2 Done output
7	bHoming2Error	Bit	...	Homing FB2 Error output
8	bHoming3Done	Bit	...	Homing FB3 Done output
9	bHoming3Error	Bit	...	Homing FB3 Error output
10			...	

1) These labels are automatically added when the user drags and drops the FB (MC_Home) to the program editor.

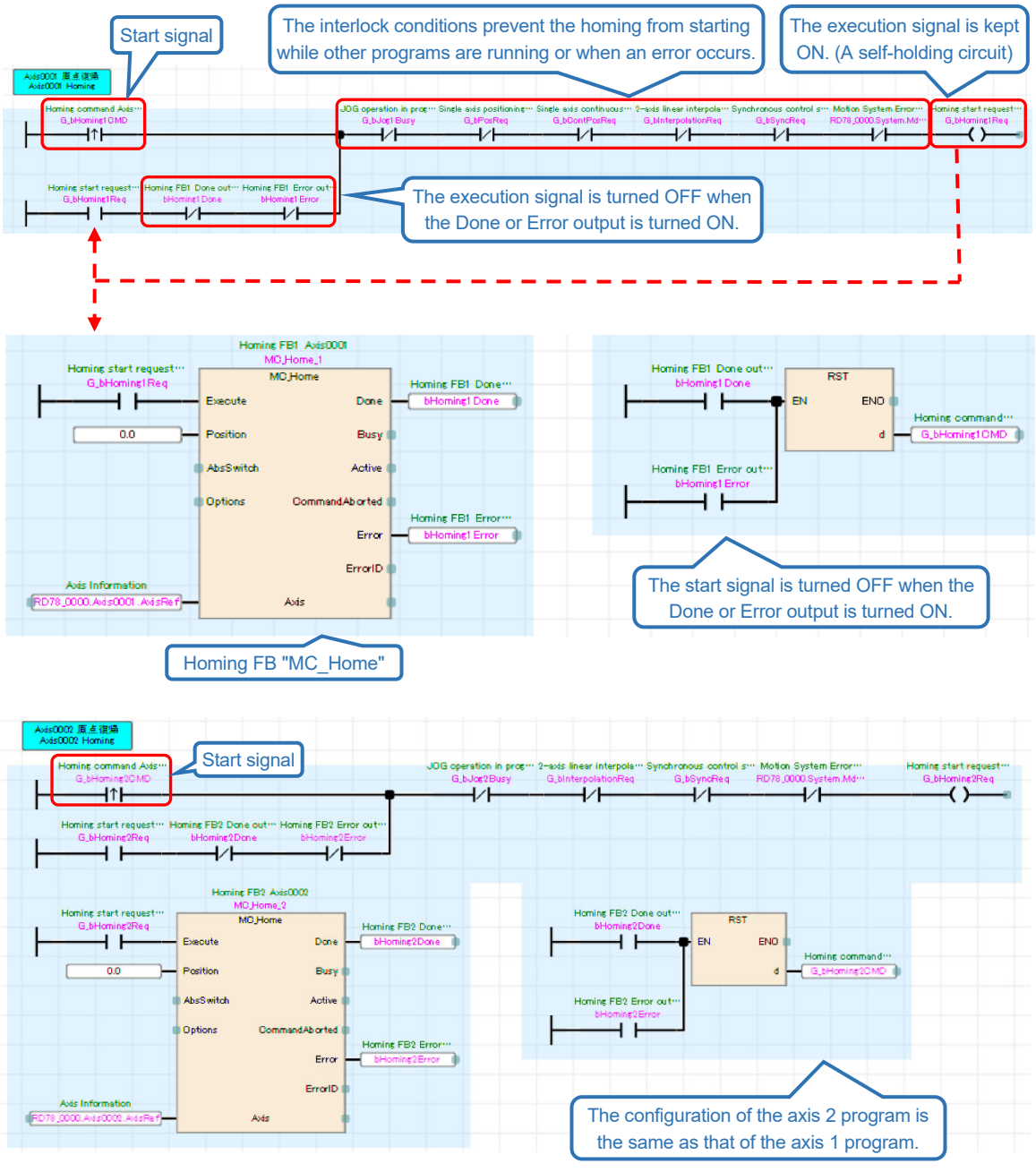
2) These labels are registered manually.

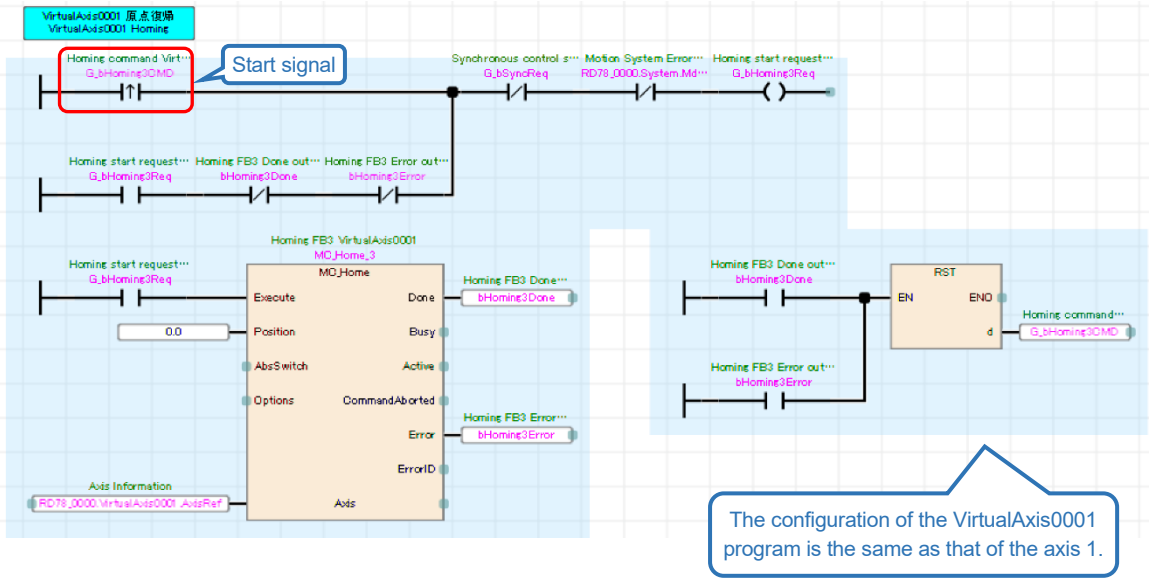
(4) Program example

This program executes homing for the axis 1 (Axis0001), the axis 2 (Axis0002), and the virtual drive axis (VirtualAxis0001). The homing request label is kept ON by a self-holding circuit and is connected to the Execute input of MC_Home. The start signal is turned OFF when the completion signal (the Done output of MC_Home) is turned ON or when an error occurs.
The interlock conditions are added to prevent execution of the homing while other programs are running or when an error occurs.

[Start signal]

	Axis0001	Axis0002	VirtualAxis0001
Homing command	G_bHoming1CMD	G_bHoming2CMD	G_bHoming3CMD




VirtualAxis0001 原点復帰
VirtualAxis0001 Homing
Homing command Virt···
G_bHoming3CMD
Start signal
Synchronous control s···
G_bSyncReq
Motion System Error···
RD78_0000.System.Md···
Homing start request···
G_bHoming3Req
Homing start request···
G_bHoming3Req
Homing FB3 Done out···
bHoming3Done
Homing FB3 Error out···
bHoming3Error
Homing FB3 VirtualAxis0001
MC_Home_3
Homing start request···
G_bHoming3Req
MC_Home
Execute
Done
Homing FB3 Done···
bHoming3Done
0.0
Position
Busy
AbsSwitch
Active
Options
CommandAborted
Homing FB3 Error···
bHoming3Error
Error
ErrorID
Axis Information
RD78_0000.VirtualAxis0001_AxisRef
Axis
Homing FB3 Done out···
bHoming3Done
RST
EN
ENO
d
Homing command···
G_bHoming3CMD
Homing FB3 Error out···
bHoming3Error
The configuration of the VirtualAxis0001 program is the same as that of the axis 1.

4.8 Single Axis Positioning Control (Program Name: Positioning)

The single axis positioning control executes positioning to a specified position by using the address information.

(1) FBs

Type	Command	Description
MCFB (motion)	MC_MoveAbsolute	Executes positioning using a specified absolute position.
	MC_MoveRelative	Executes positioning using a specified relative distance.

(2) Local labels

1) ----------
2) ----------

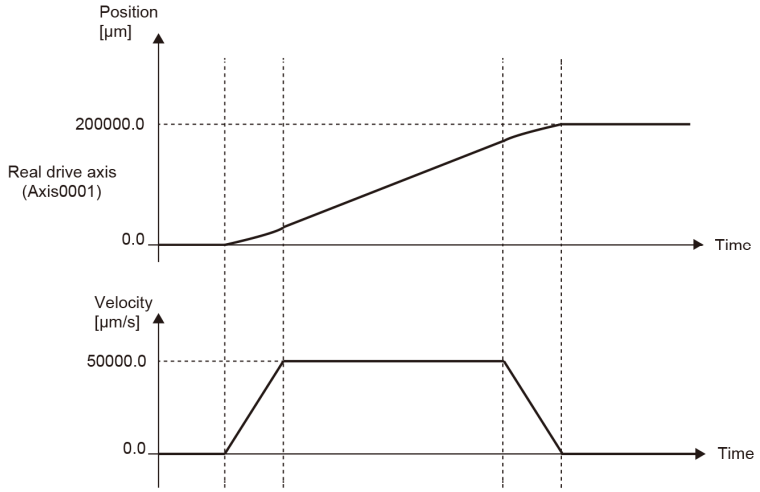
	Label Name	Data Type		English(Display Target)
1	MC_MoveRelative_1	MC_MoveRelative	...	Relative value positioning FB
2	leDistance	FLOAT [Double Precision]	...	Distance
3	leVelocity	FLOAT [Double Precision]	...	Velocity
4	leAcceleration	FLOAT [Double Precision]	...	Acceleration
5	leDeceleration	FLOAT [Double Precision]	...	Deceleration
6	leJerk	FLOAT [Double Precision]	...	Jerk
7	bDone	Bit	...	Relative value positioning FB Done output
8	bError	Bit	...	Relative value positioning FB Error output
9	bValueSet	Bit	...	Variable set complete
10	bCommandAborted	Bit	...	Relative value positioning FB CommandAborted output
11			...	

1) This label is automatically added when the user drags and drops the FB (MC_MoveRelative) to the program editor.

2) These labels are registered manually.

(3) Program example

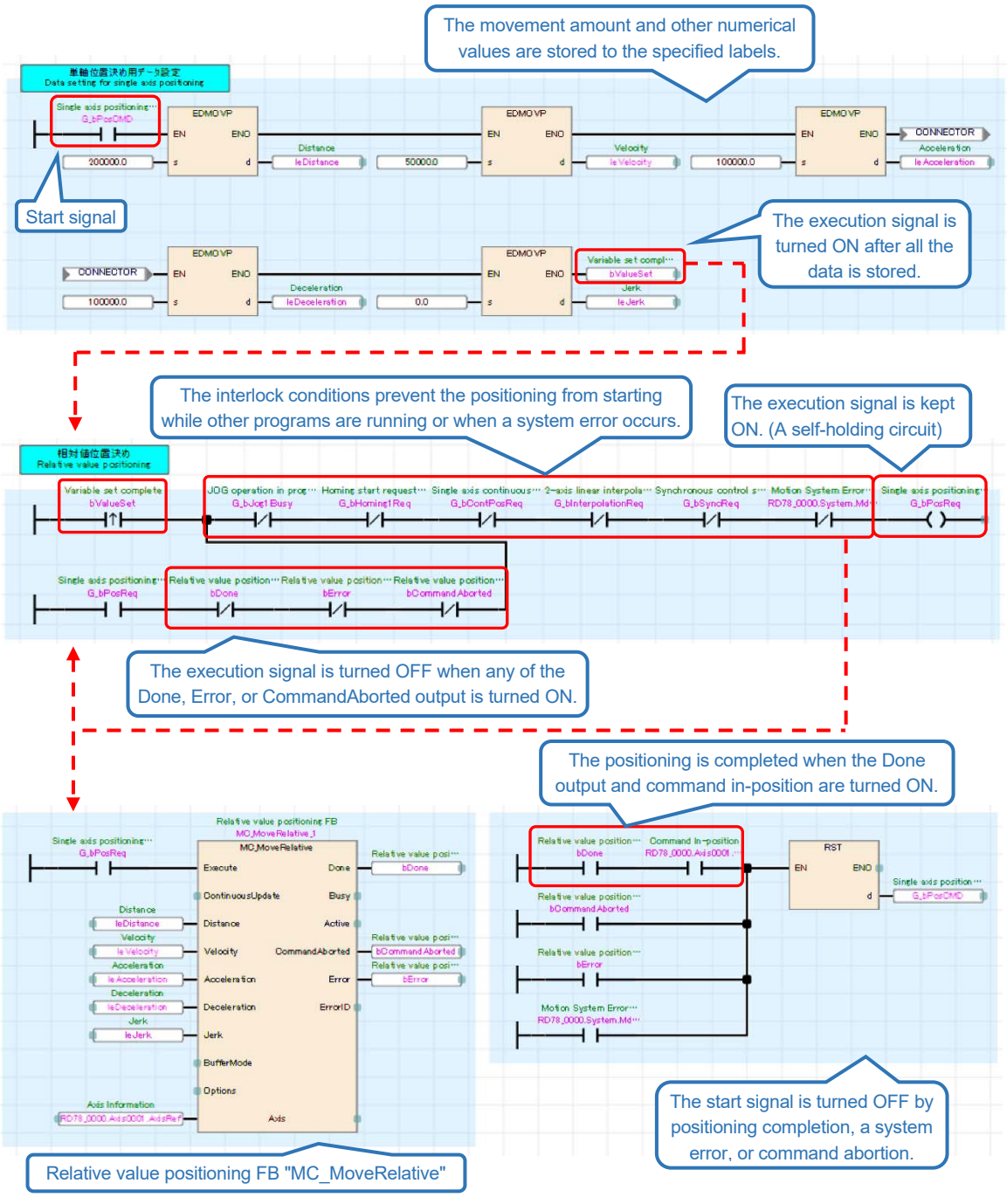
This program executes relative positioning in the following operation pattern.



When the single axis positioning command is turned ON, each positioning data is stored to the specified label. When all the data is stored, the execution signal of the relative value positioning FB (MC_MoveRelative) is kept ON by a self-holding circuit. The start signal is turned OFF when any of the following three conditions is satisfied: the positioning is completed (the Done output of MC_MoveRelative and command in-position); an error occurs; or the execution is aborted. The interlock conditions are added to prevent execution of the single axis positioning while other programs are running or when a system error occurs.

[Start signal]

Single axis positioning command	G_bPosCMD

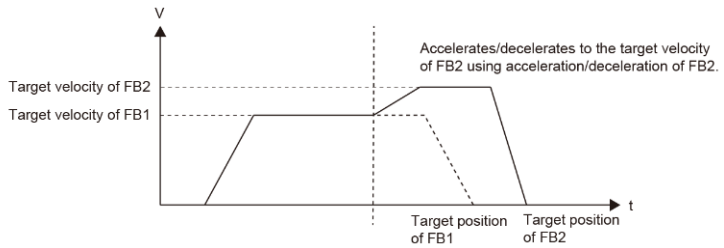
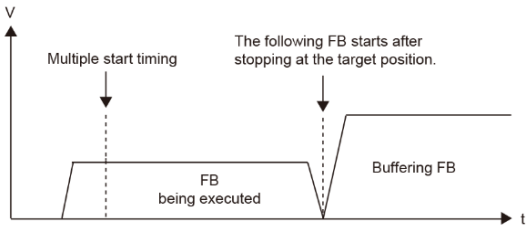
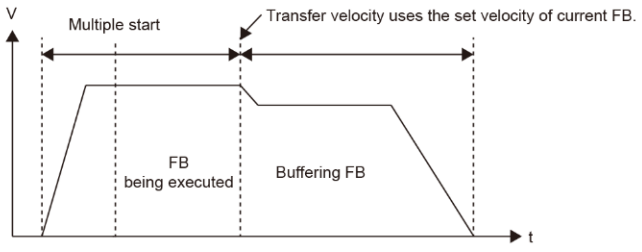
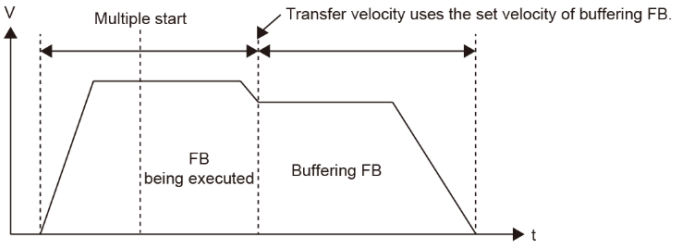


4.9 Single Axis Continuous Positioning (Program Name: ContinuousPositioning)

(1) Overview

An axis can continuously execute multiple motion FBs without stopping by buffering the next motion FB of another instance while executing the first motion FB. Select the buffer mode through the BufferMode input of a motion control FB.
Up to two motion FBs can be buffered on one axis.

(2) Operation pattern of buffer mode

Buffer mode	Operation
0: mcAborting The on-going FB is interrupted (canceled), and the next FB is immediately executed.	
1: mcBuffered The next FB is executed after completion of the on-going FB (the axis decelerates to a stop).	
2: mcBlendingLow The lower target velocity of the on-going FB and the next FB is used as the switching speed.	
3: mcBlendingPrevious The next FB is executed after the target position of the on-going FB is reached. The target velocity is that of the next FB.	
4: mcBlendingNext The next FB is executed after the target position of the on-going FB is reached. When the target position is reached, the velocity is switched to that of the next FB.	
5: mcBlendingHigh The higher target velocity of the on-going FB and the next FB is used as the switching speed.	

(3) Local labels

	Label Name	Data Type		English(Display Target)
1	leDistance1	FLOAT [Double Precision]	...	Distance1
2	leVelocity1	FLOAT [Double Precision]	...	Velocity1
3	leDistance2	FLOAT [Double Precision]	...	Distance2
4	leVelocity2	FLOAT [Double Precision]	...	Velocity2
5	leAcceleration1	FLOAT [Double Precision]	...	Acceleration1
6	leDeceleration1	FLOAT [Double Precision]	...	Deceleration1
7	leAcceleration2	FLOAT [Double Precision]	...	Acceleration2
8	leDeceleration2	FLOAT [Double Precision]	...	Deceleration2
9	leJerk	FLOAT [Double Precision]	...	Jerk
10	MC_MoveRelative_1	MC_MoveRelative	...	Relative value positioning FB1
11	MC_MoveRelative_2	MC_MoveRelative	...	Relative value positioning FB2
12	bError	Bit	...	Relative value positioning FB Error output
13	TON_1	TON	...	On-delay timer FB
14	bDwell_out	Bit	...	Timer output
15	bDwell_in	Bit	...	Timer input
16	bValueSet	Bit	...	Variable set complete
17	bCommandAborted	Bit	...	Relative value positioning FB CommandAborted output
18			...	

2)

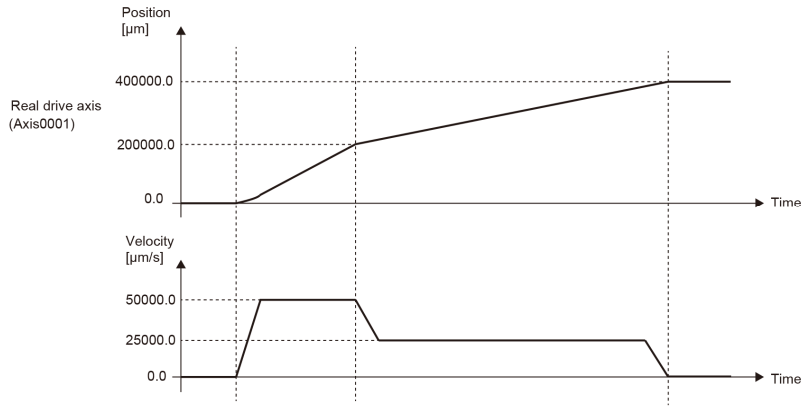
1)

1) These labels are automatically added when the user drags and drops the FB (MC_MoveRelative and TON) to the program editor.

2) These labels are registered manually.

(4) Program example

This program executes relative positioning in the following operation pattern.



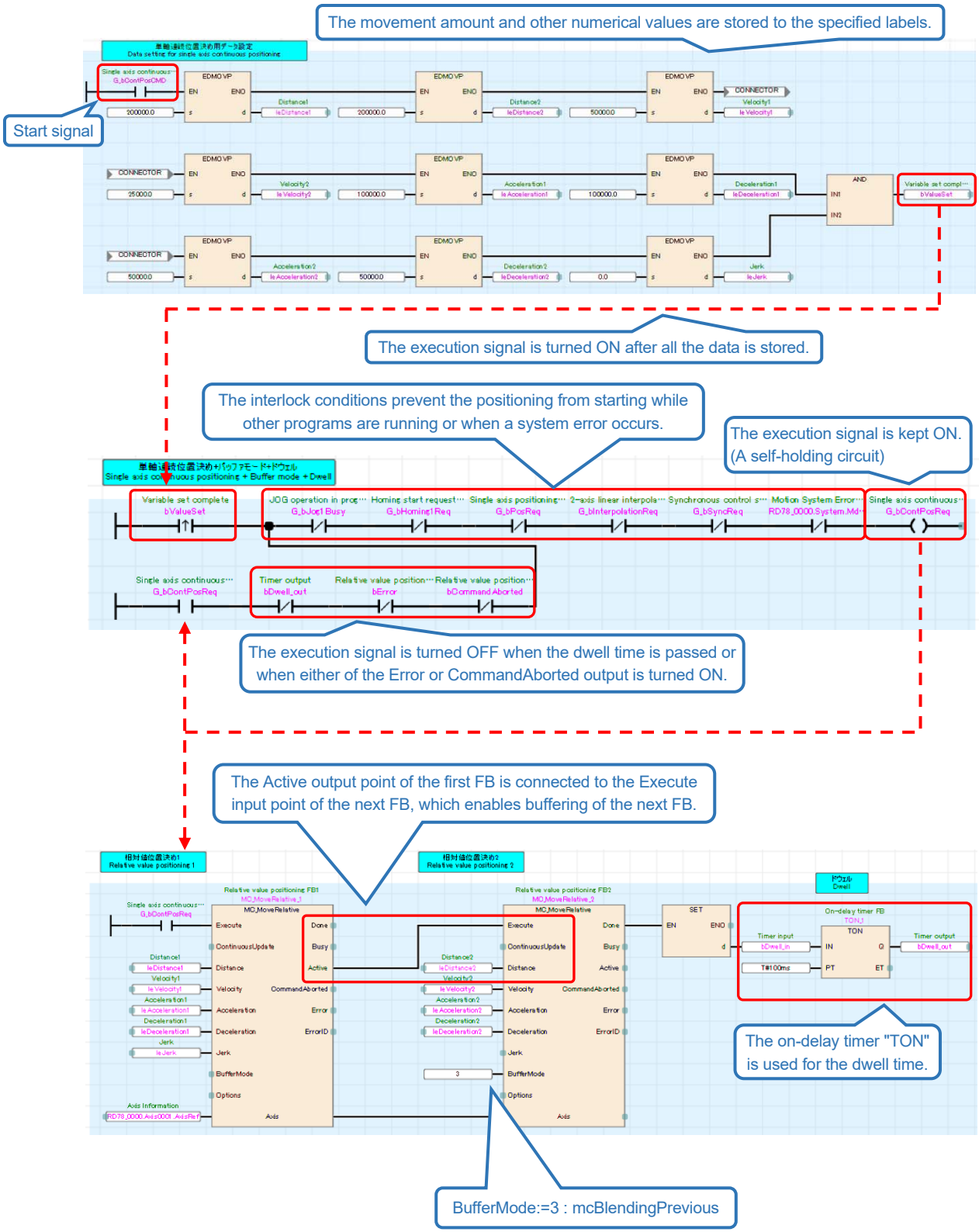
When the single axis positioning command is turned ON, each positioning data is stored to the specified label. When all the data is stored, the execution signal of the FB (MC_MoveRelative_1) is kept ON by a self-holding circuit.

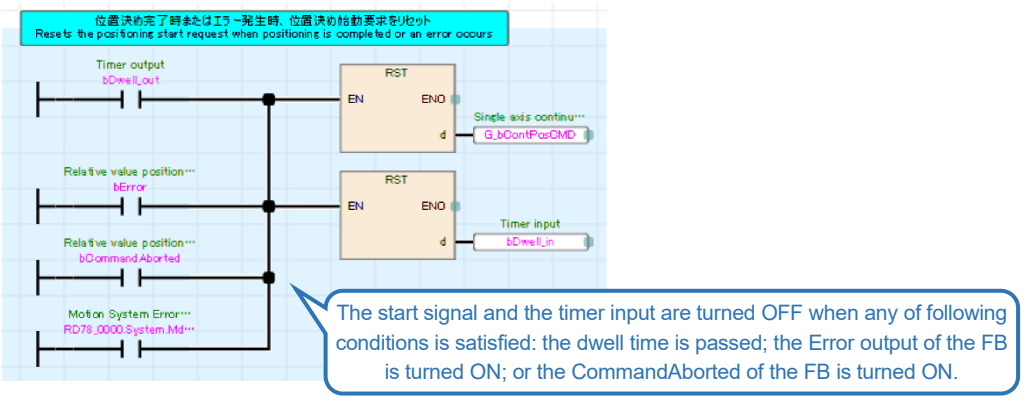
The Active output of the first FB (MC_MoveRelative_1) is connected to the Execute input of the next FB (MC_MoveRelative_2), which enables buffering of the next FB while executing the first FB. When the first FB is finished, the next FB is continuously executed.

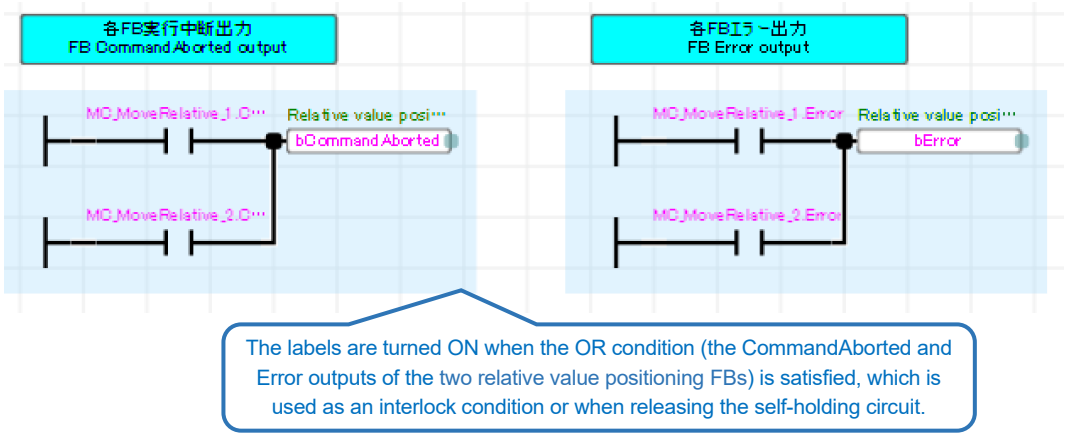
To set dwell time, the on-delay timer (100 [ms]) is used. The on-delay timer input and the start signal are reset when any of the following three conditions is satisfied: the dwell time is passed; an error occurs; or the execution is aborted. The interlock conditions are added to prevent execution of the single axis continuous positioning while other programs are running or when a system error occurs.

[Start signal]

Single axis continuous positioning command	G_bContPosCMD


The movement amount and other numerical values are stored to the specified labels.
Start signal
The execution signal is turned ON after all the data is stored.
The interlock conditions prevent the positioning from starting while other programs are running or when a system error occurs.
The execution signal is kept ON. (A self-holding circuit)
The execution signal is turned OFF when the dwell time is passed or when either of the Error or CommandAborted output is turned ON.
The Active output point of the first FB is connected to the Execute input point of the next FB, which enables buffering of the next FB.
The on-delay timer "TON" is used for the dwell time.
BufferMode:=3 : mcBlendingPrevious





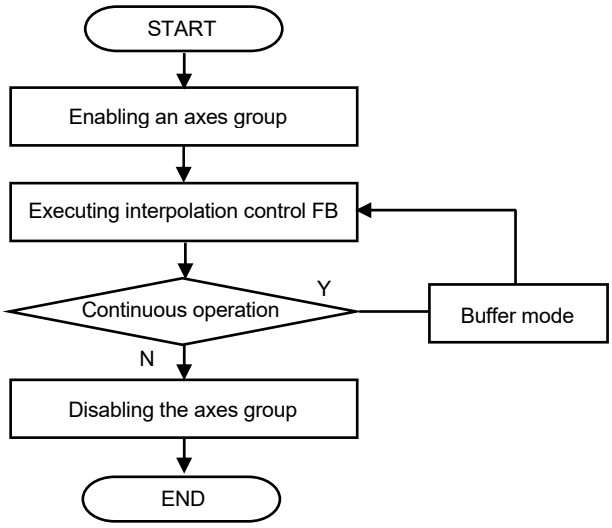
[POINTS]

Instead of connecting a label, output signals of a FB can be directly entered by describing "(FB name). (output signal name)" in the editor.

4.10 Interpolation Control (Program Name: LinearInterpolation)

4.10.1 Procedure for interpolation control

The following shows the procedure flow for executing interpolation control of two or more axes.



4.10.2 Enabling/disabling an axes group

Refer to Section 3.12 for axes group setting.
To execute interpolation control, set the axes group status to "4: GroupStandby".

(1) FBs

Type	Command	Description
MCFB (administrative)	MC_GroupEnable	Transits the specified axes group status from "0: GroupDisabled" to "4: GroupStandby".
	MC_GroupDisable	Transits the specified axes group status to "0: GroupDisabled".

4.10.3 Interpolation control

The FBs for linear and circular interpolation control are as follows. Execute the FBs after the axes group is enabled.

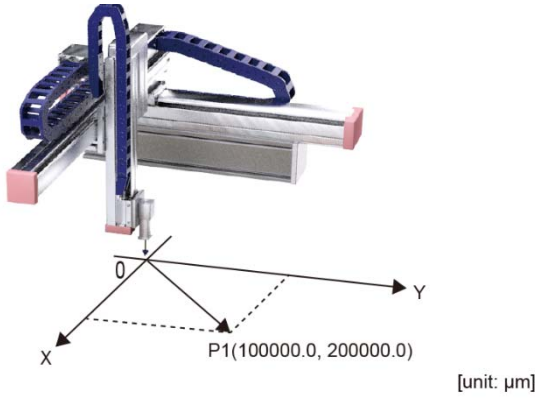
(1) FBs

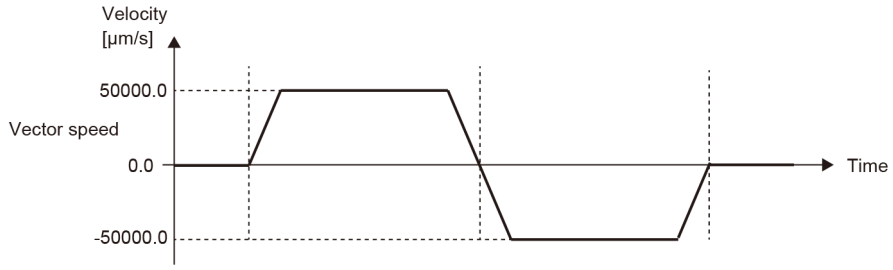
Type	Command	Description
MCFB (motion)	MCv_MoveLinearInterpolateAbsolute	Absolute value linear interpolation control
	MCv_MoveLinearInterpolateRelative	Relative value linear interpolation control
	MCv_MoveCircularInterpolateAbsolute	Absolute value circular interpolation control
	MCv_MoveCircularInterpolateRelative	Relative value circular interpolation control

4.10.4 Program example for linear interpolation

(1) Operation pattern

The machine goes back and forth between the origin point (0.0, 0.0) [um] and P1 (100000.0, 200000.0) [um].





(2) Axis No. and movement amount settings

(a) LinearAxes input of MCv_MoveLinearInterpolateRelative

Use the array of INT (signed word) type with 16 elements.

In the sample program, wAxes[0..15] (label) is used.

The interpolation control axes are selected from the structuring axis [1] to [16] set in the axes group (in Section 3.12). Specify the structuring axis No. to be used for the interpolation control by wAxes. (Note that the wAxes must be set from the index No. [0] in order.)

(b) Distance input

Use the array of LREAL (double-precision real number) type with 16 elements

In the sample program, lePosition[0..15] is used.

Define the movement amount of the structuring axis [1] to [16] in lePosition[0] to lePosition[15].

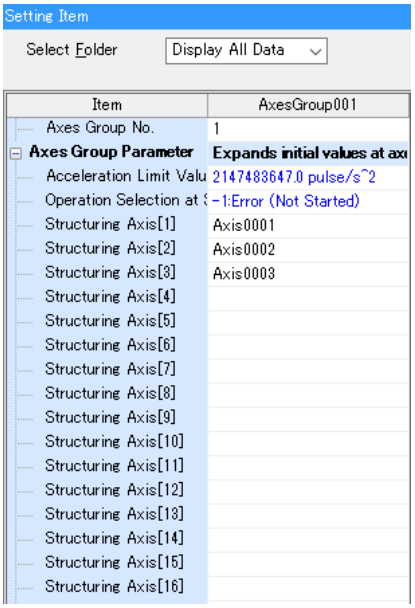
[POINTS]

Regardless of the number of the interpolation control axes, the number of elements must be 16 for the INT-type array for the LinearAxes input and the LREAL-type array for the Distance input.

[Setting example 1]

• Axes group
Structuring axis [1]: Axis0001, structuring axis [2]: Axis0002,
and structuring axis [3]: Axis0003

• Linear interpolation
Axis0001 and Axis0002

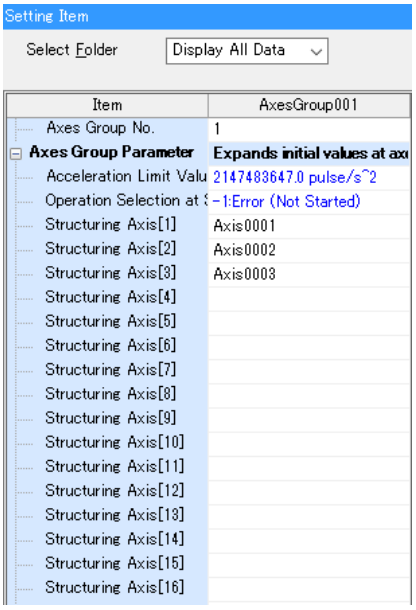



```
wAxes[0] := 1; (← structuring axis [1])
wAxes[1] := 2; (← structuring axis [2])

lePosition[0] := (movement amount of structuring
                 axis 1 (=Axis0001));
lePosition[1] := (movement amount of structuring
                 axis 2 (=Axis0002));
```

[Setting example 2]

• Axes group
Structuring axis [1]: Axis0001, structuring axis [2]: Axis0002,
and structuring axis [3]: Axis0003

• Linear interpolation
Axis0002 and Axis0003




```
wAxes[0] := 2; (← structuring axis [2])
wAxes[1] := 3; (← structuring axis [3])

lePosition[0] := 0.0;
lePosition[1] := (movement amount of structuring
                 axis 2 (=Axis0002));
lePosition[2] := (movement amount of structuring
                 axis 3 (=Axis0003));
```

(3) Local labels

2)

1)

	Label Name	Data Type		English(Display Target)
1	wAxes	Word [Signed](0..15)	...	Interpolation axis
2	lePosition1	FLOAT [Double Precision](0..15)	...	Position data 1
3	lePosition2	FLOAT [Double Precision](0..15)	...	Position data 2
4	leVelocity	FLOAT [Double Precision]	...	Velocity
5	leAcceleration	FLOAT [Double Precision]	...	Acceleration
6	leDeceleration	FLOAT [Double Precision]	...	Deceleration
7	leJerk	FLOAT [Double Precision]	...	Jerk
8	MC_GroupEnable_1	MC_GroupEnable	...	Axes group enable FB
9	MCv_MoveLinearInterpolateRelative_1	MCv_MoveLinearInterpolateRelative	...	Relative value linear interpolation control FB1
10	MCv_MoveLinearInterpolateRelative_2	MCv_MoveLinearInterpolateRelative	...	Relative value linear interpolation control FB2
11	MC_GroupDisable_1	MC_GroupDisable	...	Axes group disable FB
12	bDone2	Bit	...	Relative value linear interpolation control FB2 Done output
13	bGrpEnError	Bit	...	Axes group enable FB Error output
14	bError	Bit	...	Relative value linear interpolation control FB Error output
15	bGrpDsblDone	Bit	...	Axes group disable FB Done output
16	TON_1	TON	...	On-delay timer FB
17	bDwell_in	Bit	...	Timer input
18	bDwell_out	Bit	...	Timer output
19	bValueSet	Bit	...	Variable set complete
20	bCommandAborted	Bit	...	Relative value linear interpolation control FB CommandAborted output
21	bDone_Set	Bit	...	Done
22	bCommandAborted_Set	Bit	...	FB abortion of execution
23			...	

1) These labels are automatically added when the user drags and drops the corresponding FB to the program editor.

2) These labels are registered manually.

(4) Program example

When the two-axis interpolation control start is turned ON, each positioning data is stored to the specified label. When all the data is stored, the execution signal of the FB (MC_GroupEnable_1) is kept ON through a self-holding circuit. After the axes group is enabled by MC_GroupEnable, the two FBs for relative value linear interpolation control (MCv_MoveLinearInterpolateRelative) are started in buffer mode.

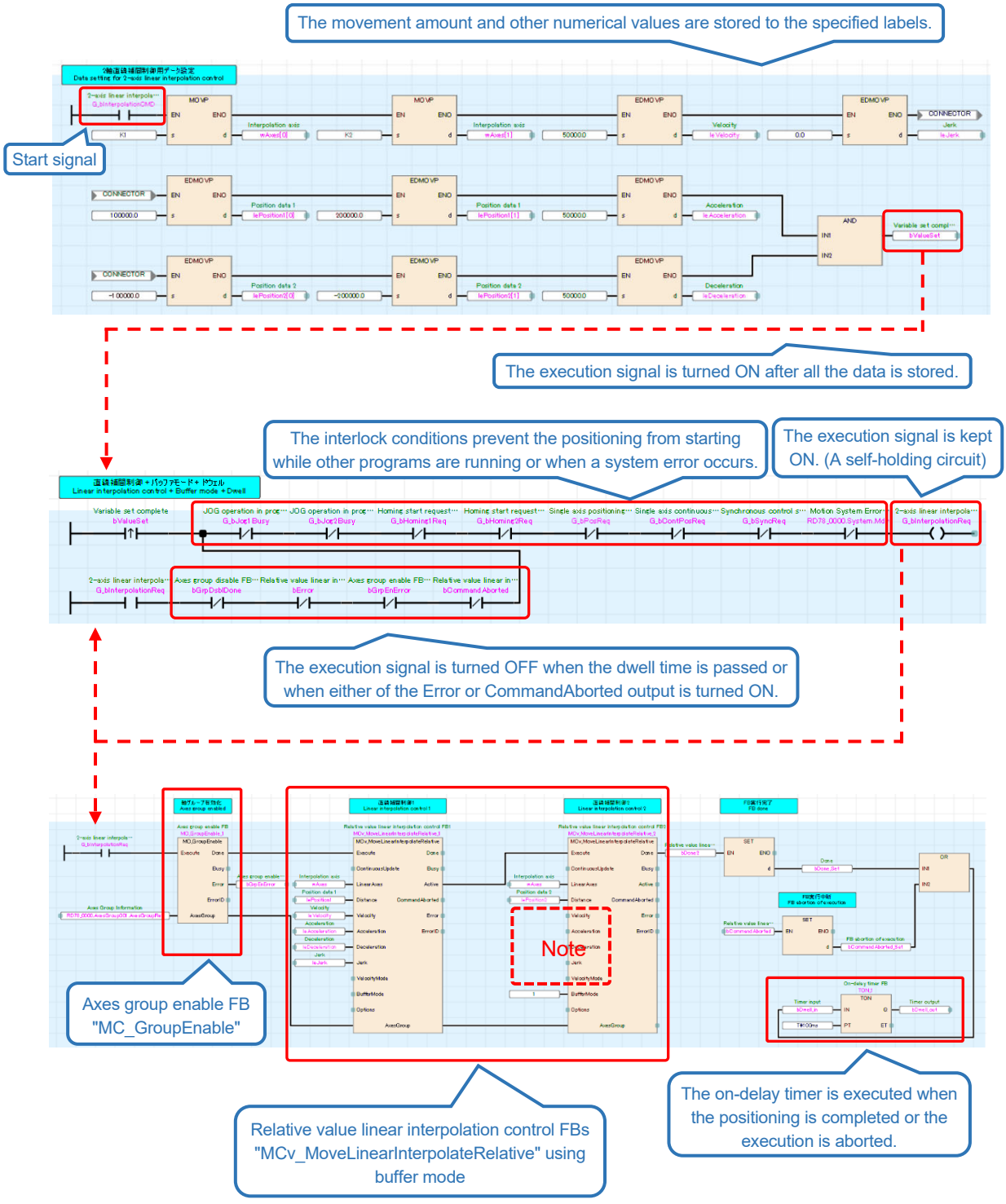
The axes group is disabled when any of the following conditions is satisfied: the dwell time is passed after the positioning is completed; an error occurs; or the execution is aborted.

The on-delay timer and the start signal are reset when the group is disabled.

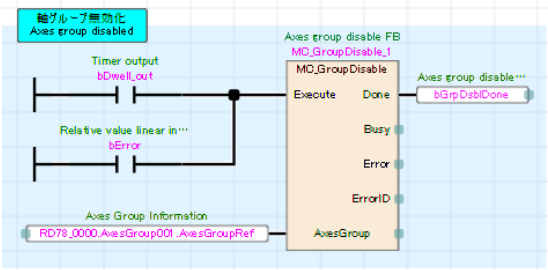
The interlock conditions are added to prevent execution of the two-axis interpolation control while other programs are running or when a system error occurs.

[Start signal]

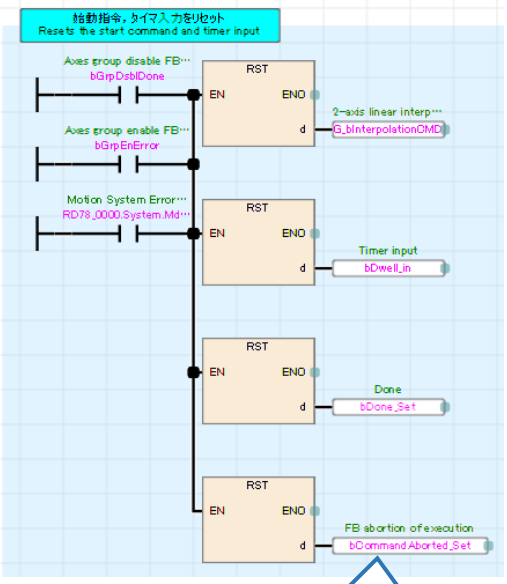
Two-axis linear interpolation control start	G_bInterpolationCMD



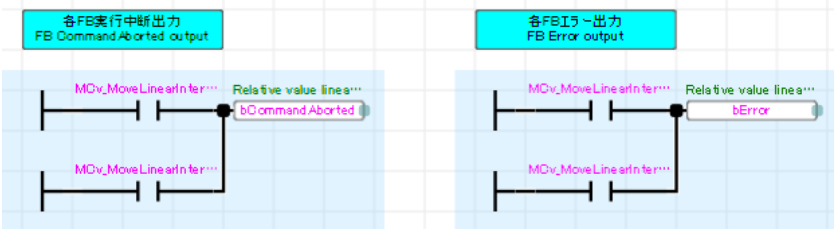
(Note) When the values for the Velocity, Acceleration, and Deceleration inputs are omitted, those of the previous FB are applied.



"MC_GroupDisable" disables the axes group when the dwell time is passed or when the linear interpolation control error is turned ON.



The start signal and the timer input, etc. are turned OFF when any of the following conditions is satisfied.
- Group disable done
- Group enable error
- System error

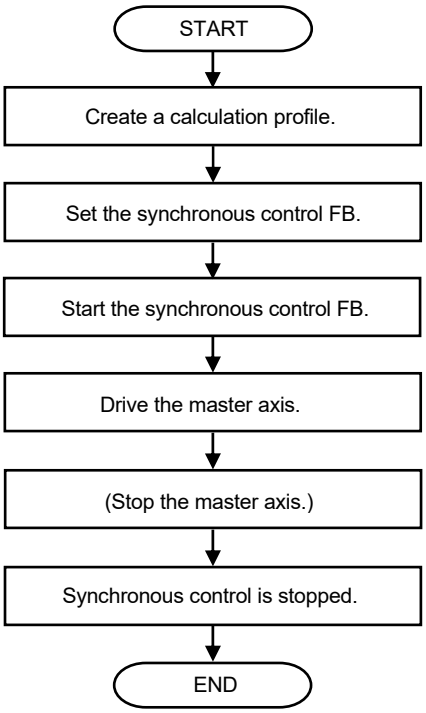


The labels are turned ON when the OR condition (CommandoAborted and Error outputs of the two linear interpolation control FBs) is satisfied, which is used as an interlock condition or when releasing the self-holding circuit.

4.11 Synchronous Control (Program Name: Synchronous)

4.11.1 Procedures for executing synchronous control

The following shows the procedure flow for executing synchronous control.

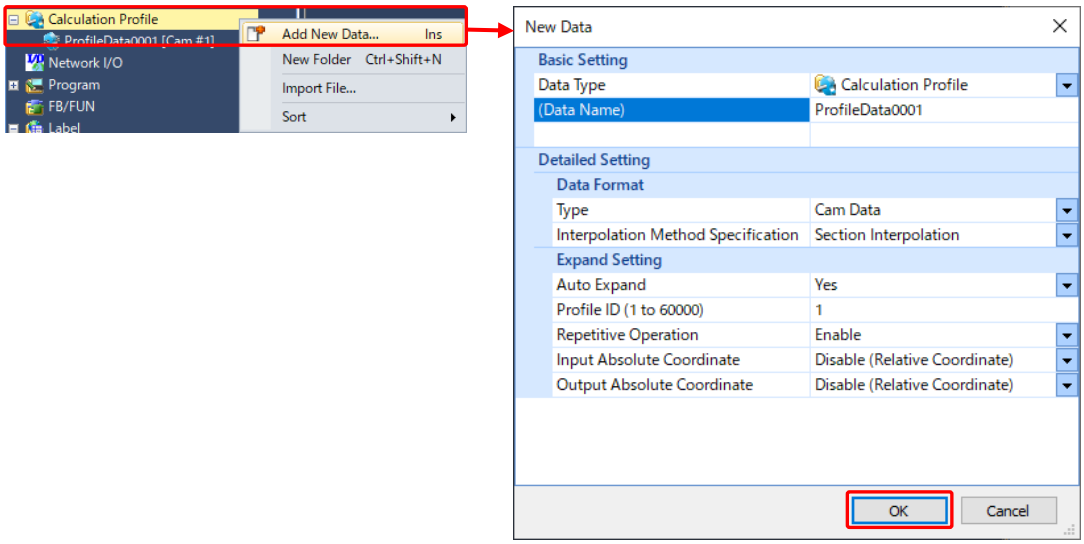


4.11.2 Calculation profile

Waveform data used for control is collectively called calculation profile data.
This section explains how to create a cam data.

(1) Creating a new calculation profile

In the navigation window of the motion control setting function, right-click "Calculation Profile" and select "Add New Data".



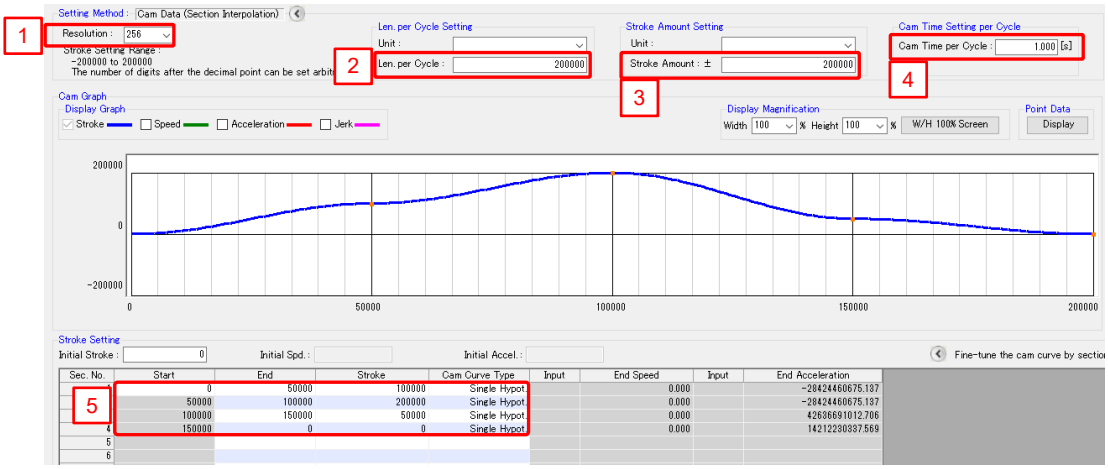
The following shows the setting items on the "New data" screen.

Item	Description
Automatic open	Yes: The calculation profile data is automatically opened at power-on. No: The open FB for calculation profile data needs to be executed.
Periodic	Invalid: The control ends when it executes until the end of calculation profile data. Valid: The execution of calculation profile data is continuously repeated.
Output absolute coordinate	Invalid (relative): When the calculation profile (cam) is started, an output value is calculated based on the current value. When executing a feed cam operation, select this setting. Valid (absolute): The output value at the time the calculation profile (cam) is started is calculated to be always the start point for one cycle of the calculation profile data. When the start point and the end point of the calculation profile data are different, the command is output in one operation cycle in order to return to the first output value at the next one cycle start.

The example in this document uses the initial value. Click the [OK] button.

(2) Creating a cam data

Set the calculation profile waveform.



The following shows the necessary setting items.

No.	Item	Description
1	Resolution	Set the resolution of the cam data.
2	Length per Cycle	Set the length per cycle and its unit. (Set the movement amount of the master axis for one cam cycle)
3	Stroke Amount	Set the stroke amount and its unit. (Set the movement amount of the slave axis for one cam cycle)
4	Cam Time per Cycle	Set the time for one cam cycle. This setting is used when velocity, acceleration, and jerk are calculated.
5	Stroke Setting	Set the stroke.

The following shows the example settings.

No.	Item	Setting value
1	Resolution	256
2	Length per Cycle	200000 (blank for the unit setting)
3	Stroke Amount	200000 (blank for the unit setting)
4	Cam Time per Cycle	1.000
5	Stroke Setting	Refer to the following table.

Section No.	Start point	End point	Stroke	Cam curve type
1	0	50000	100000	Single hypotenuse
2	50000	100000	200000	Single hypotenuse
3	100000	150000	50000	Single hypotenuse
4	150000	0	0	Single hypotenuse

[POINTS]
When executing a linear cam operation (the same operation as the master axis, or the operation that changes the master axis speed based on a specified speed ratio), create a calculation profile for the linear cam, or use MC_GearIn.
The calculation profile data for the linear cam is not provided in the system.

4.11.3 Single axis synchronous control FBs

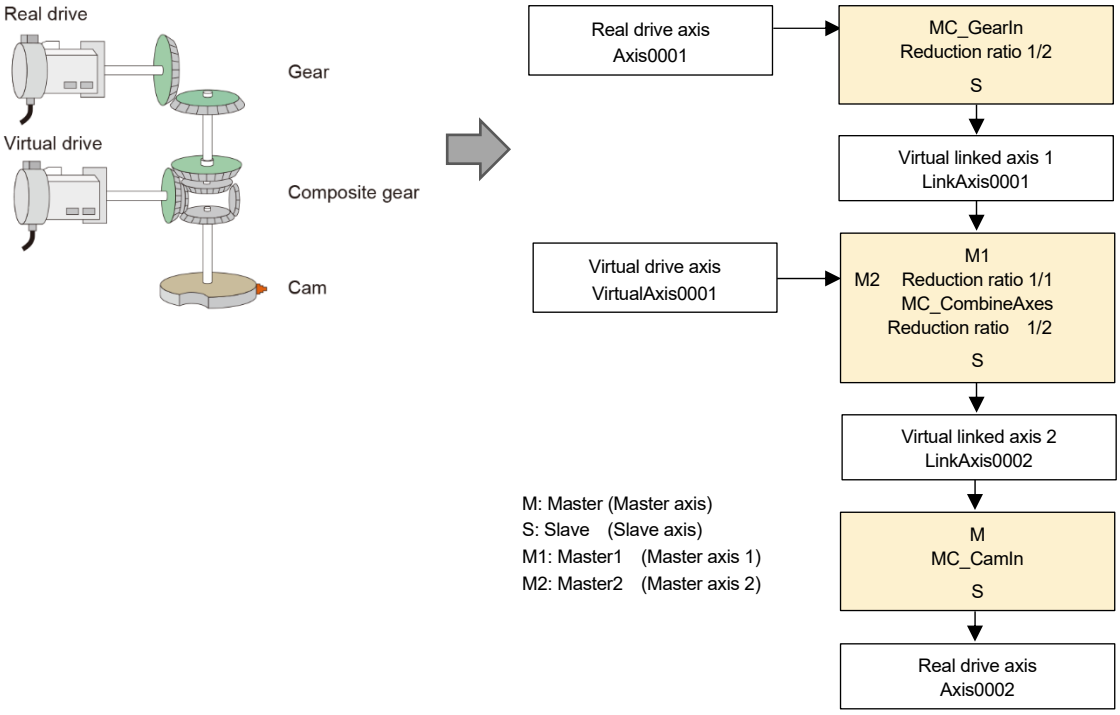
The single axis synchronous control FBs operate as software-based mechanical modules such as gears, speed change gears, and cams. These FBs transmit the position information (command) of Slave that is synchronized with Master.

(1) FBs

Type	Command	Description
MCFB (motion)	MC_CamIn	Executes cam operation.
	MC_GearIn	Executes gear operation based on the specified speed ratio between the master axis and the slave axis.
	MC_CombineAxes	Combines motion of two axes by a selectable combination method, and outputs the result to the third axis.
	MCv_ChangeCycle	Changes the cam current value per cycle to the specified value during MC_CamIn control. It is used to compensate the cam current value per cycle into an arbitrary value.
	MCv_******Filter	Executes the specific filter processing to the input of Master, and outputs the result to Slave.
	MC_Stop	Stops the synchronous control.

4.11.4 Axes configuration

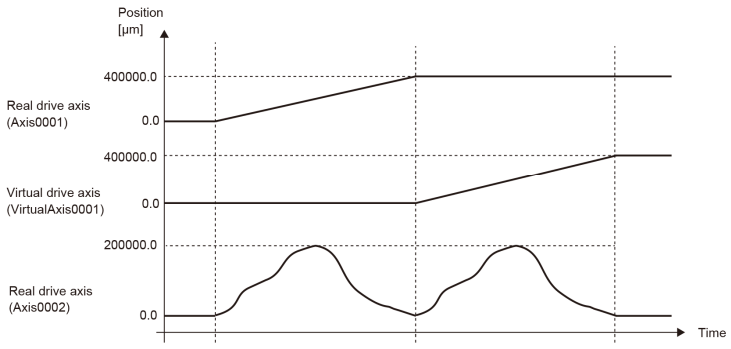
This chapter explains the following cam system.



4.11.5 Program example

(1) Operation pattern

The X-axis moves from the origin point for 400000.0 [um] while the Y-axis (Axis0002) is operated according to the cam pattern created in Section 4.11.2. When the X-axis reaches the target position (400000.0 [um]), the virtual drive axis starts operation and the Y-axis repeats the cam operation. At this time, the X-axis is stopped and only the Y-axis is operated.



(2) Virtual drive axis and virtual linked axis

This program uses the virtual drive axis (VirtualAxis0001) and the virtual liked axes (LinkAxis0001, LinkAxis0002) in addition to the real drive axes (Axis0001, Axis0002).
The AxisRef type structure is used in the PLC CPU program that drives VirtualAxis and LinkAxis. Set each of the AxisRef structures to the public label setting. (Refer to Section 4.3.2.)

(3) Local labels

2)

1)

	Label Name	Data Type		English(Display Target)
1	leVelocity	FLOAT [Double Precision]	...	Velocity
2	leAcceleration	FLOAT [Double Precision]	...	Acceleration
3	leDeceleration	FLOAT [Double Precision]	...	Deceleration
4	leJerk	FLOAT [Double Precision]	...	Jerk
5	lePosition	FLOAT [Double Precision]	...	Distance
6	MC_GearIn_1	MC_GearIn	...	Gear operation FB
7	MC_CombineAxes_1	MC_CombineAxes	...	FB combining the motion of two master axes
8	MC_CamIn_1	MC_CamIn	...	Cam operation FB
9	CamID	MC_CAM_ID	...	Cam ID
10	bInSync	Bit	...	Cam operation FB inSync output
11	MC_MoveRelative_1	MC_MoveRelative	...	Relative value positioning FB
12	bError	Bit	...	Relative value positioning FB Error output
13	bDone1	Bit	...	Relative value positioning FB1 Done output
14	MC_MoveRelative_2	MC_MoveRelative	...	Relative value positioning FB
15	bDone2	Bit	...	Relative value positioning FB2 Done output
16	bStopDone	Bit	...	Axis Stop complete
17	bSyncMove	Bit	...	Relative value positioning start
18	bValueSet	Bit	...	Variable set complete
19	bCommandAborted	Bit	...	Relative value positioning FB CommandAborted output
20	MC_Stop_1	MC_Stop	...	Axis Stop FB1
21	MC_Stop_2	MC_Stop	...	Axis Stop FB2
22	MC_Stop_3	MC_Stop	...	Axis Stop FB3
23			...	

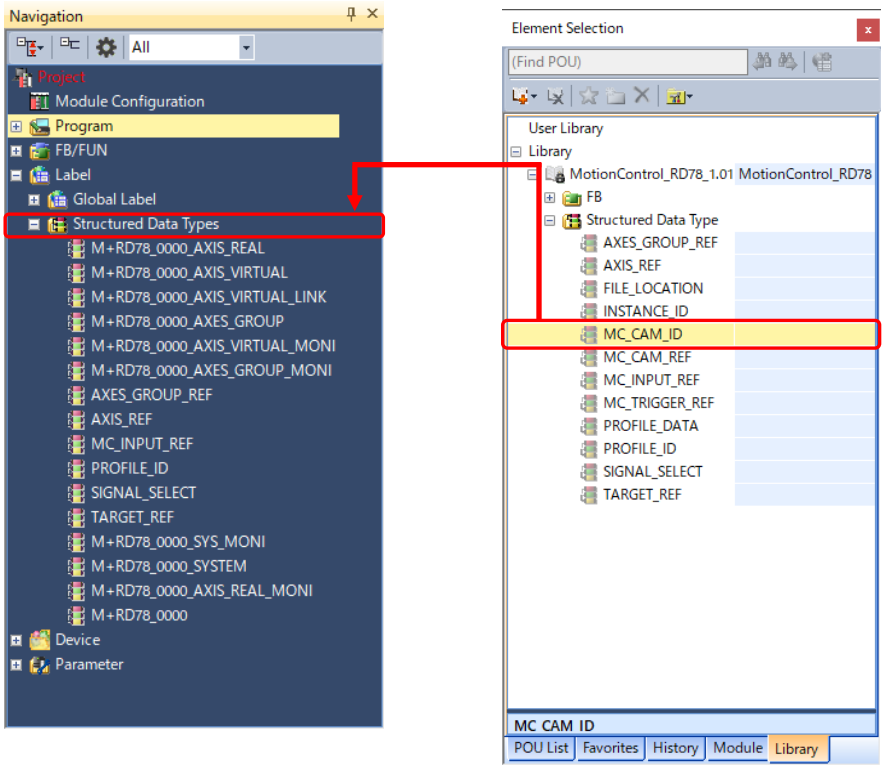
Note

1) These labels are automatically added when the user drags and drops the corresponding FB to the program editor.

2) These labels are registered manually.

(Note): Registering MC_CAM_ID type structure

Click the [Library] tab in the element selection window, and select "Library" → "MotionControl_RD78_****" → "Structured Data Type". Drag and drop "MC_CAM_ID" to "Structured Data Types" under "Label" in the navigation window. The structure is registered under the "Structured Data Types" tree and available as a data type on the label editor.

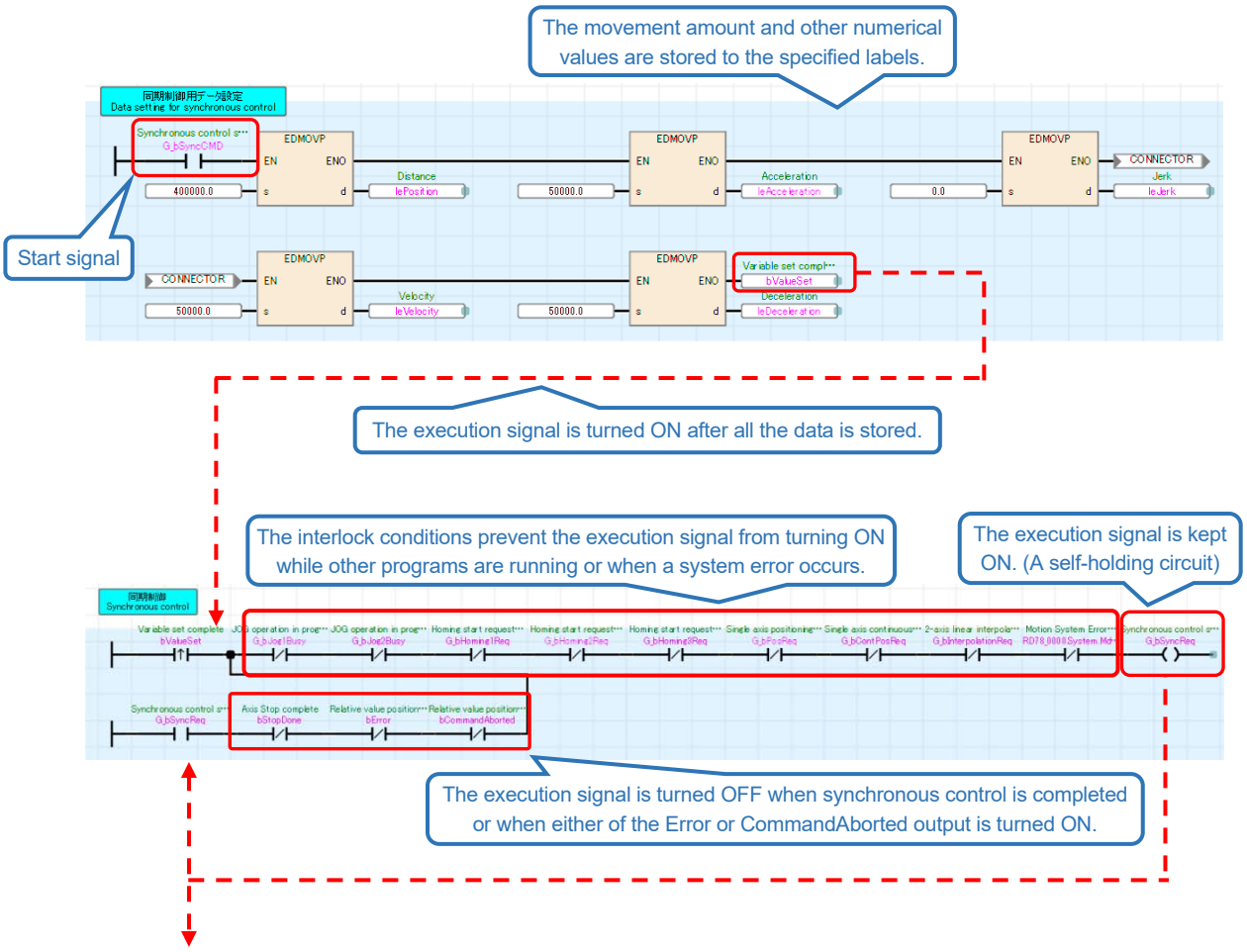


(4) Program example

When the synchronous control start is turned ON, each positioning data is stored to the specified label. When all the data is stored, the execution signal of the FB (MC_GearIn, MC_CombineAxes, and MC_CamIn) is kept ON by a self-holding circuit.
When the Axis0002 status is turned to "7: SynchronizedMotion", Axis0001 (Master axis) starts positioning. At this time, Axis0002 is operated according to the specified calculation profile.
When positioning of Axis0001 is completed, the virtual drive axis (VirtualAxis0001) starts positioning. At this time, Axis0001 is stopped, but Axis0002 is operated according to the specified calculation profile. The synchronous control is stopped by executing MC_Stop on the real drive axis (Axis0002) and the virtual linked axes (LinkAxis0001 and LinkAxis0002) when any of the following conditions is satisfied: the positioning of the virtual drive axis (VirtualAxis0001) is completed; an error occurs; or the execution is aborted. The start signal is reset when the real drive axis (Axis0002) status is out of synchronization.
The interlock conditions are added to prevent execution of the synchronous control while other programs are running or when a system error occurs.

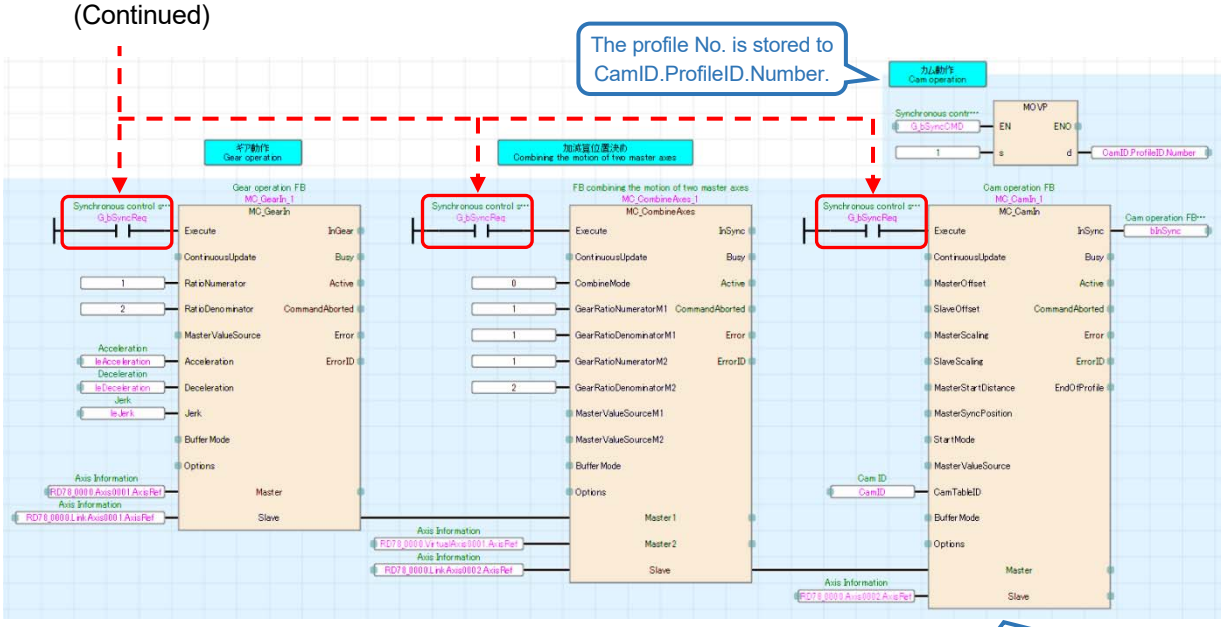
[Start signal]

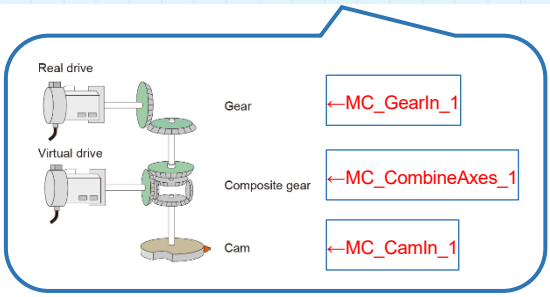
Synchronous control start	G_bSyncCMD



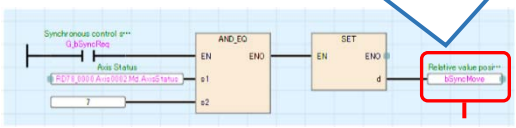
(To the next page)

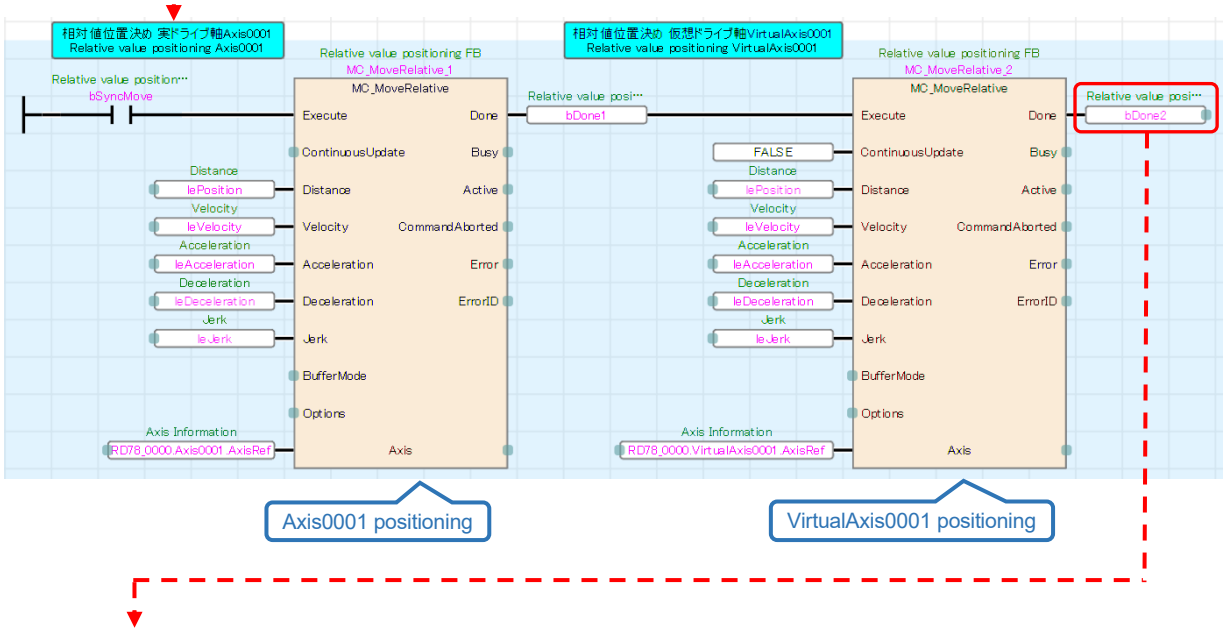
(Continued)





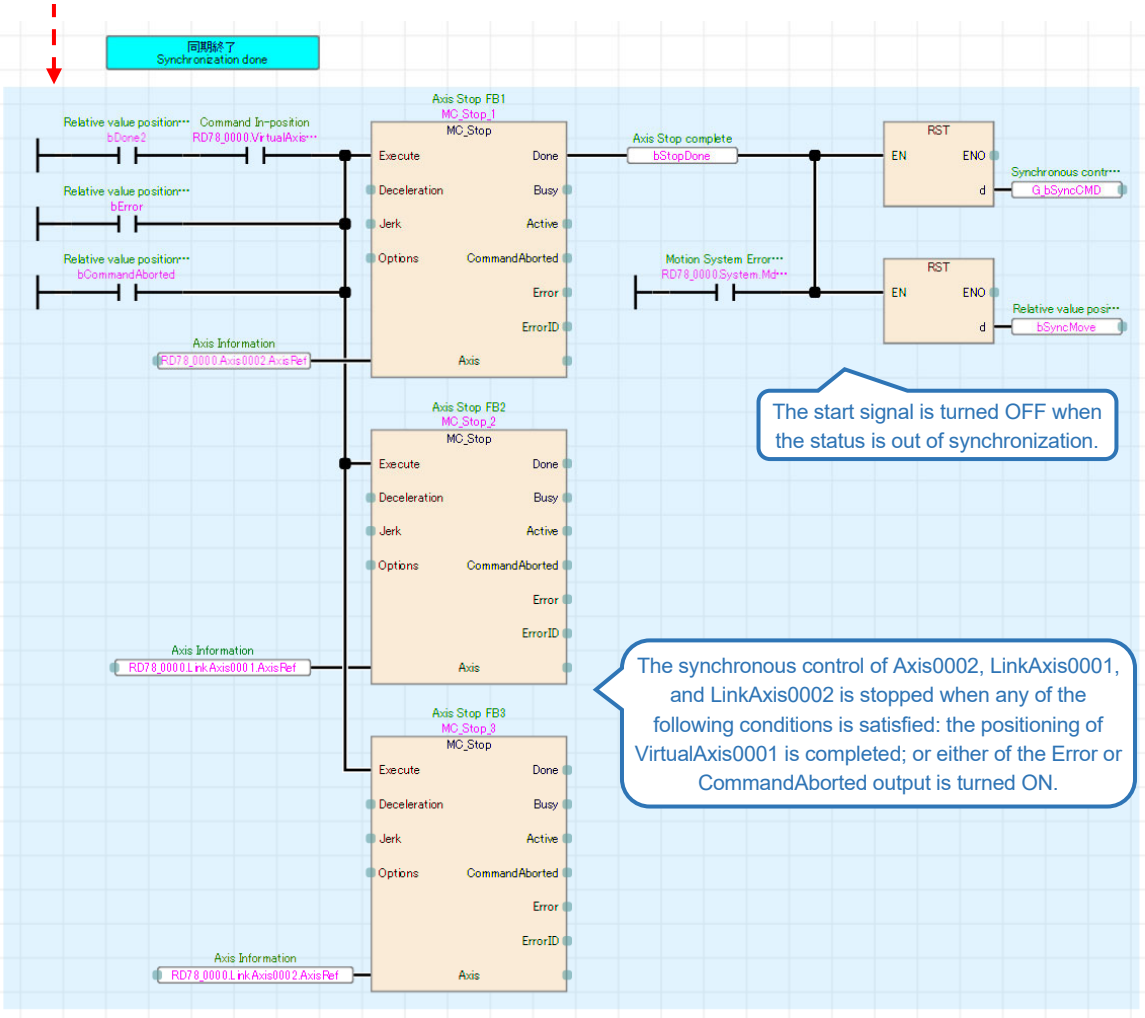
Axis0001 executes positioning when the status of Axis0002 turns to "7: SynchronizedMotion".

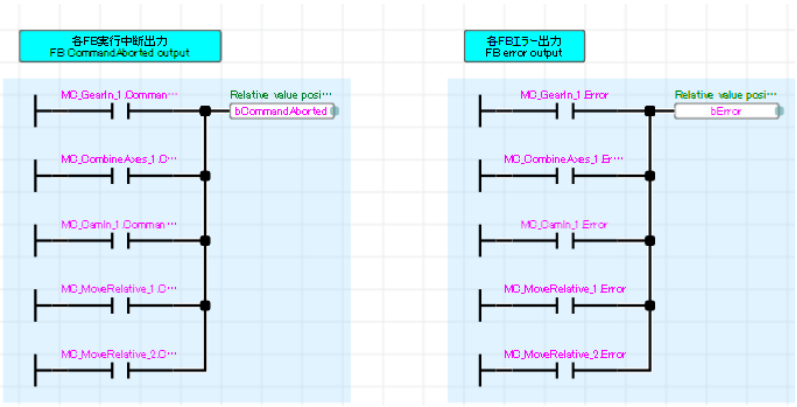




(To the next page)

(Continued)





The labels are turned ON when the OR condition (the CommandoAborted and Error outputs of the synchronous control FBs and the positioning FBs) is satisfied, which is used as an interlock condition or when releasing the self-holding circuit.

4.12 Error Reset (Program Name: ErrorReset)

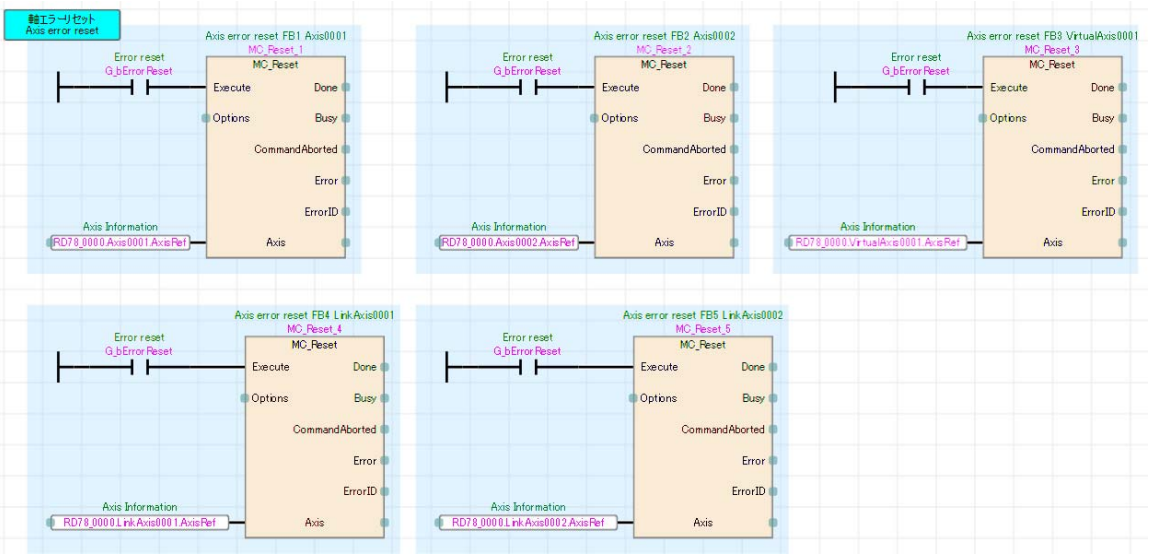
This program resets the error on each axis.

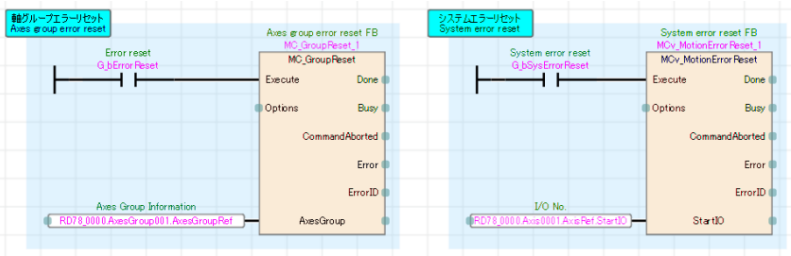
(1) FBs

Type	Command	Description
MCFB (administrative)	MC_Reset	Resets errors and warnings of the axis.
	MC_GroupReset	Resets errors and warnings of the axes group.
	MCv_MotionErrorReset	Reset all errors and warnings of the motion system.

(2) Program example

The following shows the program example for error reset.
When the error reset label is turned ON, MC_Reset and MC_GroupReset are executed.
When the system error reset label is turned ON, MCv_MotionErrorReset is executed.

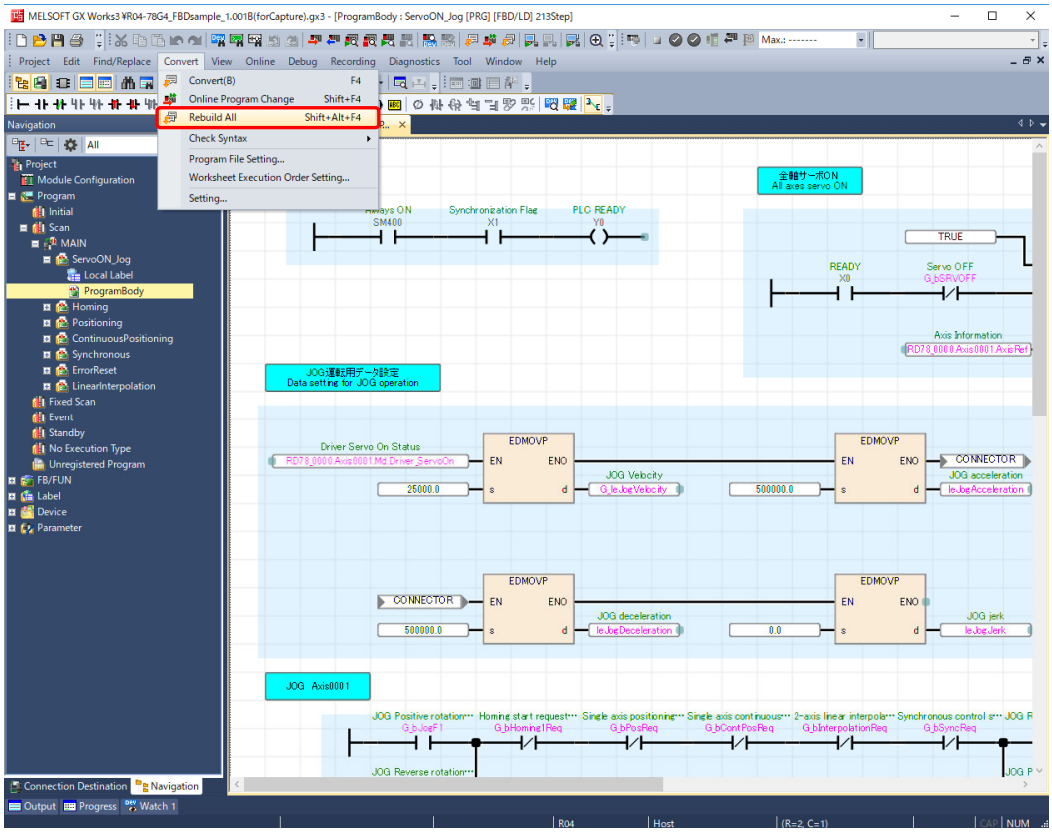




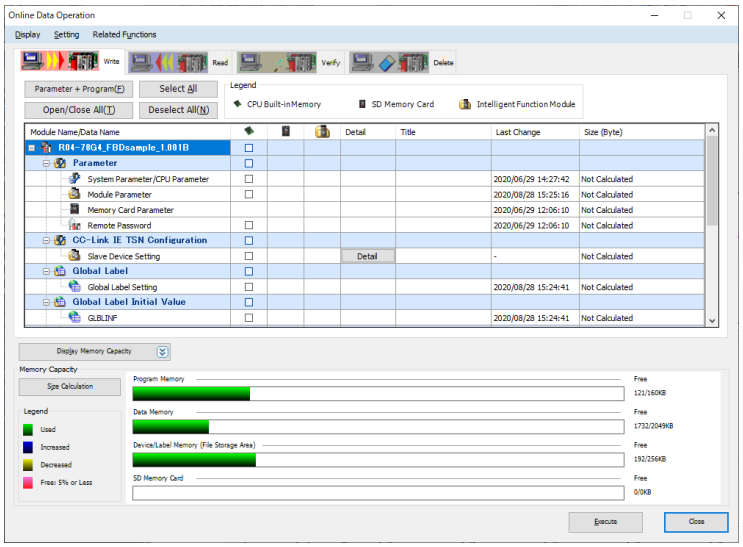
4.13 Checking Operation

4.13.1 Conversion and Writing of Programs

Write the program to the PLC CPU. Select [Convert] → [Rebuild All] to convert all the program.



Confirm that no error occurs after executing "Rebuild All". Select [Online] → [Write to PLC], and write the program to the PLC CPU.
When the parameters and the public labels of the Motion module are changed, write the program after converting all the program and reflecting the public labels in the motion control setting function.



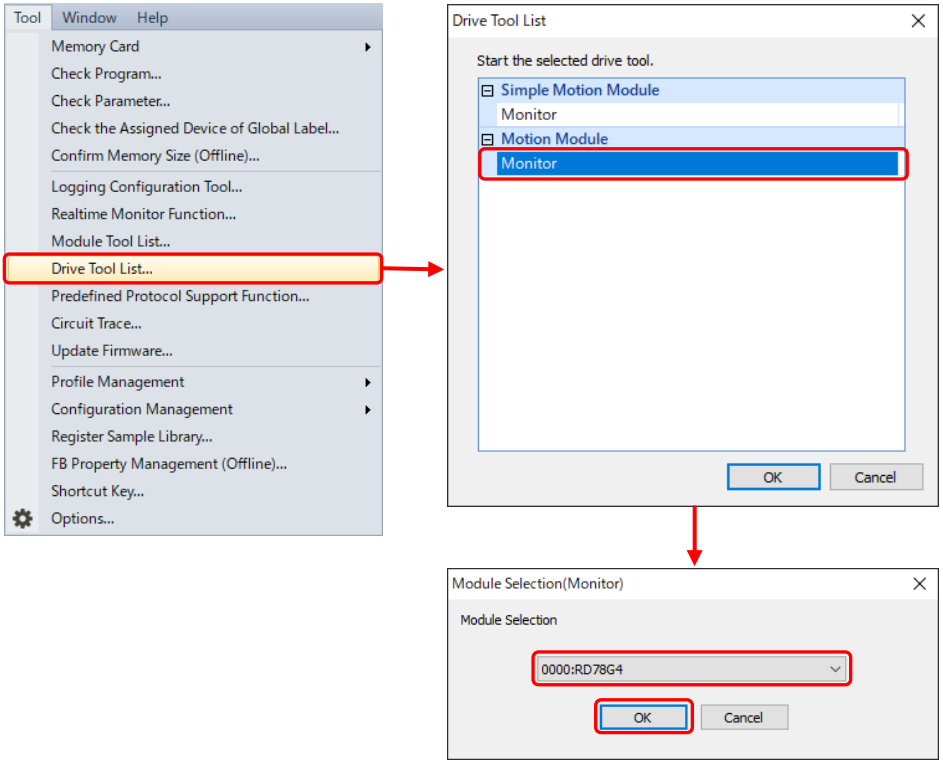
4.13.2 Axis monitor

The monitor screen displays the current values and the error codes of all axes in operation all at once. Users can check the current values and whether any error occurs during the operation.

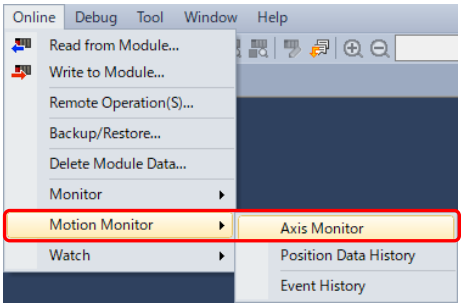
(1) Displaying monitor screens

The monitor screen can be displayed by the following two methods.

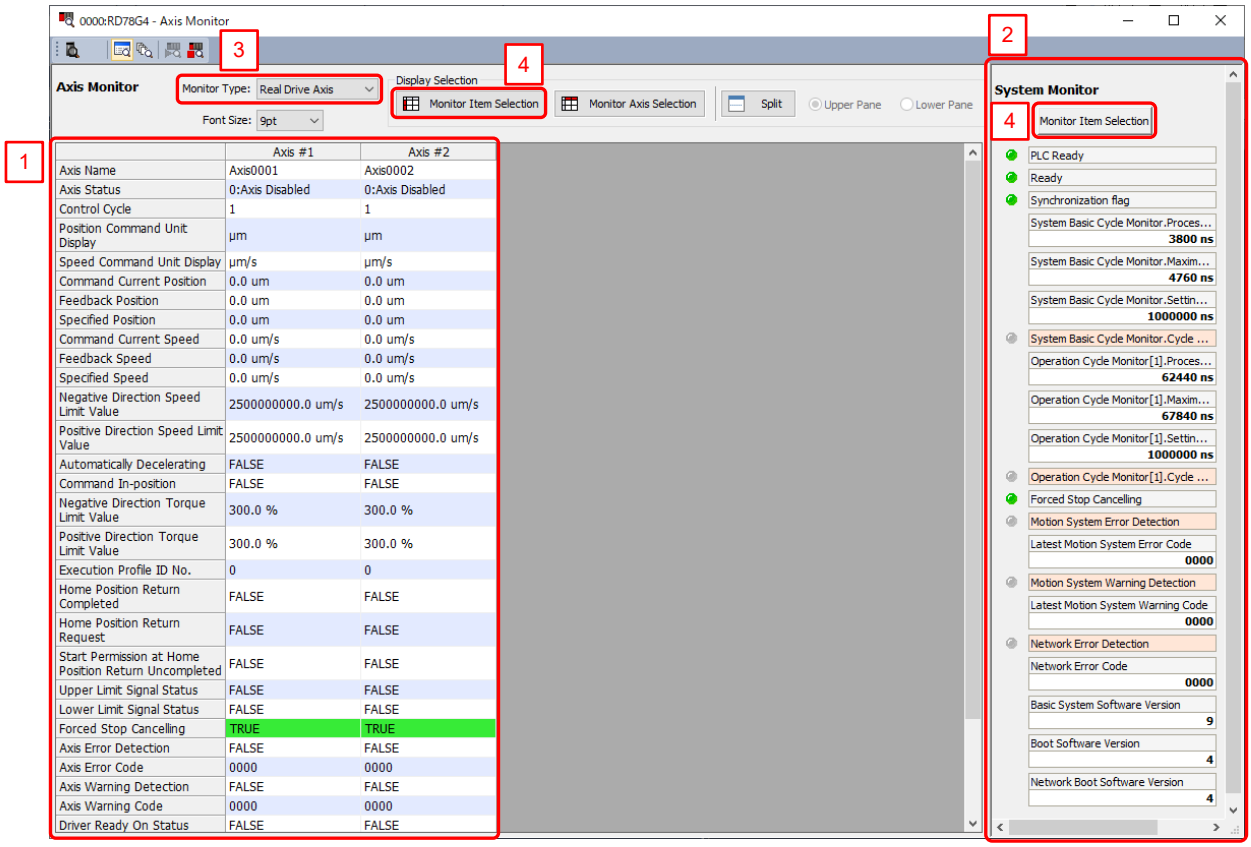
1) Select [Tool] → [Drive Tool List] in MELSOFT GX Works3, and double-click "Monitor" on the "Drive Tool List" screen. Select the Motion module on the "Module Selection (Monitor)" screen, and click the [OK] button.



2) In the motion control setting function, select [Online] → [Motion Monitor] → [Axis Monitor].



(2) Monitor screen



No.	Description
1	The monitor items for each axis are displayed.
2	The system monitor items are displayed.
3	The axis type to be monitored can be changed.
4	The monitor items can be added/deleted.

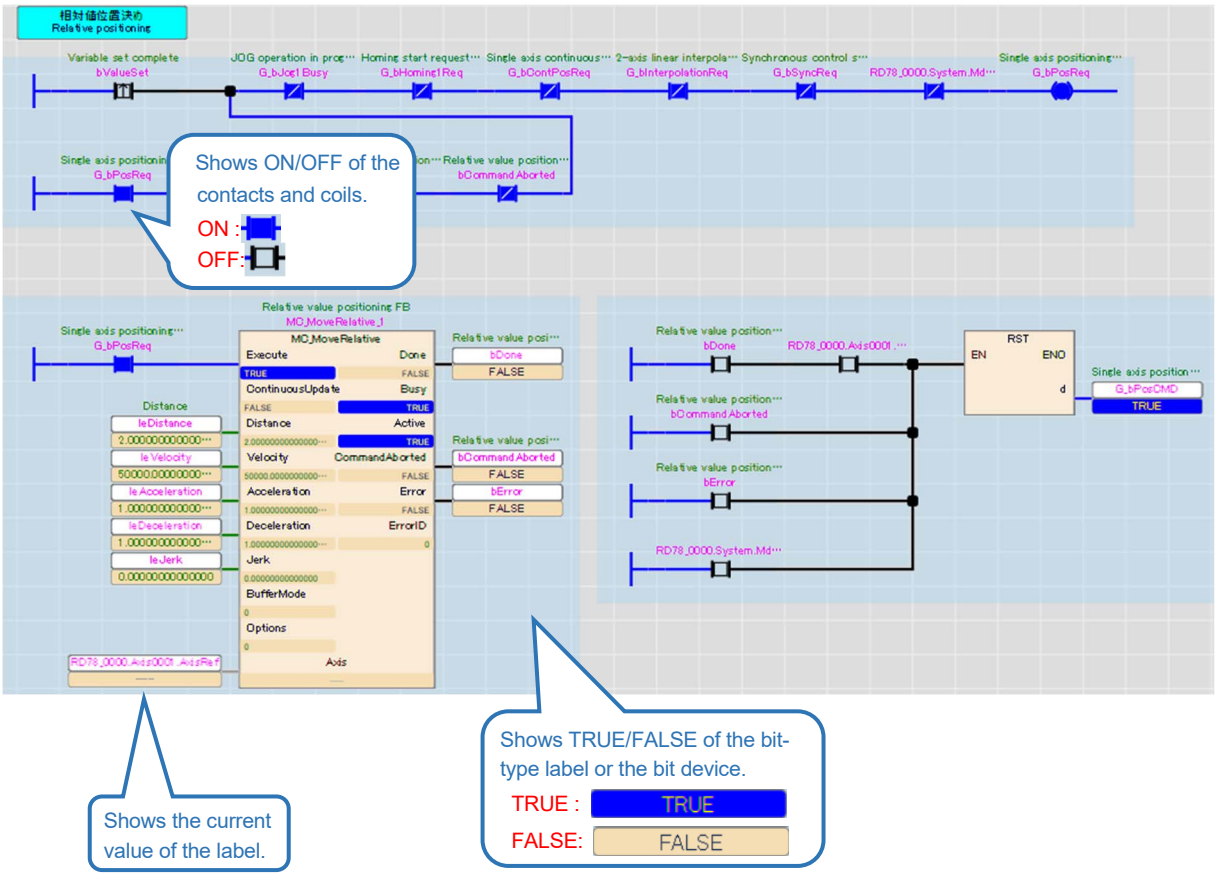
4.13.3 Program monitor

The program monitor enables the user to check the current status of the program in execution on the program editor.

(1) Displaying monitor screen

Select [Online] → [Monitor] → [Start Monitoring (All Windows)], or click the icon [ ] on the tool bar.

(2) Monitor screen

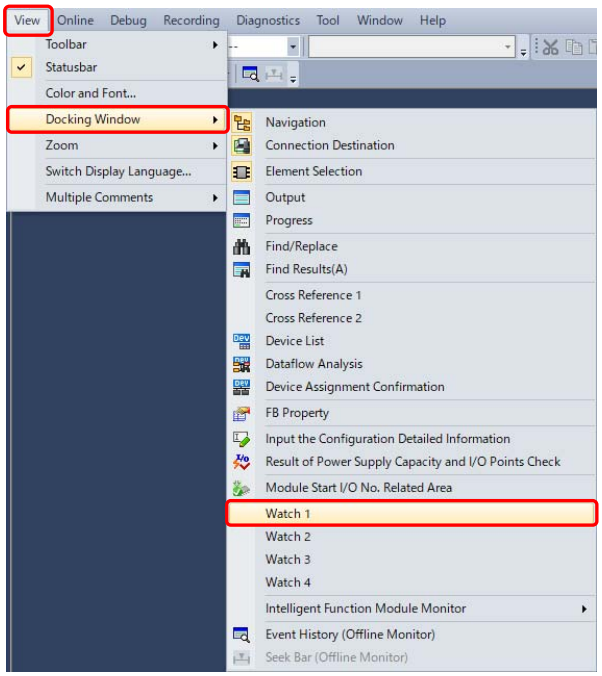


4.13.4 Watch

The watch function enables the user to check the current values of the registered devices and labels. Register the devices and labels to be monitored to a watch window.

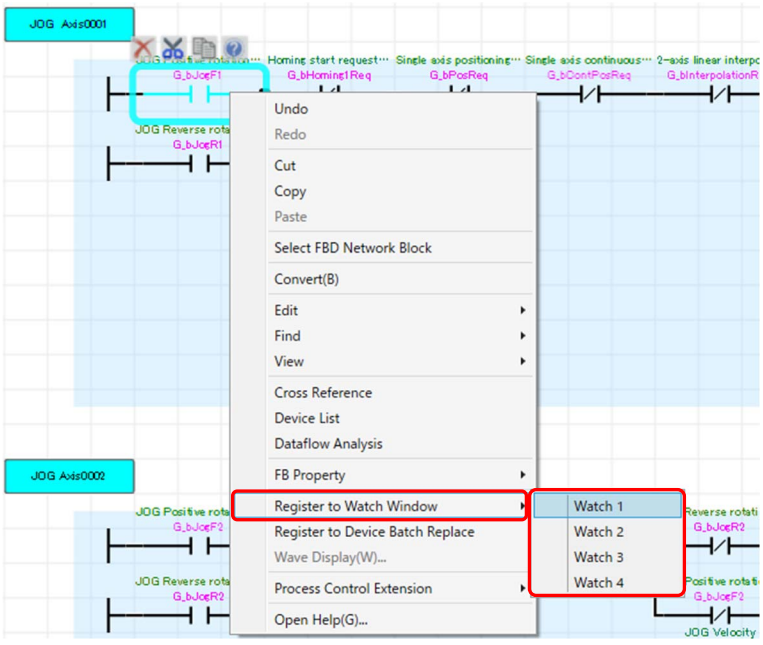
(1) Displaying monitor screens

Select [View] → [Docking Window] → [Watch 1 to Watch 4] in MELSOFT GX Works3.



(2) Registration of devices/labels/structures to a watch window

Enter the device No./label/structure to be registered in the "Name" column of a watch window or right-click a label or a structure to be registered on the program editor, and select [Register to Watch Window] → [Watch 1 to 4].



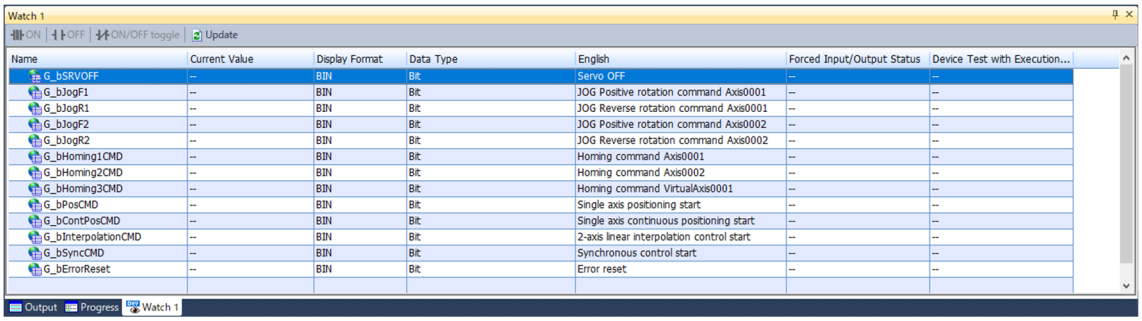
(3) Monitoring start

Select [Online] → [Watch] → [Start Watching] in MELSOFT GX Works3.

(4) Changing current values

Directly enter a value in "Current Value" during monitoring.

The bit device can be switched to ON/OFF with double-click while holding the [Shift] key or with [Shift] + [Enter] after selecting the row.



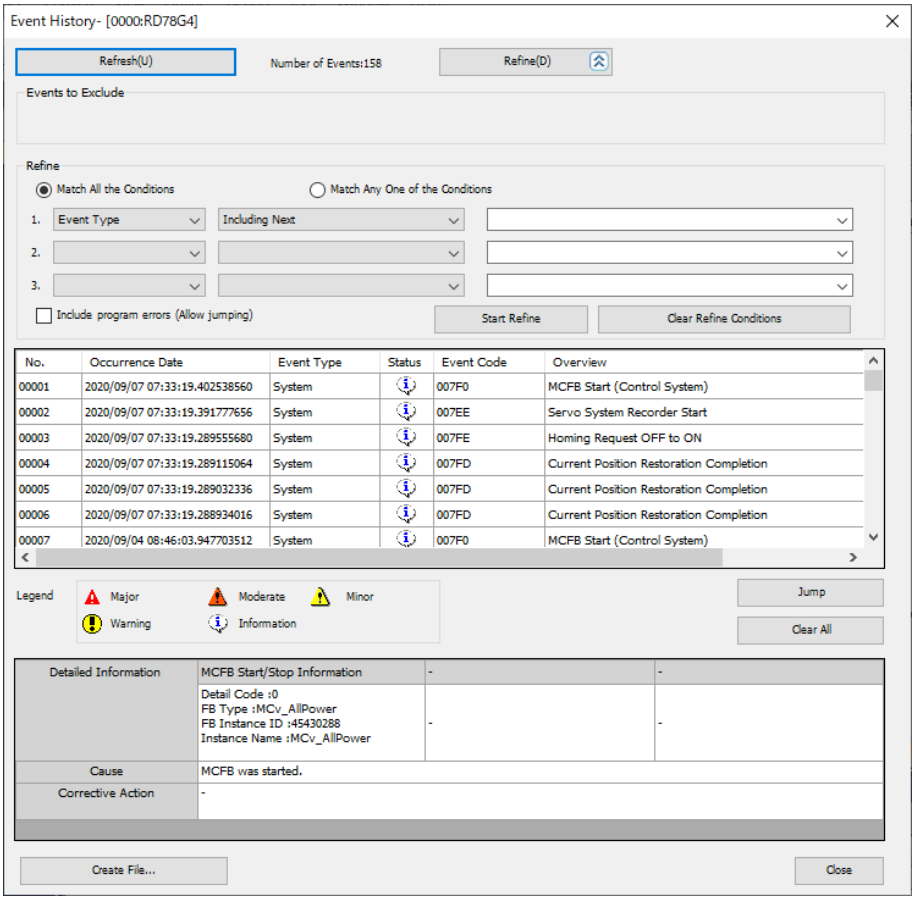
Name	Current Value	Display Format	Data Type	English	Forced Input/Output Status	Device Test with Execution...
G_bSRVOFF	--	BIN	Bit	Servo OFF	--	--
G_bJogF1	--	BIN	Bit	JOG Positive rotation command Axis0001	--	--
G_bJogR1	--	BIN	Bit	JOG Reverse rotation command Axis0001	--	--
G_bJogF2	--	BIN	Bit	JOG Positive rotation command Axis0002	--	--
G_bJogR2	--	BIN	Bit	JOG Reverse rotation command Axis0002	--	--
G_bHoming1CMD	--	BIN	Bit	Homing command Axis0001	--	--
G_bHoming2CMD	--	BIN	Bit	Homing command Axis0002	--	--
G_bHoming3CMD	--	BIN	Bit	Homing command VirtualAxis0001	--	--
G_bPosCMD	--	BIN	Bit	Single axis positioning start	--	--
G_bContPosCMD	--	BIN	Bit	Single axis continuous positioning start	--	--
G_bInterpolationCMD	--	BIN	Bit	2-axis linear interpolation control start	--	--
G_bSyncCMD	--	BIN	Bit	Synchronous control start	--	--
G_bErrorReset	--	BIN	Bit	Error reset	--	--

[POINTS]

In the sample program, "RD78_0000.G_bStopSignalX" and "RD78_0000.G_bStopSignalY" are registered in the watch window 1. Turning these signals ON aborts the operation.

4.13.5 Event History

The event history of the Motion module can be checked from [Online] → [Motion Monitor] → [Event History] in the motion control setting function.
If an error occurs, check the details of the error.
The error occurrence time in the event history of the Motion module synchronizes with those recorded in the servo amplifiers. Check the error details of the event history together with the data in the servo amplifier.



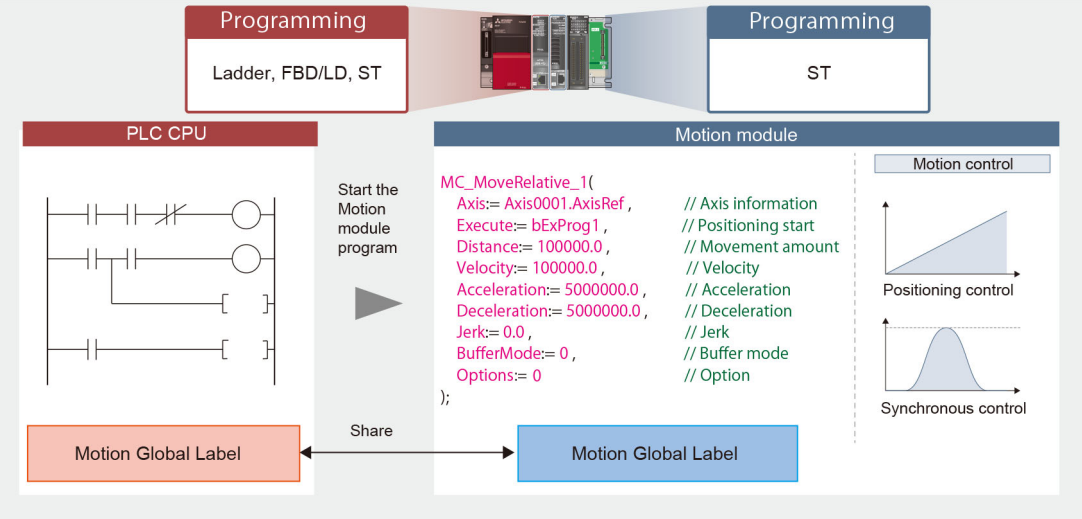
To check the event history of the PLC CPU, click [Diagnosis] → [System monitor] in MELSOFT GX Works3, and click "Event history" on the "System monitor" screen.

MEMO

5. PROGRAMMING BY A PLC CPU AND THE MOTION MODULE

This chapter explains programming by a PLC CPU and the Motion module.
To obtain the sample program used in this chapter, contact your local sales office.

■ File name: **R04-78G4_ STsample_*****.gx3** (***** indicates the version)



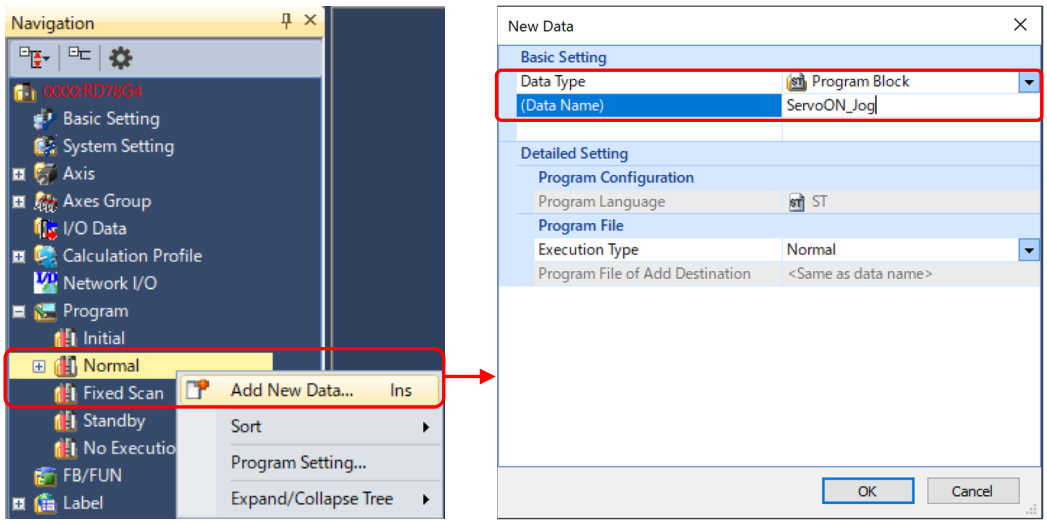
When applying the sample program to an actual system, ensure the applicability and confirm that it will not cause system control problems.
Consider and add interlock conditions according to the user's system.

5.1 Programming Procedure for Motion Module

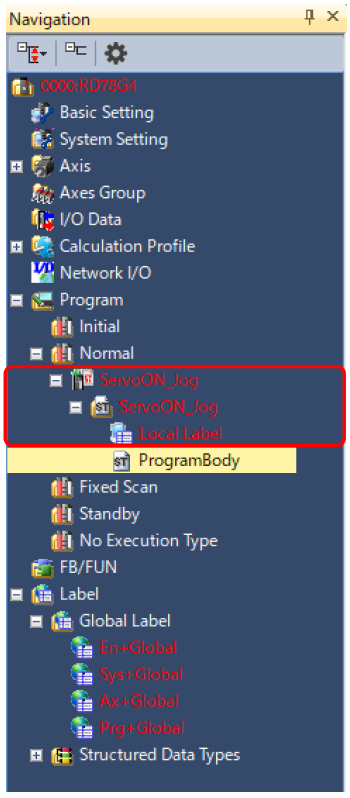
The motion control setting function is used to create a program for the Motion module.

5.1.1 Creating a program block

1) In the navigation window of the motion control setting function, right-click an execution type ("Normal" in this example) under "Program", and click "Add New Data".
2) On the "New Data" screen, set the data name. Only alphanumeric characters can be used to for the data name.



3) A program block is added to the navigation window.



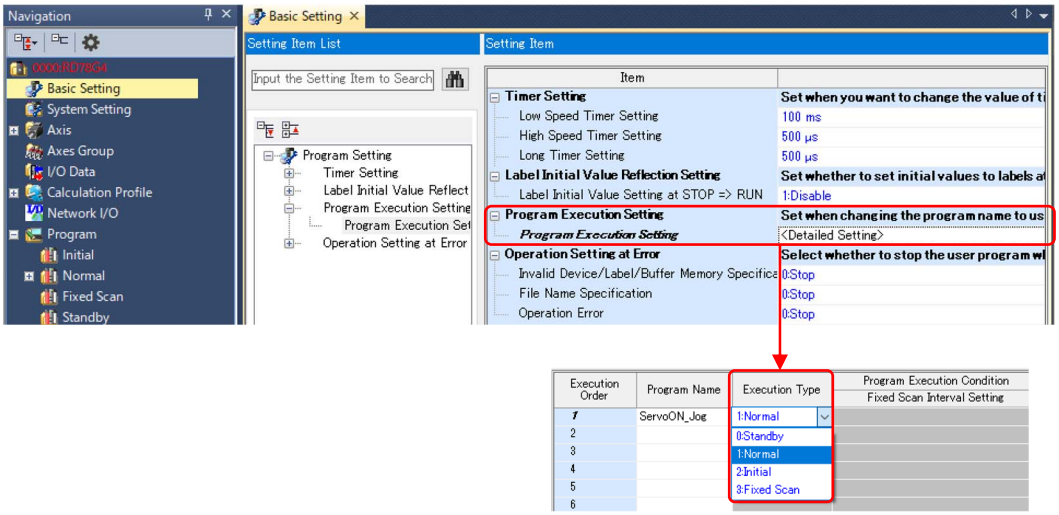
5.1.2 Program execution type

There are following four execution types.

Execution type	Description
0: Standby	This type of program is executed only when its execution is requested. The program is executed after converted to a normal execution type program by using a PSCAN instruction from another running program.
1: Normal	This type of program is executed by normal task of the motion system.
2: Initial	This type of program is executed only once when PLC READY [Y0] changes from OFF to ON.
3: Fixed scan	This type of program is executed at each specified time. (The first operation cycle: 62.5 [us](Note-1) to 6000 [ms]) Note that the time shorter than the operation cycle cannot be set.

(Note-1): "us" indicates microsecond.

To set the execution type, drag and drop the program block onto the target execution type, or change the type from "Basic Setting" → "Program Execution Setting".



5.1.3 Inputting a FB

This section explains the procedure for setting the following JOG operation FB (MCv_Jog).
The sample program adds some interlock conditions and public label settings to this program.

```
1  //Axis0001 JOG
2  MCv_Jog_1(
3      Axis         := Axis0001.AxisRef ,
4      JogForward   := G_bJogF1CMD ,
5      JogBackward  := G_bJogR1CMD ,
6      Velocity     := G_leJogVelocity ,
7      Acceleration:= leJogAcceleration ,
8      Deceleration:= leJogDeceleration ,
9      Jerk         := leJogJerk ,
10     Busy         => G_bJog1Busy
11 );
12
```

(1) Preparing labels

Prepare labels for input/output signals of the FB. This example uses labels for the Velocity, Acceleration, Deceleration, and Jerk inputs of MCv_Jog. When registering a new label, consider whether the label is registered as a global or a local label.

In this example, the labels for JOG command, JOG signal during operation, and JOG velocity are registered as a global label, considering the JOG velocity is operated from an external device, such as GOT (HMI). The labels for JOG acceleration, JOG deceleration, and JOG jerk are as local labels of ServoON_Jog (refer to Section 5.1.1) since these labels are used only within the program. Refer to Section 5.2 for naming rules of labels.

[Global label]

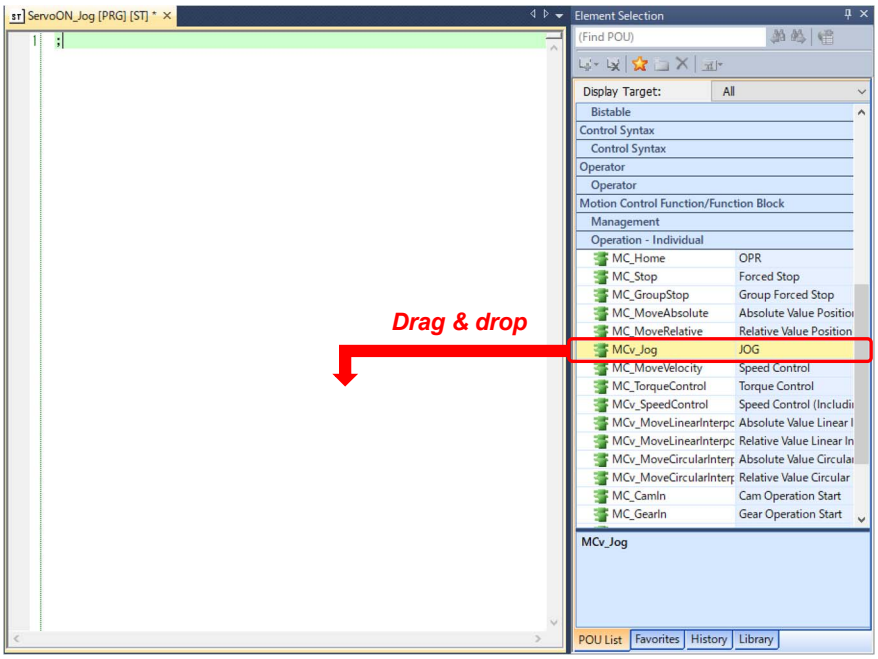
	Label Name	Data Type	Class	Ini	Co	Comment	Re	Public Label	Motion
1	G_bJogF1CMD	Bit	VAR_GLOBAL			JOG正転指令 Axis0001 / JOG Positive rotation command Axis0001		Disable	–
2	G_bJogR1CMD	Bit	VAR_GLOBAL			JOG逆転指令 Axis0001 / JOG Reverse rotation command Axis0001		Disable	–
3	G_bJog1Busy	Bit	VAR_GLOBAL			JOG運転中 Axis0001 / JOG operation in progress Axis0001		Disable	–
4	G_leJogVelocity	FLOAT [Double…	VAR_GLOBAL			JOG速度 / JOG Velocity		Disable	–
5									

[Local label]

	Label Name	Data Type	Class	Initial	Constant	Comment
1	leJogAcceleration	FLOAT [Double Precision]	VAR			JOG加速度 / JOG acceleration
2	leJogDeceleration	FLOAT [Double Precision]	VAR			JOG減速度 / JOG deceleration
3	leJogJerk	FLOAT [Double Precision]	VAR			JOGジャーク / JOG jerk
4						

(2) Inputting a FB

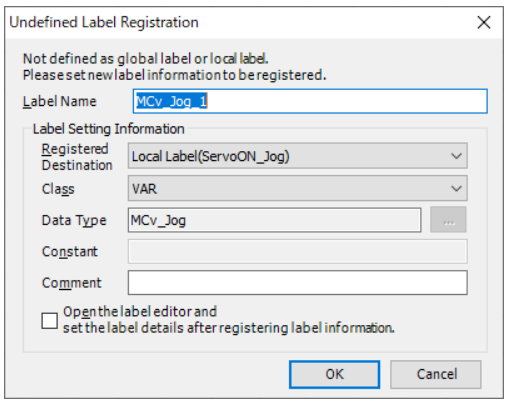
Drag and drop "MCv_Jog" under "Operation-Individual" tree on the [POU list] tab of the element selection window.



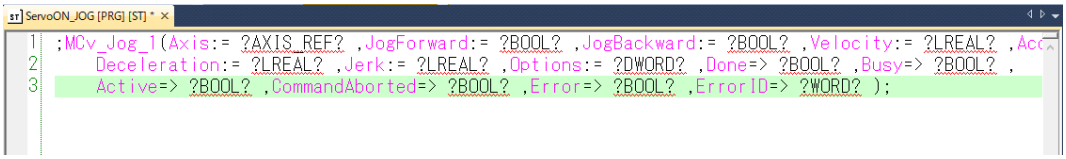
Motion module FBs are categorized in the following three groups.

Group	Description
Management (administrative FB)	A motion control FB that takes an axis or an axes group for the argument and does not change the axis status or the axes group status by execution
Operation-Individual (motion FB)	A motion control FB that takes an axis or an axes group for the argument and changes the axis status or the axes status by execution
StandardFB (general FB)	A motion control FB that does not take an axis or an axes group for the argument

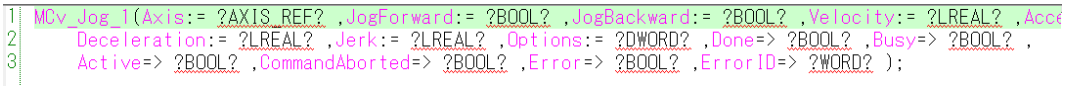
When an undefined label is entered, the "Undefined Label Registration" screen appears. Enter the label name, the registered destination, and, if necessary, the comment. This example uses the initial settings.



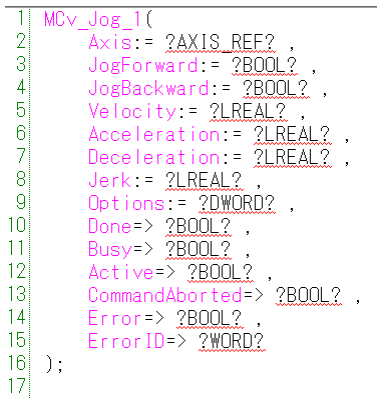
When the setting is completed, click the [OK] button.
The selected FB (MCv_Jog) is displayed on the program editor.



Delete the semicolon at the head of the FB.
(Depending on where a FB is added, there may be two semicolons at the end of the FB. In that case, delete one.)



Users can break a line or adjust indents of a FB.
The following program example breaks the line at every I/O signal and adds indents as follows.



(3) Inputting I/O signals

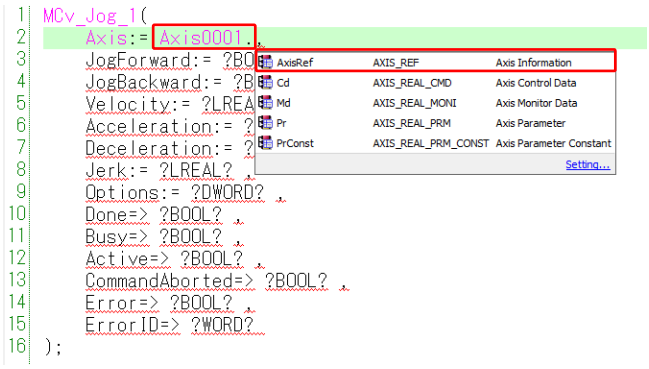
Enter labels for the undefined variables which display ?AXIS_REF?, ?BOOL?, etc.

(a) AxisRef type structure

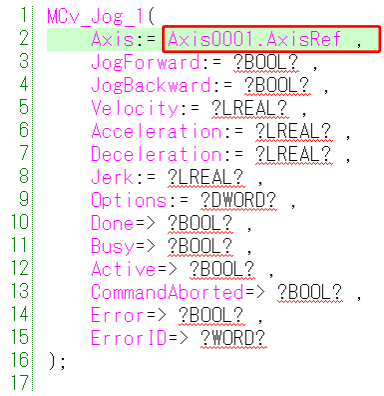
The following describes how to input AxisRef type structure (axis information) of the axis 1 in the input variable of "Axis".

1) When entering "ax" instead of ?AXIS_REF?, the registered labels are displayed as a suggestion.

2) Select Axis0001 and enter ".". Then, the suggestions of the structure member for Axis0001 are displayed.

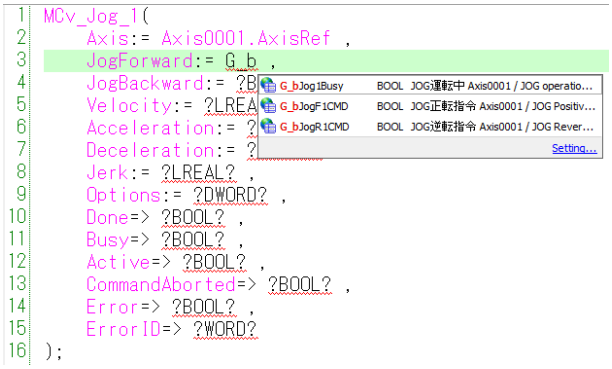


3) Select "AxisRef".



(b) Global and local labels

1) Enter G_bJogF1CMD (global label) for the JogForward input.
When entering "G_b" instead of ?BOOL?, the registered labels are displayed as a suggestion.



2) Select "G_bJogF1CMD".

```
MCv_Jog_1(
    Axis:= Axis0001.AxisRef ,
    JogForward:= G_bJogF1CMD ,
    JogBackward:= ?BOOL? ,
    Velocity:= ?LREAL? ,
    Acceleration:= ?LREAL? ,
    Deceleration:= ?LREAL? ,
    Jerk:= ?LREAL? ,
    Options:= ?DWORD? ,
    Done=> ?BOOL? ,
    Busy=> ?BOOL? ,
    Active=> ?BOOL? ,
    CommandAborted=> ?BOOL? ,
    Error=> ?BOOL? ,
    ErrorID=> ?WORD?
);

```

3) Enter the labels registered in (1) to the corresponding input signals (JogBackward, Velocity, Acceleration, Deceleration, Jerk, and Busy) in the same manner.

```
MCv_Jog_1(
    Axis:= Axis0001.AxisRef ,
    JogForward:= G_bJogF1CMD ,
    JogBackward:= G_bJogR1CMD ,
    Velocity:= G_leJogVelocity ,
    Acceleration:= leJogAcceleration ,
    Deceleration:= leJogDeceleration ,
    Jerk:= leJogJerk ,
    Options:= ?DWORD? ,
    Done=> ?BOOL? ,
    Busy=> G_bJog1Busy ,
    Active=> ?BOOL? ,
    CommandAborted=> ?BOOL? ,
    Error=> ?BOOL? ,
    ErrorID=> ?WORD?
);

```

(c) Omitting I/O signals

I/O signals of a FB can be omitted when the initial value is used, or the signal is not used. In this example, the initial value (0) is used for the Options input, and the Done, Active, CommandAborted, Error, and ErrorID outputs are not used. Therefore, these signals can be omitted.

Delete the unnecessary "," at the end of the FB.

```
MCv_Jog_1(
    Axis:= Axis0001.AxisRef ,
    JogForward:= G_bJogF1CMD ,
    JogBackward:= G_bJogR1CMD ,
    Velocity:= G_leJogVelocity ,
    Acceleration:= leJogAcceleration ,
    Deceleration:= leJogDeceleration ,
    Jerk:= leJogJerk ,
    Busy=> G_bJog1Busy ,
);

```

Delete "," at the end of the FB.

(d) Adding comments and editing for appearance

Add comments or indents as necessary.

```
1  //Axis0001 JOG
2  MCv_Jog_1(
3      Axis         := Axis0001.AxisRef ,
4      JogForward   := G_bJog1CMD ,
5      JogBackward  := G_bJog1CMD ,
6      Velocity     := G_leJogVelocity ,
7      Acceleration:= leJogAcceleration ,
8      Deceleration:= leJogDeceleration ,
9      Jerk         := leJogJerk ,
10     Busy         => G_bJog1Busy
11 );
12
```

The program creation is completed.

5.1.4 ENUM enumerator

The constant values of the axis status (Axis0001.Md.AxisStatus), the buffer mode (BufferMode), etc. are defined as an ENUM enumerator. Programming using the ENUM enumerator is possible.

(1) Inputting ENUM enumerator

[Example of ENUM enumerators for the FB argument]

The enumerator for BufferMode is defined by "MC_BUFFER_MODE__mc*****" (label).

(For details of the BufferMode, refer to Section 5.9.)

```
17
18 //相対値位置決め, Relative value positioning
19 MC_MoveRelative_1(
20     Axis            := Axis0001.AxisRef ,
21     Execute         := bExecute_P ,
22     Distance        := leDistance ,
23     Velocity        := leVelocity ,
24     Acceleration    := leAcceleration ,
25     Deceleration    := leDeceleration ,
26     Jerk            := leJerk ,
27     BufferMode      := mc_buf,
28     Done            => bDone
29     Busy            => bBusy
30     CommandAborted  => bComma
31     Error           => bError
32 );
33
34
```

MC_BUFFER_MODE__mcAborting INT Aborting
MC_BUFFER_MODE__mcBlendingHigh INT BlendingHigh
MC_BUFFER_MODE__mcBlendingLow INT BlendingLow
MC_BUFFER_MODE__mcBlendingNext INT BlendingNext
MC_BUFFER_MODE__mcBlendingPrev... INT BlendingPrevious
MC_BUFFER_MODE__mcBuffered INT Buffered
Setting...

[Example of ENUM enumerator for axis variable]

The enumerator for axis status is defined by "MC_AXIS_STATUS__*****" (label).

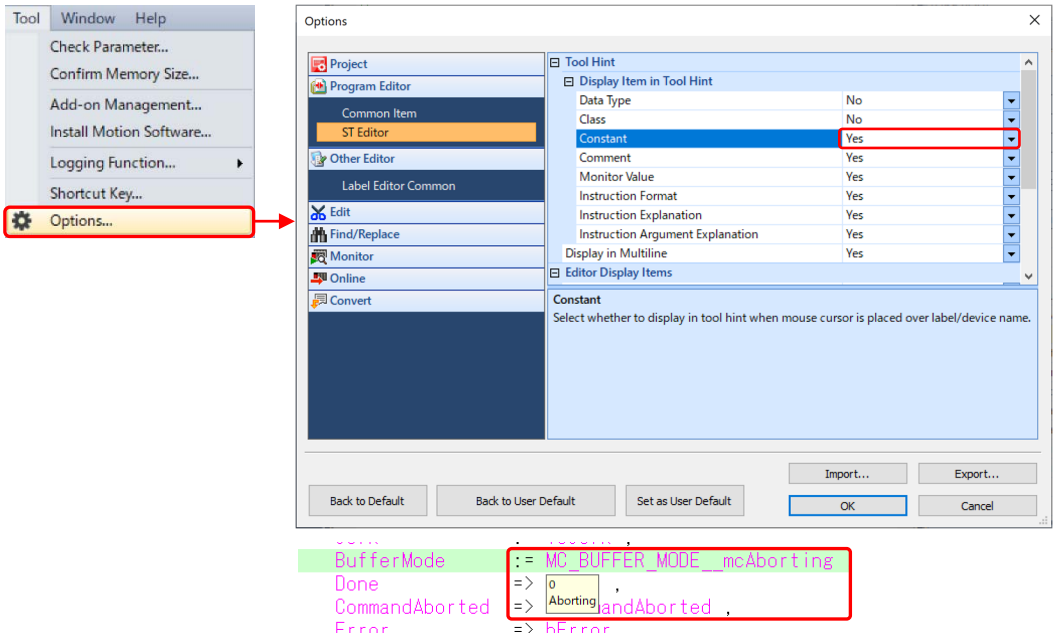
```
66
67 //入力軸始動, Input axis start
68 SET(G_bSyncCMD & (Axis0002.Md.AxisStatus = MC_AXIS_STATUS__SynchronizedMotion),bSyncMoveCMD);
69
```

(2) Checking values of ENUM enumerators

1) Select [Tool] → [Option] to display the option screen.

2) Set "Constant" of the ST editor to "Yes", and click the [OK] button.

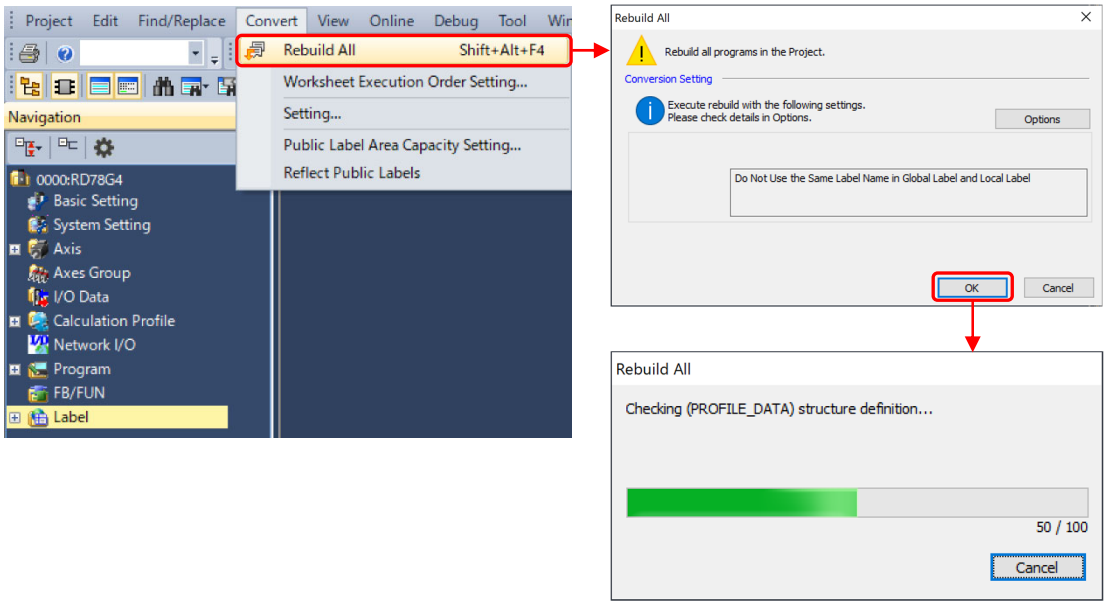
3) Check the value of an ENUM enumerator by moving the cursor on it in the program.



5.1.5 Conversion of all the programs and reflection of public labels

(1) Conversion of all the programs

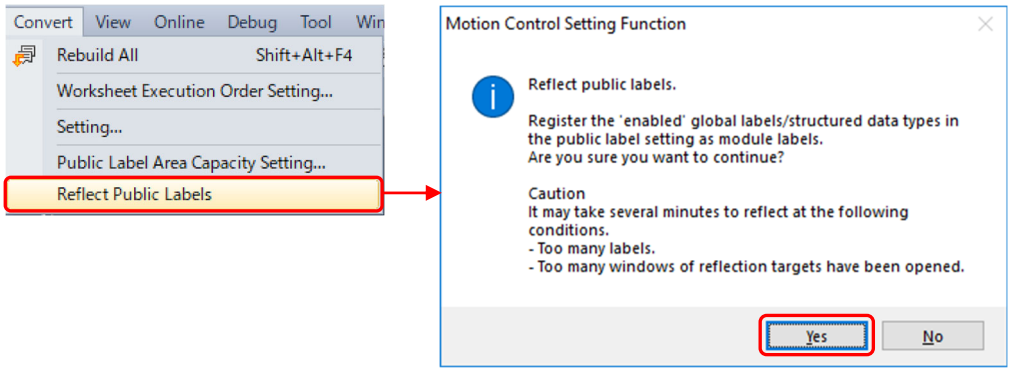
Select [Convert] → [Rebuilt All] in the motion control setting function, and then a message window appears. Click the [OK] button to convert all the programs.



(2) Reflection of public labels

When some labels are registered as a public label, reflect the public labels.

Select [Convert] → [Reflect Public Labels], and then a message window appears. Click the [OK] button to reflect the public labels to the PLC CPU. (Refer to Section 3.14.2.)



Edit the PLC CPU program after reflecting the public labels.

5.2 Naming Rules of Labels

This document uses the following prefix rules for labels to distinguish their data type.

Data type		Value range	Prefix	
			Local	Global
Bit	BOOL	FALSE(0), TRUE(1)	b	G_b
Word [unsigned] /bit string [16 bits]	WORD	0 to 65535	u	G_u
Double word [unsigned]/ bit string [32 bits]	DWORD	0 to 4294967295	ud	G_ud
Word [signed]	INT	-32468 to 32767	w	G_w
Double word [signed]	DINT	-2147483648 to 2147483647	d	G_d
Single-precision real number	REAL	-2^{128} to -2^{-126}, 0, 2^{-126} to 2^{128}	e	G_e
Double-precision real number	LREAL	-2^{1024} to -2^{-1022}, 0, 2^{-1022} to 2^{1024}	le	G_le
Time	TIME	T#-24d20h31m23s648ms to T#24d20h31m23s647ms	tm	G_tm
Timer	TIMER	TIMER structure S: contact C: coil N: current value	td	G_td

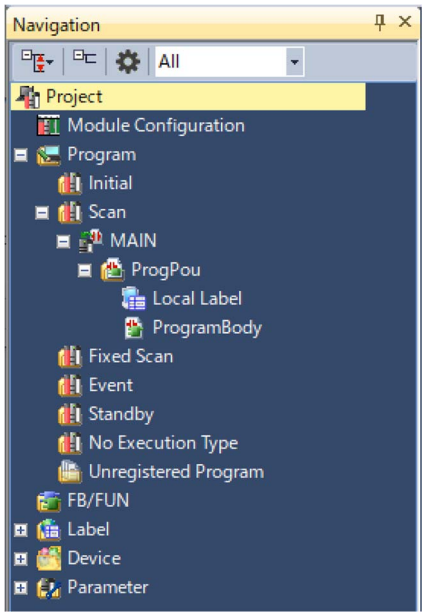
[Examples of local labels]
Bit: bMoveCMD
Double-precision real number: lePosition
Arrays: wAxes[16]
Timer: tdTimer1

[Examples of global labels]
Bit: G_bJogF1
Double-precision real number: G_leVelocity

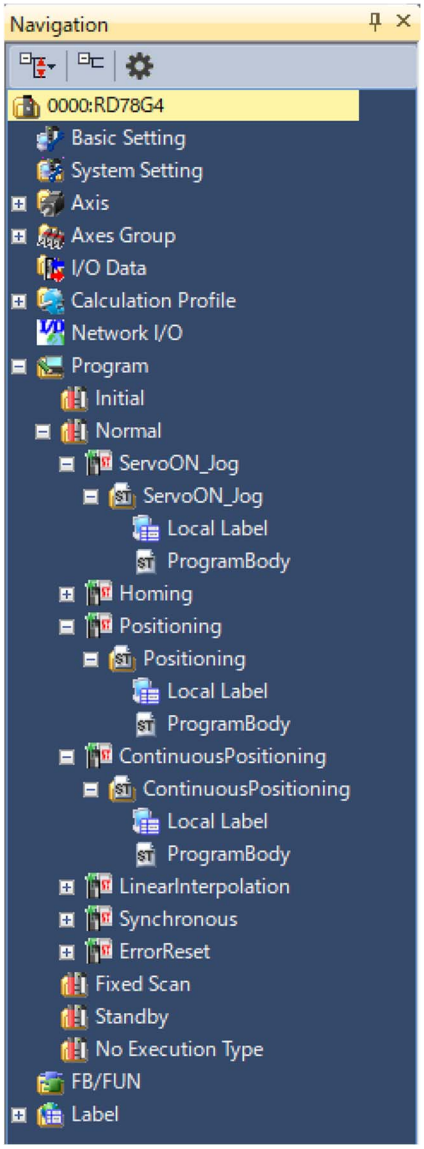
5.3 Projects

5.3.1 Program names

The following shows the program examples explained in this chapter.

[PLC CPU]  	Program name: MAIN A device in the PLC CPU turns ON/OFF the start signal of the Motion module The program is created in ladder because what the program mainly processes is contacts.

[Motion module]



Program name: ServoON_Jog
(PLC READY, all axes servo ON, JOG operation)

Program name: Homing

Program name: Positioning
(single axis positioning)

Program name: ContinuousPositioning
(single axis continuous positioning)

Program name: LinearInterpolation
(two-axis linear interpolation control)

Program name: Synchronous
(synchronous control)

Program name: ErrorReset
(error reset)

5.3.2 Settings for global and public labels

The following shows the setting examples of the global labels in the sample program.
For the local labels, refer to each program.

(1) PLC CPU

The PLC CPU uses the public labels of the Motion module.

(2) Motion module

The following shows the global and public label settings of the Motion module.

[Global]

	Label Name	Data Type	Class	Initial	Const	Comment	Rem	Public Label	Motion Control
1	G_bSRVONCMD	Bit	VAR_GLOBAL			サーボオンオフ / Servo ON/OFF		Enable	WRITE (=> Motion)
2	G_bHoming1CMD	Bit	VAR_GLOBAL			原点復帰指令 Axis0001 / Homing command Axis0001		Enable	WRITE (=> Motion)
3	G_bHoming2CMD	Bit	VAR_GLOBAL			原点復帰指令 Axis0002 / Homing command Axis0002		Enable	WRITE (=> Motion)
4	G_bHoming3CMD	Bit	VAR_GLOBAL			原点復帰指令 VirtualAxis0001 / Homing command VirtualAxis001		Enable	WRITE (=> Motion)
5	G_bPosCMD	Bit	VAR_GLOBAL			単軸位置決め始動 / Single axis positioning start		Enable	WRITE (=> Motion)
6	G_bContPosCMD	Bit	VAR_GLOBAL			単軸連続位置決め始動 / Single axis continuous positioning start		Enable	WRITE (=> Motion)
7	G_bInterpolationCMD	Bit	VAR_GLOBAL			2軸直線補間制御始動 / 2-axis linear interpolation control start		Enable	WRITE (=> Motion)
8	G_bSyncCMD	Bit	VAR_GLOBAL			同期制御始動 / Synchronous control start		Enable	WRITE (=> Motion)
9	G_bResetCMD	Bit	VAR_GLOBAL			エラーリセット / Error reset		Enable	WRITE (=> Motion)
10	G_bMotionResetCMD	Bit	VAR_GLOBAL			システムエラーリセット / System error reset		Enable	WRITE (=> Motion)
11	G_bJogF1CMD	Bit	VAR_GLOBAL			JOG正転指令 Axis0001 / JOG Positive rotation command Axis0001		Enable	WRITE (=> Motion)
12	G_bJogR1CMD	Bit	VAR_GLOBAL			JOG逆転指令 Axis0001 / JOG Reverse rotation command Axis0001		Enable	WRITE (=> Motion)
13	G_bJogF2CMD	Bit	VAR_GLOBAL			JOG正転指令 Axis0002 / JOG Positive rotation command Axis0002		Enable	WRITE (=> Motion)
14	G_bJogR2CMD	Bit	VAR_GLOBAL			JOG逆転指令 Axis0002 / JOG Reverse rotation command Axis0002		Enable	WRITE (=> Motion)
15	G_leJogVelocity	FLOAT [Doubl…	VAR_GLOBAL			JOG速度 / JOG Velocity		Enable	WRITE (=> Motion)
16	G_bHoming1Done	Bit	VAR_GLOBAL			原点復帰完了 Axis0001 / Homing complete Axis0001		Enable	READ (Motion =>)
17	G_bHoming2Done	Bit	VAR_GLOBAL			原点復帰完了 Axis0002 / Homing complete Axis0002		Enable	READ (Motion =>)
18	G_bHoming3Done	Bit	VAR_GLOBAL			原点復帰完了 VirtualAxis0001 / Homing complete VirtualAxis0001		Enable	READ (Motion =>)
19	G_bPosDone	Bit	VAR_GLOBAL			単軸位置決め完了 / Single axis positioning complete		Enable	READ (Motion =>)
20	G_bContPosDone	Bit	VAR_GLOBAL			単軸連続位置決め完了 / Single axis continuous positioning complete		Enable	READ (Motion =>)
21	G_bInterpolationDone	Bit	VAR_GLOBAL			2軸直線補間制御完了 / 2-axis linear interpolation control complete		Enable	READ (Motion =>)
22	G_bSyncDone	Bit	VAR_GLOBAL			同期制御完了 / Synchronous control complete		Enable	READ (Motion =>)
23	G_bJog1Busy	Bit	VAR_GLOBAL			JOG運転中 Axis0001 / JOG operation in progress Axis0001		Enable	READ (Motion =>)
24	G_bJog2Busy	Bit	VAR_GLOBAL			JOG運転中 Axis0002 / JOG operation in progress Axis0002		Enable	READ (Motion =>)
25	G_bStopSignalX	Bit	VAR_GLOBAL			停止指令 Axis0001 / Stop command Axis0001		Enable	READ (Motion =>)
26	G_bStopSignalY	Bit	VAR_GLOBAL			停止指令 Axis0002 / Stop command Axis0002		Enable	READ (Motion =>)
27									

1) These labels are used to execute operation of each program.

2) This double-precision real number label stores the JOG velocity.
By registering this label as a global label, the JOG velocity can be changed from a GOT (HMI) or other programs.

3) These labels transmit the completion signal of each program to the PLC CPU.

4) These labels turn ON while the JOG operation is being executed.
They are used as an interlock condition to prevent execution of the JOG operation while other programs are running.

5) These labels stop the axis operation.
The operation is aborted when these labels are turned ON during the operation.
(Refer to Section 5.13.4 for details.)

[Sys+Global]

The following is the structure that stores data related to the system. Set "Public label" to "Enable".

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Remark	Public Label	Motion Control
1	System	SYSTEM	VAR_GLOBAL	<Detailed Setting>				Enable	–
2									

[AX+Global]

These labels are registered automatically when the axis is set in the motion control setting function. No operation is required on the label editor.

	Label Name	Data Type	Class	Initial Value	Constan	Comment	Remark	Public Label	Motion Control
1	Axis0001	AXIS_REAL	VAR_GLOBAL	<Detailed Setting>		[X_Axis]		Disable	–
2	Axis0002	AXIS_REAL	VAR_GLOBAL	<Detailed Setting>		[Y_Axis]		Disable	–
3	VirtualAxis0001	AXIS_VIRTUAL	VAR_GLOBAL	<Detailed Setting>		[Vir_Axis01]		Disable	–
4	LinkAxis0001	AXIS_VIRTUAL_LINK	VAR_GLOBAL	<Detailed Setting>		[Lin_Axis01]		Disable	–
5	LinkAxis0002	AXIS_VIRTUAL_LINK	VAR_GLOBAL	<Detailed Setting>		[Lin_Axis02]		Disable	–
6									

[Gr+Global]

This label is registered automatically when the axes group is set in the motion control setting function. No operation is required on the label editor.

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Remark	Public Label	Motion Control
1	AxesGroup001	AXES_GROUP	VAR_GLOBAL	<Detailed…		[X-Y Table]		Disable	–
2									

[Prf+Global]

These labels are registered automatically when the profile data is registered in the motion control setting function. No operation is required on the label editor.

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Remark	Public Label	Motion Control
1	ProfileData0001	MC_CAM_REF	VAR_GLOBAL	<Detailed…		[Cam #1]		Disable	–
2									

[Prg+Global]

These labels are registered automatically when the program is created in the motion control setting function. No operation is required on the label editor.

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Remark	Public Label	Motion Control
1	ServoON_Jog	PROGRAM_INFO	VAR_GLOBAL	<Detailed…				Disable	–
2	Homing	PROGRAM_INFO	VAR_GLOBAL	<Detailed…				Disable	–
3	Positioning	PROGRAM_INFO	VAR_GLOBAL	<Detailed…				Disable	–
4	ContinuousPosi…	PROGRAM_INFO	VAR_GLOBAL	<Detailed…				Disable	–
5	LinearInterpolat…	PROGRAM_INFO	VAR_GLOBAL	<Detailed…				Disable	–
6	Synchronous	PROGRAM_INFO	VAR_GLOBAL	<Detailed…				Disable	–
7	ErrorReset	PROGRAM_INFO	VAR_GLOBAL	<Detailed…				Disable	–
8									

[En+Global]

This group is for defining ENUM enumerators, which is automatically registered by the system. No operation is required on the user side.

[SYS_MONI structure]

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Public Label	Motion Control
1	Addon_AbsSystem	ADDON_MONI				Add-on AbsSystem Monitor	Disable	–
2	Addon_Axis	ADDON_MONI				Add-on Axis Monitor	Disable	–
3	Addon_ExternalSignal	ADDON_MONI				Add-on ExternalSignal Monitor	Disable	–
4	Addon_FileTransfer	ADDON_MONI				Add-on FileTransfer Monitor	Disable	–

≈ ≈

23	BuffermemoryRefreshCycle	CYCLE_MONI				Buffer Memory Refresh Cycle Moni…	Disable	–
24	Environment_UserRootPath	String(127)		''		User Root Path	Disable	–
25	Error	Bit		0		Motion System Error Detection	Enable	–
26	ErrorHistory	Word [Unsigned]/Bit…		0		Error History Information	Disable	–
27	ErrorHistory_Latest	Word [Signed]		0		Latest Error History Data No.	Disable	–

[SYSTEM structure]

	Label Name	Data Type	Class	Initial Value	Constant	Comment	Public Label	Motion Control Attribute
1	PrConst	SYS_PRM_CONST				System Parameter Constant	Disable	WRITE (=> Motion): PrConst
2	Pr	SYS_PRM				System Parameter	Disable	WRITE (=> Motion): Pr
3	Md	SYS_MONI				System Monitor Data	Enable	READ (Motion =>): Md
4	Cd	SYS_CMD				System Control Data	Disable	WRITE (=> Motion): Cd
5	LOGGING_REALTIME	LOGGING_REALTIME				Logging Realtime Monitor	Disable	–
6	LoggingRef	LOGGING_REF(1..10)				Logging	Disable	–
7								

[POINTS]

When the public label settings of the Motion module are changed, reflect the public labels.
(Refer to Section 3.14.2.)

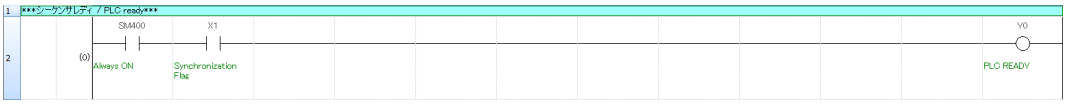
5.4 PLC READY (Program Name: ServoON_Jog)

The Motion module uses 32 points each for inputs and outputs to send/receive data to/from a PLC CPU.
Regardless of the programming language, PLC READY [Y0] needs to be turned ON by a PLC CPU to start operation of the Motion module.

(1) Program example

PLC READY flag [Y0] is turned ON when the PLC CPU is powered ON and the synchronization flag [X1] for the Motion module is turned ON.

[PLC CPU]



[Motion module]

None

5.5 Servo ON (Program Name: ServoON_Jog)

This program turns ON the real drive axes connected to the servo system.
The following two FBs are used for executing servo ON.

(1) FBs

Type	Command	Description
MCFB (administrative)	MC_Power	Switches a specified axis to the operation possible status.
	MCv_AllPower	Switches all axes to the operation possible status.

(2) Local labels

	Label Name	Data Type	Class	Initial Value	Constant	Comment
1	MCv_AllPower_1	MCv_AllPower	VAR			全軸サーボオンFB / All axes servo ON FB
2	MCv_Jog_1	MCv_Jog	VAR			JOG運転FB1 Axis0001 / JOG operation FB1 Axis0001
3	MCv_Jog_2	MCv_Jog	VAR			JOG運転FB2 Axis0002 / JOG operation FB2 Axis0002
4	leJogAcceleration	FLOAT [Double Precision]	VAR			JOG加速度 / JOG acceleration
5	leJogDeceleration	FLOAT [Double Precision]	VAR			JOG減速度 / JOG deceleration
6	leJogJerk	FLOAT [Double Precision]	VAR			JOGジャーク / JOG jerk
7						

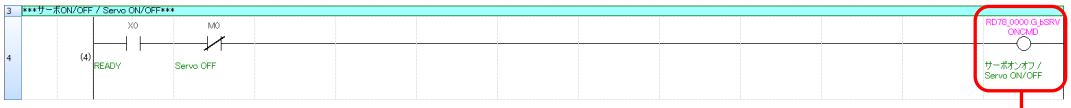
The labels for JOG operation

Drag and drop MCv_AllPower to the program editor. The FB (MCv_AllPower_1) is added as a local label. (Refer to Section 5.1.3.)

(3) Program example

When PLC READY [Y0] is turned ON, Ready [X0] is turned ON.
Turning ON of X0 is used as the condition for executing all axes servo ON.
When executing servo OFF, turn ON [M0].

[PLC CPU]



[Motion module]

```
1  //----全軸サーボON----------
2  //----All axes servo ON-----
3
4  MCv_AllPower_1(
5      Enable   := TRUE ,
6      ServoON  := G_bSRVONCMD
7  );
```

5.6 JOG Operation (Program Name: ServoON_Jog)

The servo system outputs commands to a target axis and operates the axis in a specified direction while the JOG positive/reverse rotation commands are ON.

(1) FB

Type	Command	Description
MCFB (motion)	MCv_Jog	Executes a JOG operation according to the target velocity.

(2) Acceleration/deceleration processing function

The acceleration/deceleration function is used to adjust the acceleration/deceleration curve of the motion control according to the user's machine.

(a) Overview

Select the acceleration/deceleration methods from the following.

Method	Description
Acceleration/deceleration specification method (Initial)	Accelerates/decelerates the axis by the acceleration, deceleration, and jerk values specified in the FB.
Acceleration/deceleration time-fixed method	Accelerates/decelerates the axis based on the time specified in the FB regardless of speed.

(b) Setting method

The acceleration/deceleration method can be specified by the Options input of motion FBs, such as MCv_Jog.

Bit	Description
0 to 2	Acceleration/deceleration method setting 0: mcAccDec (acceleration/deceleration specification method) 1: mcFixedTime (acceleration/deceleration time-fixed method)
3 to 15	The function varies depending on the FB.
16 to 31	

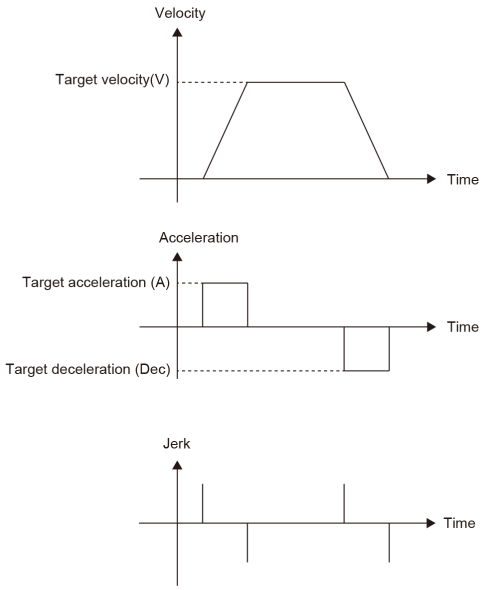
(c) Acceleration/deceleration specification method

Select "0: mcAccDec" in the Options input (using bit 0 to 2 for acceleration/deceleration method setting) of the FB, and set acceleration, deceleration, and jerk.

1) Trapezoidal acceleration/deceleration

When "0.0" is set for the Jerk input of the FB, the operation is referred to as the trapezoidal acceleration/deceleration.

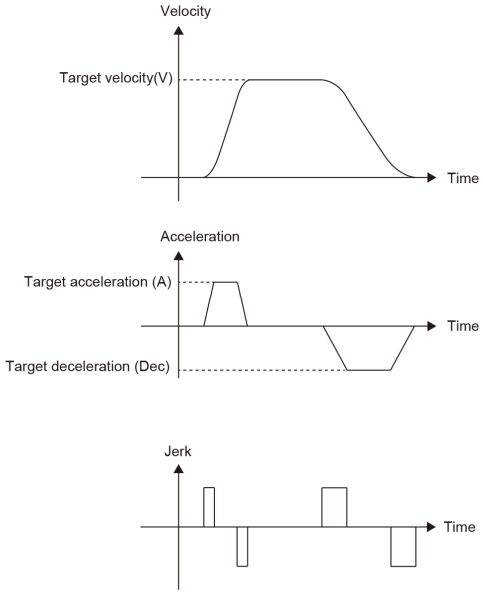
The velocity creates a trapezoidal shape.



2) Jerk acceleration/deceleration

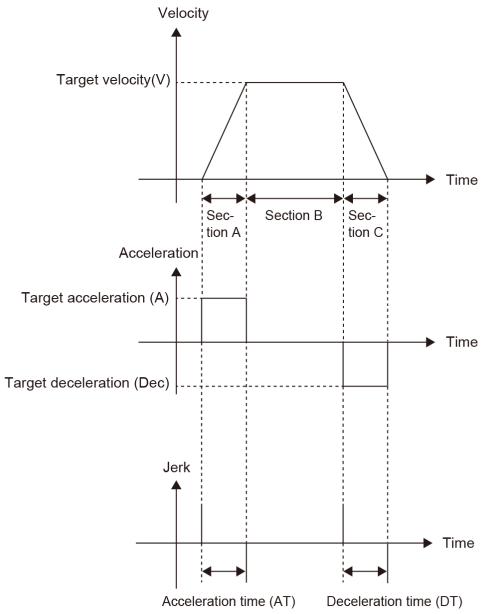
When values other than "0.0" is set for the Jerk input of the FB, the operation is referred to as the jerk acceleration/deceleration.

The velocity creates an S-curve shape.



(d) Acceleration/deceleration time-fixed method

Select "1: mcFixedTime " in the Options input (using bit 0 to 2 for acceleration/deceleration method setting) of the FB, and set the acceleration time in the Acceleration input of the FB. The Deceleration and Jerk inputs are not used.



(e) FB input variables

The following table lists the input variables explained in (b) to (d).

Input variable	Name	Details
Velocity	Target velocity	Specify the target velocity.
Acceleration	Acceleration or acceleration/ deceleration time	Specify the acceleration or the acceleration/deceleration time. • The unit is [U/s^2] when the method is set to "0: mcAccDec". • The unit is [s] when the method is set to "1: mcFixedTime".
Deceleration	Deceleration	Specify the deceleration. • The unit is [U/s^2] when the method is set to "0: mcAccDec". • Not used when the method is set to "1: mcFixedTime"
Jerk	Jerk	Specify the jerk. • The unit is [U/s^3] when the method is set to "0: mcAccDec". • Not used when the method is set to "1: mcFixedTime".
Options	Options	Specify the acceleration/deceleration method using bit 0 to 2. 0: mcAccDec Acceleration/deceleration specification method 1: mcFixedTime Acceleration/deceleration time-fixed method

[POINTS]

When the "Acceleration/deceleration specification method" (Options: 0) is selected, the acceleration/deceleration can be calculated using the acceleration/deceleration time as follows.

[Target velocity: V [U/s], acceleration time [s], deceleration time [s]],

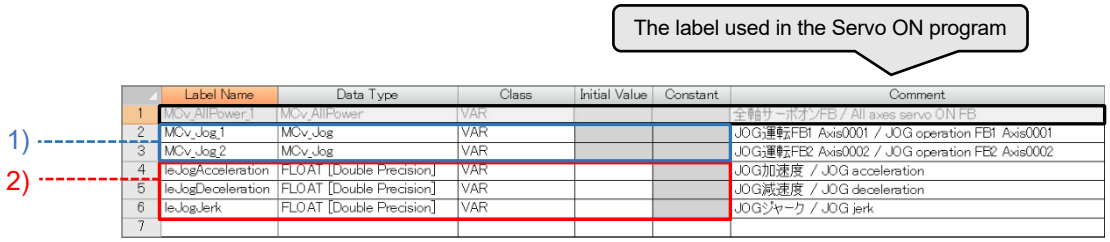
Velocity := (target velocity V);

Acceleration := (target velocity V/acceleration time);

Deceleration := (target velocity V/deceleration time);

Options := (mcAccDec);

(3) Local labels



	Label Name	Data Type	Class	Initial Value	Constant	Comment
1	MCv_AllPower_1	MCv_AllPower	VAR			全軸サーボオンFB / All axes servo ON FB
2	MCv_Jog_1	MCv_Jog	VAR			JOG運転FB1 Axis0001 / JOG operation FB1 Axis0001
3	MCv_Jog_2	MCv_Jog	VAR			JOG運転FB2 Axis0002 / JOG operation FB2 Axis0002
4	leJogAcceleration	FLOAT [Double Precision]	VAR			JOG加速度 / JOG acceleration
5	leJogDeceleration	FLOAT [Double Precision]	VAR			JOG減速度 / JOG deceleration
6	leJogJerk	FLOAT [Double Precision]	VAR			JOGジャーク / JOG jerk
7						

1) These labels are automatically added when the user drags and drops the FB "MCv_Jog" to the program editor.

2) These labels are registered manually.

(4) Program example

The JOG velocity is set from the PLC CPU program since it is registered as a public label ("Write(=>Motion)" is selected). After the PLC CPU is turned to RUN state, the JOG velocity can be changed from the PLC CPU program or on GOT (HMI).

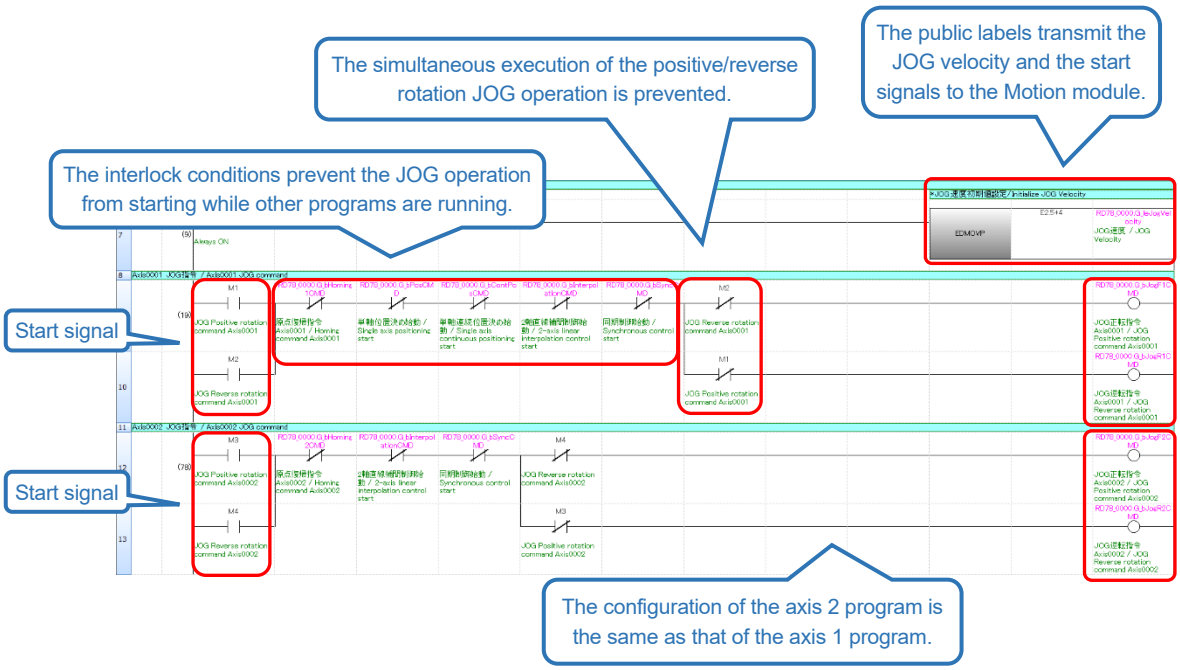
The ON/OFF status of the devices assigned as JOG operation signals (positive/reverse rotation) is transmitted to the Motion module through the public labels.

The interlock conditions are added to prevent execution of the JOG operation while other programs are running.

[Start signal]

	Axis0001	Axis0002
JOG positive rotation command	M1	M3
JOG reverse rotation command	M2	M4

[PLC CPU]



[Motion module]

```
//----JOG運転用データ設定---------------
//----Data setting for JOG operation----

IF MCv_AllPower_1.Busy THEN
    leJogAcceleration := 50000.0;
    leJogDeceleration := 50000.0;
    leJogJerk         := 0.0;
END_IF;

//----Axis0001 JOG運転----------
//----Axis0001 JOG operation----

MCv_Jog_1(
    Axis         := Axis0001.AxisRef ,
    JogForward   := G_bJogF1CMD ,
    JogBackward  := G_bJogR1CMD ,
    Velocity     := G_leJogVelocity ,
    Acceleration := leJogAcceleration ,
    Deceleration := leJogDeceleration ,
    Jerk         := leJogJerk ,
    BUSY         => G_bJog1Busy
);

//----Axis0002 JOG運転----------
//----Axis0002 JOG operation----

MCv_Jog_2(
    Axis         := Axis0002.AxisRef ,
    JogForward   := G_bJogF2CMD ,
    JogBackward  := G_bJogR2CMD ,
    Velocity     := G_leJogVelocity ,
    Acceleration := leJogAcceleration ,
    Deceleration := leJogDeceleration ,
    Jerk         := leJogJerk  ,
    BUSY         => G_bJog2Busy
);
```

When Axis0001 is changed to servo ON status, the values for JOG acceleration, deceleration, and jerk are stored to the specified labels.

These labels are operated from the PLC CPU program.

This signal is transmitted to the PLC CPU, which is used as an interlock condition of other programs.

The configuration of the axis 2 program is the same as that of the axis 1 program.

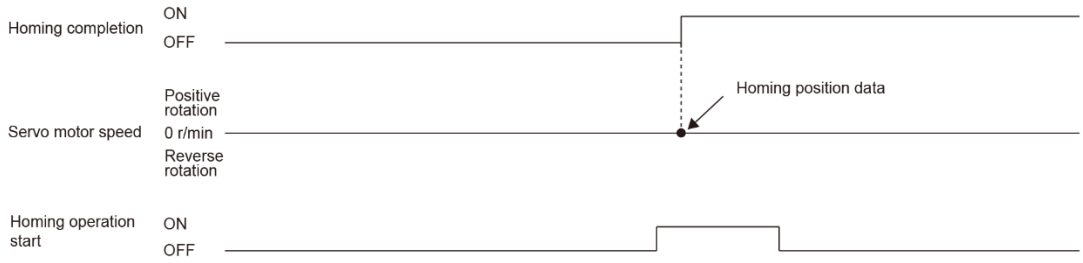
5.7 Homing (Program Name: Homing)

The homing method is set using the parameters of the servo amplifier.
This section explains the data set method homing.
For the dog type homing, refer to Appendix.

(1) Overview

The data set type homing sets the current position as the home position.

[Time chart of data set type homing (Method37)]



(2) FB

Type	Command	Description
MCFB (motion)	MC_Home	Executes homing of the specified axis.

(3) Local labels

1) ----------
2) ----------

	Label Name	Data Type	Class	Initial Value	Constant	Comment
1	MC_Home_1	MC_Home	VAR			原点復帰FB1 Axis0001 / Homing FB1 Axis0001
2	MC_Home_2	MC_Home	VAR			原点復帰FB2 Axis0002 / Homing FB2 Axis0002
3	MC_Home_3	MC_Home	VAR			原点復帰FB3 VirtualAxis0001 / Homing FB3 VirtualAxis0001
4	bHoming1Done	Bit	VAR			原点復帰FB1 Done出力 / Homing FB1 Done output
5	bHoming1Error	Bit	VAR			原点復帰FB1 Error出力 / Homing FB1 Error output
6	bHoming2Done	Bit	VAR			原点復帰FB2 Done出力 / Homing FB2 Done output
7	bHoming2Error	Bit	VAR			原点復帰FB2 Error出力 / Homing FB2 Error output
8	bHoming3Done	Bit	VAR			原点復帰FB3 Done出力 / Homing FB3 Done output
9	bHoming3Error	Bit	VAR			原点復帰FB3 Error出力 / Homing FB3 Error output
10						

1) These labels are automatically added when the user drags and drops the FB (MC_Home) to the program editor.
2) These labels are registered manually.

(4) Program example

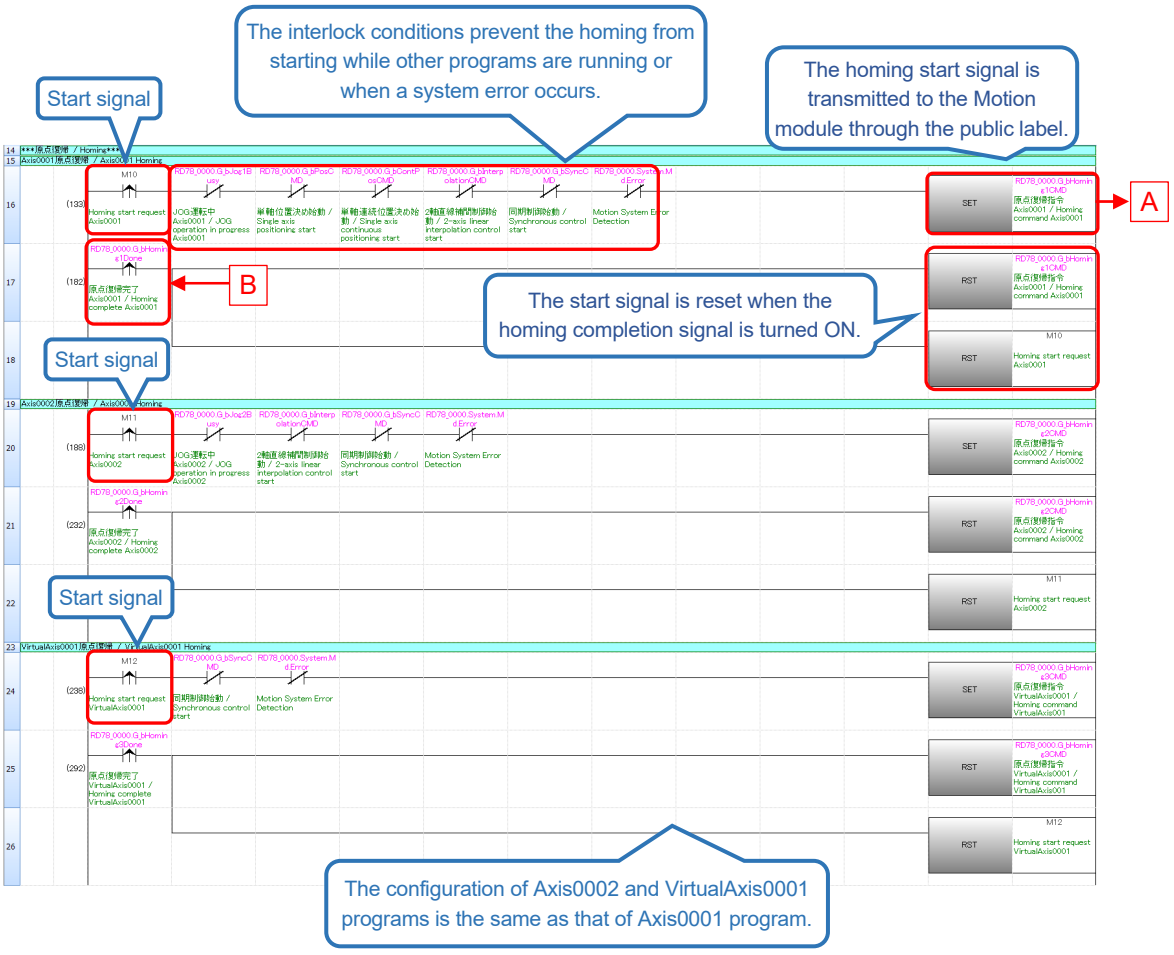
The start signal (the rising edge of the homing command device) is transmitted to the Motion module through the public label. When the homing is completed, the homing completion signal is transmitted to the PLC CPU to turn OFF the start signal.

The interlock conditions are added to prevent execution of the homing while other programs are running or when a system error occurs.

[Start signal]

	Axis0001	Axis0002	VirtualAxis0001
Homing command	M10	M11	M12

[PLC CPU]



[Motion module]

```
1  //----原点復帰 Axis0001----
2  //----Homing Axis0001------
3
4  //原点復帰, Homing
5  MC_Home_1(
6      Axis     := Axis0001.AxisRef ,
7      Execute  := G_bHoming1CMD ,
8      Position := 0.0 ,
9      Done     => bHoming1Done,
10     Error    => bHoming1Error
11 );
12
13 //シーケンサに送る原点復帰完了信号をONする
14 //Transmits homing complete signal to the PLC CPU
15 G_bHoming1Done := bHoming1Done OR bHoming1Error;
16
17 //----原点復帰 Axis0002----
18 //----Homing Axis0002------
19
20 //原点復帰, Homing
21 MC_Home_2(
22     Axis     := Axis0002.AxisRef ,
23     Execute  := G_bHoming2CMD ,
24     Position := 0.0 ,
25     Done     => bHoming2Done,
26     Error    => bHoming2Error
27 );
28
29 //シーケンサに送る原点復帰完了信号をONする
30 //Transmits homing complete signal to the PLC CPU
31 G_bHoming2Done := bHoming2Done OR bHoming2Error;
32
33 //----原点復帰  VirtualAxis0001----
34 //----Homing VirtualAxis0001-------
35
36 //原点復帰, Homing
37 MC_Home_3(
38     Axis     := VirtualAxis0001.AxisRef ,
39     Execute  := G_bHoming3CMD ,
40     Position := 0.0 ,
41     Done     => bHoming3Done,
42     Error    => bHoming3Error
43 );
44
45 //シーケンサに送る原点復帰完了信号をONする
46 //Transmits homing complete signal to the PLC CPU
47 G_bHoming3Done := bHoming3Done OR bHoming3Error;
48
```

A

B

The configuration of Axis0002 and VirtualAxis0001 programs is the same as that of Axis0001 program.

5.8 Single Axis Positioning Control (Program Name: Positioning)

The single axis positioning control executes positioning to a specified position by using the address information.

(1) FBs

Type	Command	Description
MCFB (motion)	MC_MoveAbsolute	Executes positioning using a specified absolute position.
	MC_MoveRelative	Executes positioning using a specified relative distance.

(2) Local labels

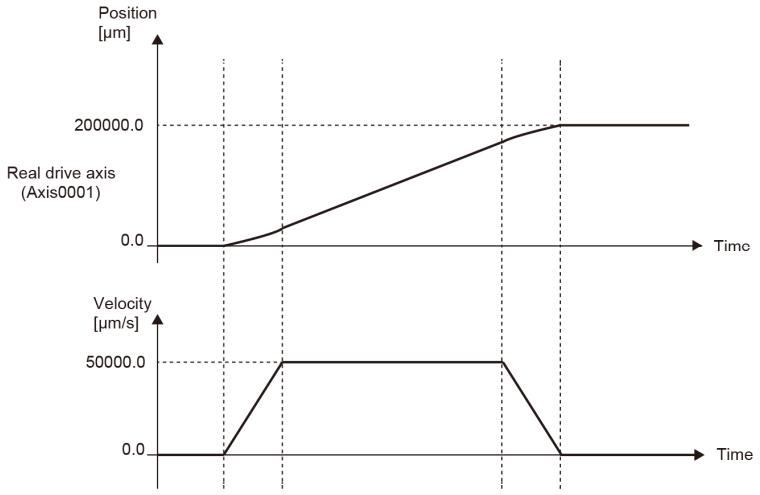
1)
2)

	Label Name	Data Type	Class	Initial Value	Constant	Comment
1	MC_MoveRelative_1	MC_MoveRelative	VAR			相対値位置決めFB / Relative value positioning FB
2	leDistance	FLOAT [Double Precision]	VAR			移動量 / Distance
3	leVelocity	FLOAT [Double Precision]	VAR			速度 / Velocity
4	leAcceleration	FLOAT [Double Precision]	VAR			加速度 / Acceleration
5	leDeceleration	FLOAT [Double Precision]	VAR			減速度 / Deceleration
6	leJerk	FLOAT [Double Precision]	VAR			ジャーク / Jerk
7	bDone	Bit	VAR			相対値位置決めFB Done出力 / Relative value positioning FB Done output
8	bBusy	Bit	VAR			相対値位置決めFB Busy出力 / Relative value positioning FB Busy output
9	bError	Bit	VAR			相対値位置決めFB Error出力 / Relative value positioning FB Error output
10	bExecute_P	Bit	VAR			実行指令 / Start
11	bCommandAborted	Bit	VAR			相対値位置決めFB 実行中断出力 / Relative value positioning FB CommandAborted output
12						

1) This label is automatically added when the user drags and drops the FB (MC_MoveRelative) to the program editor.
2) These labels are registered manually.

(3) Program example

This program executes relative positioning in the following operation pattern.

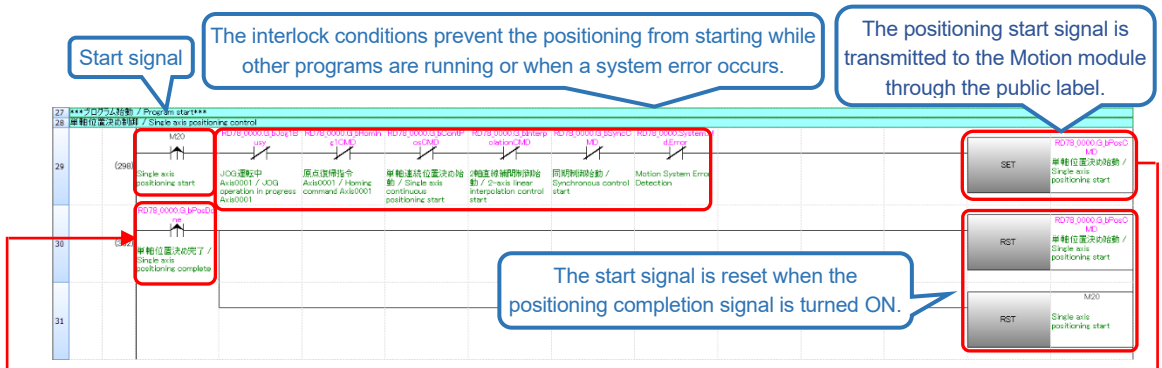


The start signal (the rising edge of the single axis positioning command device) is transmitted to the Motion module through the public label. Responding to the start signal, the Motion module stores each positioning data to the specified local label. When all the data are stored, the positioning FB is executed. When the positioning is completed, the completion signal is transmitted to the PLC CPU to turn OFF the start signal.
The interlock conditions are added to prevent execution of the single axis positioning while other programs are running or when a system error occurs.

[Start signal]

Single axis positioning command	M20

[PLC CPU]



[Motion module]

```
1  //----単軸位置決め制御用データ設定・実行指令&リセット  ----------------------
2  //----Single axis positioning control (data setting, execution, and reset)----
3
4  IF G_bPosCMD THEN;
5      leDistance      := 200000.0;
6      leVelocity      := 50000.0;
7      leAcceleration  := 100000.0;
8      leDeceleration  := 100000.0;
9      leJerk          := 0.0;
10     bExecute_P := TRUE;
11 ELSE
12     bExecute_P := FALSE;
13 END_IF;
14
15 //----単軸位置決め---------------
16 //----Single axis positioning----
17
18 //相対値位置決め, Relative value positioning
19 MC_MoveRelative_1(
20     Axis            := Axis0001.AxisRef ,
21     Execute         := bExecute_P ,
22     Distance        := leDistance ,
23     Velocity        := leVelocity ,
24     Acceleration    := leAcceleration ,
25     Deceleration    := leDeceleration ,
26     Jerk            := leJerk ,
27     Done            => bDone ,
28     Busy            => bBusy ,
29     CommandAborted  => bCommandAborted ,
30     Error           => bError
31 );
32
33 //シーケンサに送る <= 位置決め完了信号・FB実行中断・FBエラーのいずれかの出力がONする ＊制御終了信号
34 //Transmits positioning complete signal to the PLC CPU
35 G_bPosDone := ( bDone & Axis0001.Md.CmdInPos ) OR bCommandAborted OR bError;
36
```

When the start signal is turned ON, each positioning data is stored to the specified label.

When all the data is stored, the positioning FB is executed.

The positioning is completed when the Done output and the command in-position are turned ON.

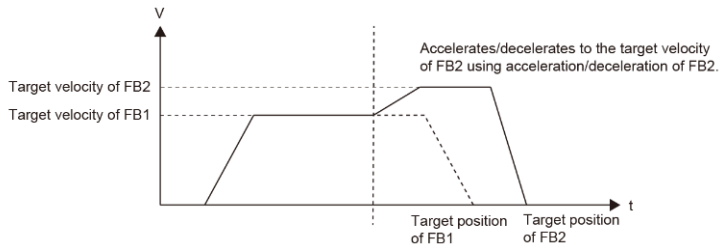
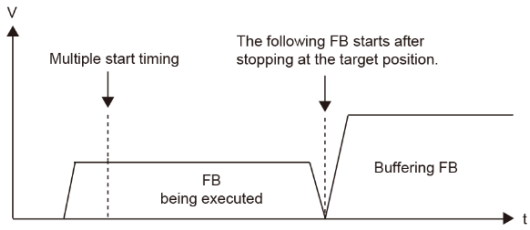
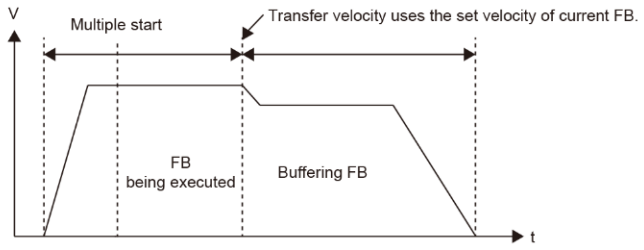
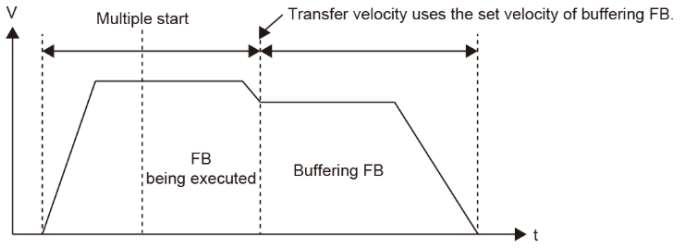
The completion signal is transmitted to the PLC CPU.

5.9 Single Axis Continuous Positioning (Program Name: ContinuousPositioning)

(1) Overview

An axis can continuously execute multiple motion FBs without stopping by buffering the next motion FB of another instance while executing the first motion FB. Select the buffer mode through the BufferMode input of a motion control FB.
Up to two motion FBs can be buffered on one axis.

(2) Operation pattern of buffer mode

<table>
<tr><th>Buffer mode</th><th>Operation</th></tr>
<tr><td>0: mcAborting
The on-going FB is interrupted (canceled), and the next FB is immediately executed.</td><td> 
</td></tr>
<tr><td>1: mcBuffered
The next FB is executed after completion of the on-going FB (the axis decelerates to a stop).</td><td> 
</td></tr>
<tr><td colspan="2">2: mcBlendingLow
The lower target velocity of the on-going FB and the next FB is used as the switching speed.</td></tr>
<tr><td>3: mcBlendingPrevious
The next FB is executed after the target position of the on-going FB is reached. The target velocity is that of the next FB.</td><td> 
</td></tr>
<tr><td>4: mcBlendingNext
The next FB is executed after the target position of the on-going FB is reached. When the target position is reached, the velocity is switched to that of the next FB.</td><td> 
</td></tr>
<tr><td colspan="2">5: mcBlendingHigh
The higher target velocity of the on-going FB and the next FB is used as the switching speed.</td></tr>
</table>

(3) Local labels

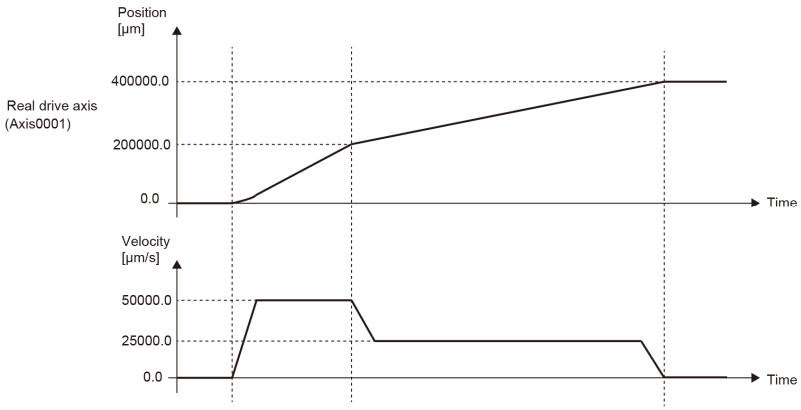
1)
2)

	Label Name	Data Type	Class	Initial Value	Constant	Comment
1	MC_MoveRelative_1	MC_MoveRelative	VAR			相対値位置決めFB1 / Relative value positioning FB1
2	MC_MoveRelative_2	MC_MoveRelative	VAR			相対値位置決めFB2 / Relative value positioning FB2
3	leDistance1	FLOAT [Double Precision]	VAR			移動量1 / Distance1
4	leDistance2	FLOAT [Double Precision]	VAR			移動量2 / Distance2
5	leVelocity1	FLOAT [Double Precision]	VAR			速度1 / Velocity1
6	leVelocity2	FLOAT [Double Precision]	VAR			速度2 / Velocity2
7	leAcceleration1	FLOAT [Double Precision]	VAR			加速度1 / Acceleration1
8	leDeceleration1	FLOAT [Double Precision]	VAR			減速度1 / Deceleration1
9	leAcceleration2	FLOAT [Double Precision]	VAR			加速度2 / Acceleration2
10	leDeceleration2	FLOAT [Double Precision]	VAR			減速度2 / Deceleration2
11	leJerk	FLOAT [Double Precision]	VAR			ジャーク / Jerk
12	bBusy1	Bit	VAR			相対値位置決めFB1 Busy出力 / Relative value positioning FB1 Busy output
13	bActive1	Bit	VAR			相対値位置決めFB1 Active出力 / Relative value positioning FB1 Active output
14	bDone2	Bit	VAR			相対値位置決めFB2 Done出力 / Relative value positioning FB2 Done output
15	bBusy2	Bit	VAR			相対値位置決めFB2 Busy出力 / Relative value positioning FB2 Busy output
16	TON_1	TON	VAR			オンディレイタイマFB / On-delay timer FB
17	bDwell_in	Bit	VAR			タイマ入力 / Timer input
18	bDwell_out	Bit	VAR			タイマ出力 / Timer output
19	bExecute_OP	Bit	VAR			実行指令 / Start
20	bCommandAborted	Bit	VAR			相対値位置決めFB 実行中断出力 / Relative value positioning FB CommandAborted output
21	bError	Bit	VAR			相対値位置決めFB Error出力 / Relative value positioning FB Error output
22						

1) These labels are automatically added when the user drags and drops the FB (MC_MoveRelative and TON) to the program editor.
2) These labels are registered manually.

(4) Program example

This program executes relative positioning in the following operation pattern.

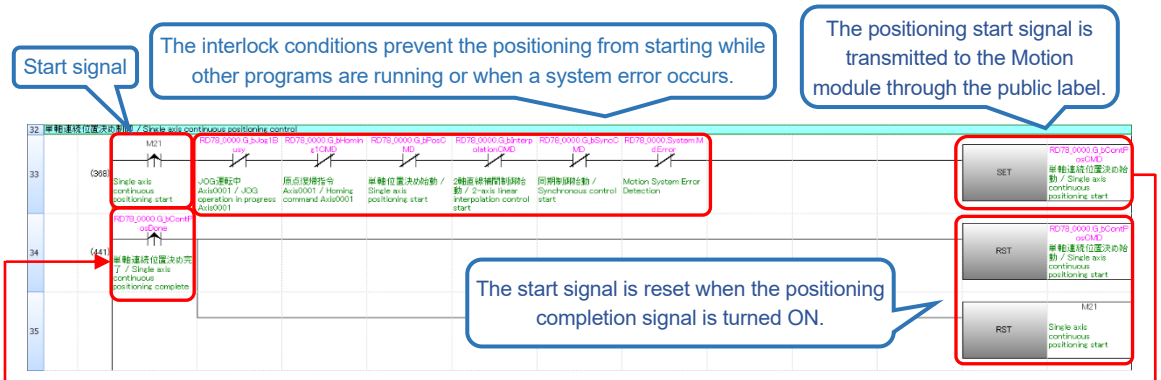


The start signal (the rising edge of the single axis continuous positioning command device) is transmitted to the Motion module through the public label. Responding to the start signal, the Motion module stores each positioning data to the specified local label. When all the data are stored, the relative value positioning FBs (MC_MoveRelative) are executed. The Active output of the first FB is connected to the Execute input of the next FB, which enables buffering of the next FB while executing the first FB. When the first FB is finished, the next FB is continuously executed. To set dwell time, the on-delay timer (100 [ms]) is used.
The completion signal is transmitted to the PLC CPU to turn OFF the start signal when any of the following conditions is satisfied: the dwell time is passed; an error occurs; or the execution is aborted.
The interlock conditions are added to prevent execution of the single axis continuous positioning while other programs are running or when a system error occurs.

[Start signal]

Single axis continuous positioning command	M21

[PLC CPU]



[Motion module]

```
//----単軸連続位置決め用データ設定・実行指令&リセット----------------------------------
//----Single axis continuous positioning control(data setting, execution, and reset)----

IF G_bContPosCMD THEN;
    leDistance1      :=  200000.0;
    leDistance2      :=  200000.0;
    leVelocity1      :=  50000.0;
    leVelocity2      :=  25000.0;
    leAcceleration1  :=  100000.0;
    leDeceleration1  :=  100000.0;
    leAcceleration2  :=  50000.0;
    leDeceleration2  :=  50000.0;
    leJerk           :=  0.0;
    bExecute_CP := TRUE;
ELSE
    bExecute_CP := FALSE;
END_IF;

//----単軸連続位置決め+バッファモード+ドウェル----
//----Single axis continuous positioning + Buffer mode + Dwell----

//相対値位置決め1, Relative value positioning 1
MC_MoveRelative_1(
    Axis         := AxisQ001.AxisRef ,
    Execute      := bExecute_CP ,
    Distance     := leDistance1 ,
    Velocity     := leVelocity1 ,
    Acceleration := leAcceleration1 ,
    Deceleration := leDeceleration1 ,
    Jerk         := leJerk ,
    Busy         => bBusy1 ,
    Active       => bActive1
);

//相対値位置決め2, Relative value positioning 2
MC_MoveRelative_2(
    Axis         := AxisQ001.AxisRef ,
    Execute      := bActive1,
    Distance     := leDistance2 ,
    Velocity     := leVelocity2 ,
    Acceleration := leAcceleration2 ,
    Deceleration := leDeceleration2 ,
    BufferMode   := MC_BUFFER_MODE__mcBlendingPrevious ,
    Done         => bDone2 ,
    Busy         => bBusy2
);

//FB実行中断出力, FB CommandAborted output
bCommandAborted := MC_MoveRelative_1.CommandAborted OR MC_MoveRelative_2.CommandAborted;

//FBエラー出力, FB Error output
bError := MC_MoveRelative_1.Error OR MC_MoveRelative_2.Error;

//相対値位置決め2完了信号でタイマ入力をセット
//Sets timer input with relative value positioning 2 complete signal
SET( bDone2, bDwell_in );

//ドウェル, Dwell
TON_1( IN := bDwell_in, PT := T#100ms, Q=> bDwell_out );

//シーケンサに送る <= ドウェル時間経過・FB実行中断・FBエラーのいずれかの出力がONする ＊制御終了信号
//Transmits continuous positioning complete signal to the PLC CPU
G_bContPosDone := bDwell_out OR bCommandAborted OR bError ;

//位置決め完了時、相対位置決め2完了信号保持(タイマ入力)をリセット
//Resets timer input when positioning is completed
RST( bDwell_out, bDwell_in );
```

When the start signal is turned ON, each positioning data is stored to the specified label.

When all the data is stored, the positioning FB is executed.

The Active output of the first FB is connected to the Execute input of the next FB, which enables buffering of the next FB.

Refer to "POINTS" in this section.

The on-delay timer "TON" is used for the dwell time.

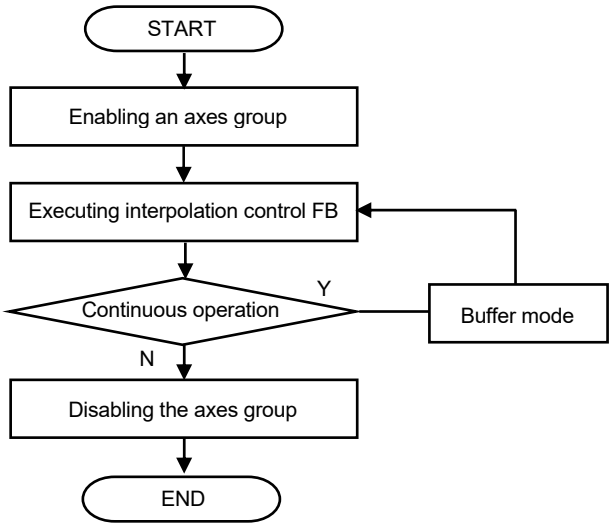
[POINTS]

Instead of connecting a label, output signals of a FB can be directly entered by describing "(FB name). (output signal name)" in the editor.

5.10 Interpolation Control (Program Name: LinearInterpolation)

5.10.1 Procedure for interpolation control

The following shows the procedure flow for executing interpolation control of two or more axes.



5.10.2 Enabling/disabling an axes group

Refer to Section 3.12 for axes group setting.
To execute interpolation control, change the axes group status to "4: GroupStandby".

(1) FBs

Type	Command	Description
MCFB (administrative)	MC_GroupEnable	Transits the specified axes group status from "0: GroupDisabled" to "4: GroupStandby".
	MC_GroupDisable	Transits the specified axes group status to "0: GroupDisabled".

5.10.3 Interpolation control

The FBs for linear and circular interpolation control are as follows. Execute the FB after the axes group is enabled.

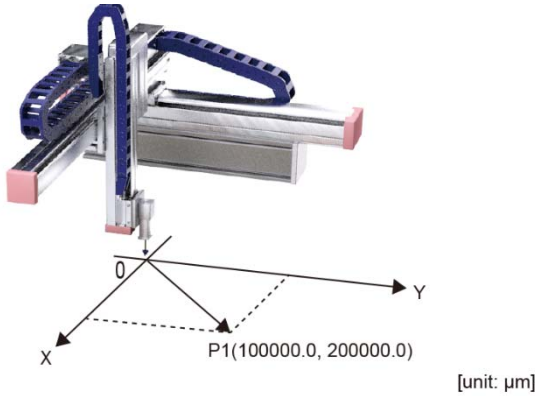
(1) FBs

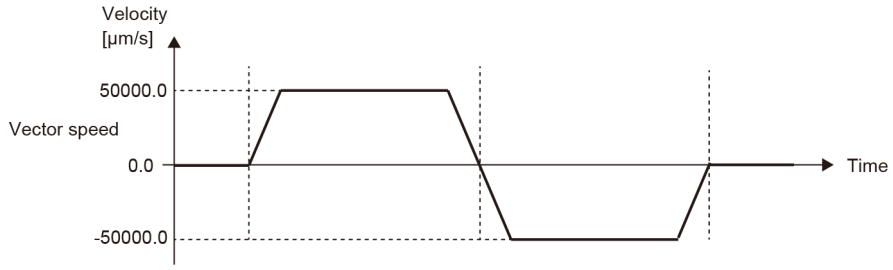
Type	Command	Description
MCFB (motion)	MCv_MoveLinearInterpolateAbsolute	Absolute value linear interpolation control
	MCv_MoveLinearInterpolateRelative	Relative value linear interpolation control
	MCv_MoveCircularInterpolateAbsolute	Absolute value circular interpolation control
	MCv_MoveCircularInterpolateRelative	Relative value circular interpolation control

5.10.4 Program example for linear interpolation

(1) Operation pattern

The machine goes back and forth between the origin point (0.0, 0.0) [um] and P1 (100000.0, 200000.0) [um].





(2) Axis No. and movement amount settings

(a) LinearAxes input of MCv_MoveLinearInterpolateRelative

Use the array of INT (signed word) type with 16 elements.

In the sample program, wAxes[0..15] (label) is used.

The interpolation control axes are selected from the structuring axis [1] to [16] set in the axes group (in Section 3.12). Specify the structuring axis No. to be used for the interpolation control by wAxes. (Note that the wAxes must be set from the index No. [0] in order.)

(b) Distance input

Use the array of LREAL (double-precision real number) type with 16 elements

In the sample program, lePosition[0..15] is used.

Define the movement amount of the structuring axis [1] to [16] in lePosition[0] to lePosition[15].

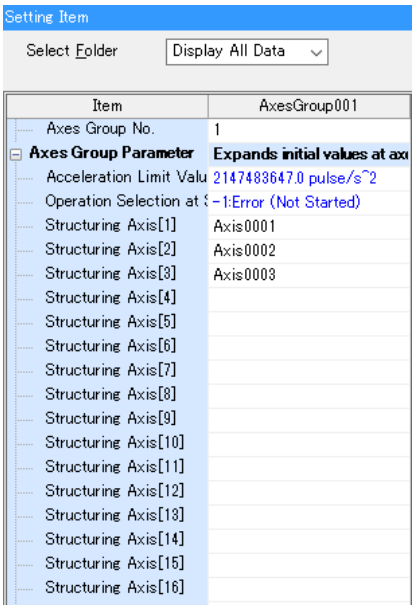
[POINTS]

Regardless of the number of the interpolation control axes, the number of elements must be 16 for the INT-type array for the LinearAxes input and the LREAL-type array for the Distance input.

[Setting example 1]

• Axes group
Structuring axis [1]: Axis0001, structuring axis [2]: Axis0002,
and structuring axis [3]: Axis0003

• Linear interpolation
Axis0001 and Axis0002

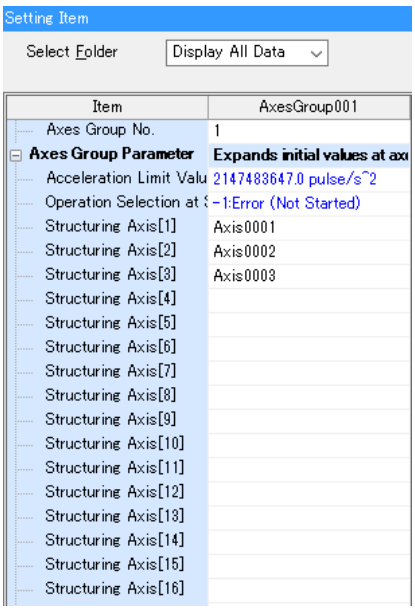



```
wAxes[0] := 1; (← structuring axis [1])
wAxes[1] := 2; (← structuring axis [2])

lePosition[0] := (movement amount of structuring
                 axis 1 (=Axis0001));
lePosition[1] := (movement amount of structuring
                 axis 2 (=Axis0002));
```

[Setting example 2]

• Axes group
Structuring axis [1]: Axis0001, structuring axis [2]: Axis0002,
and structuring axis [3]: Axis0003

• Linear interpolation
Axis0002 and Axis0003




```
wAxes[0] := 2; (← structuring axis [2])
wAxes[1] := 3; (← structuring axis [3])

lePosition[0] := 0.0;
lePosition[1] := (movement amount of structuring
                 axis 2 (=Axis0002));
lePosition[2] := (movement amount of structuring
                 axis 3 (=Axis0003));
```

(3) Local label

1) ----------
2) ----------

	Label Name	Data Type	Class	Initia	Con	Comment
1	MCv_MoveLinearInterpolateRelative_1	MCv_MoveLinear…	VAR			相対値直線補間制御FB1 / Relative value linear interpolation control FB1
2	MCv_MoveLinearInterpolateRelative_2	MCv_MoveLinear…	VAR			相対値直線補間制御FB2 / Relative value linear interpolation control FB2
3	MC_GroupEnable_1	MC_GroupEnable	VAR			軸グループ有効FB / Axes group enable FB
4	MC_GroupDisable_1	MC_GroupDisable	VAR			軸グループ無効FB / Axes group disable FB
5	wAxes	Word [Signed][0…]	VAR			補間軸 / Interpolation axis
6	lePosition1	FLOAT [Double…	VAR			位置データ1 / Position data 1
7	lePosition2	FLOAT [Double…	VAR			位置データ2 / Position data 2
8	leVelocity	FLOAT [Double…	VAR			速度 / Velocity
9	leAcceleration	FLOAT [Double…	VAR			加速度 / Acceleration
10	leDeceleration	FLOAT [Double…	VAR			減速度 / Deceleration
11	leJerk	FLOAT [Double…	VAR			ジャーク / Jerk
12	bGroupEnableDone	Bit	VAR			軸グループ有効完了 / Axes group enable done
13	bGroupDisableDone	Bit	VAR			軸グループ無効完了 / Axes group disable done
14	bBusy1	Bit	VAR			相対値直線補間制御FB1 Busy出力 / Relative value linear interpolation control FB1 Busy output
15	bActive1	Bit	VAR			相対値直線補間制御FB1 Active出力 / Relative value linear interpolation control FB1 Active output
16	bDone2	Bit	VAR			相対値直線補間制御FB2 Done出力 / Relative value linear interpolation control FB2 Done output
17	bBusy2	Bit	VAR			相対値直線補間制御FB2 Busy出力 / Relative value linear interpolation control FB2 Busy output
18	TON_1	TON	VAR			オンディレイタイマFB / On-delay timer FB
19	bDwell_out	Bit	VAR			タイマ出力 / Timer output
20	bDwell_in	Bit	VAR			タイマ入力 / Timer input
21	bExcute_LP	Bit	VAR			実行指令 / Start
22	bCommandAborted	Bit	VAR			相対値直線補間制御FB 実行中断出力 / Relative value linear interpolation control FB CommandAborted output
23	bError	Bit	VAR			相対値直線補間制御FB Error出力 / Relative value linear interpolation control FB Error output
24						

1) These labels are automatically added when the user drags and drops the corresponding FB to the program editor.

2) These labels are registered manually.

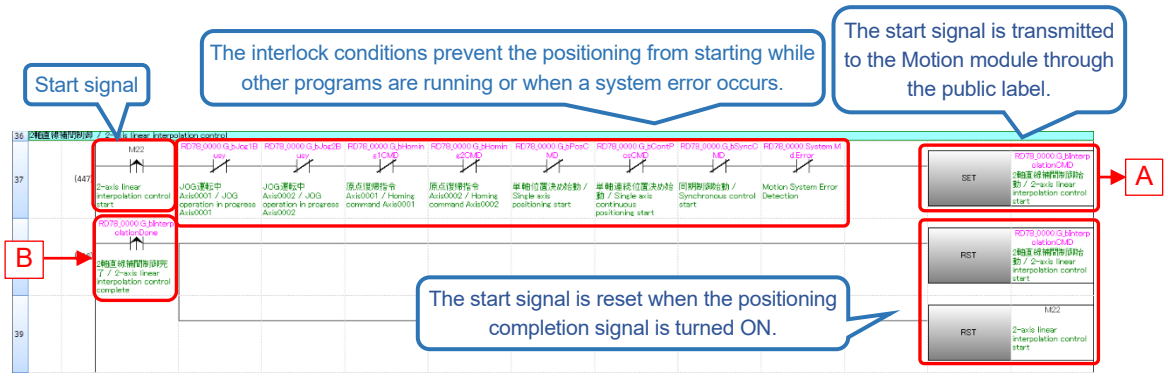
(4) Program example

	Program description	Module
1)	The start signal (the rising edge of the two-axis linear interpolation control start device) is transmitted to the Motion module through the public label.	PLC CPU
2)	• Responding to the start signal, the Motion module stores the positioning data to the specified local labels. When all the data are stored, the axes group is enabled. • After the axes group is enabled, the two relative value positioning FBs (MCv_MoveLinearInterpolateRelative) are started in buffer mode. • The axes group is disabled when any of the following conditions is satisfied: the dwell time is passed; an error occurs; or the execution is aborted. • When the axes group is disabled, the start signal and the on-delay timer input are reset.	Motion module
3)	The interlock conditions are added to prevent execution of the two-axis interpolation control while other programs are running or when a system error occurs.	PLC CPU

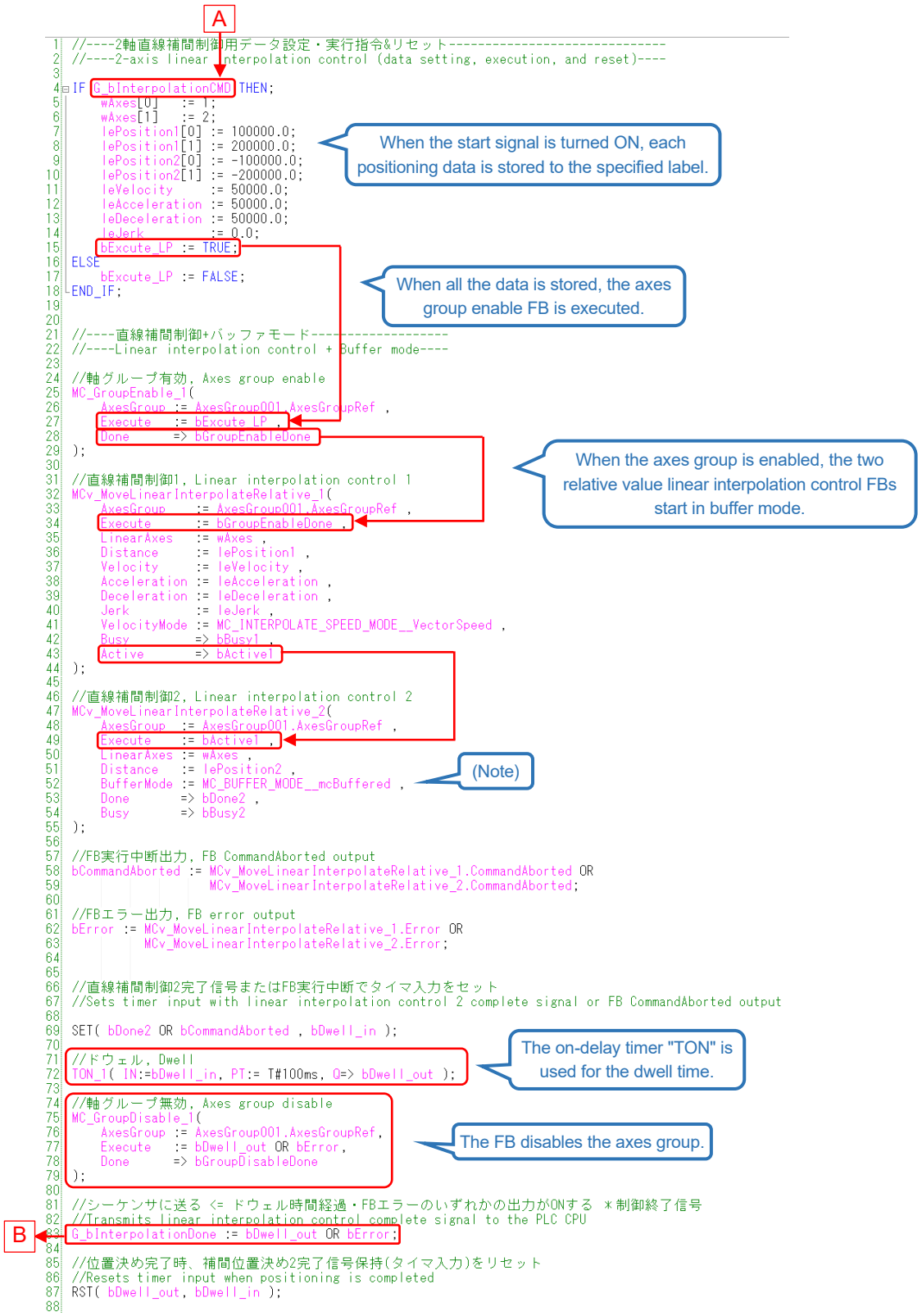
[Start signal]

Two-axis linear interpolation control start	M22

[PLC CPU]


Start signal
The interlock conditions prevent the positioning from starting while other programs are running or when a system error occurs.
The start signal is transmitted to the Motion module through the public label.
A
B
The start signal is reset when the positioning completion signal is turned ON.

[Motion module]



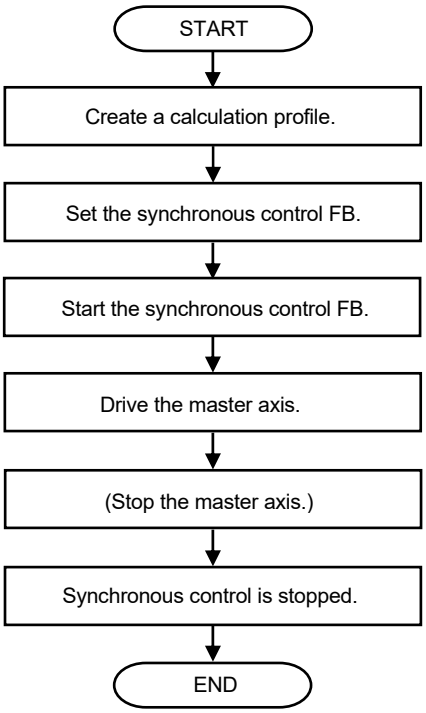

```
//----2軸直線補間制御用データ設定・実行指令&リセット------------------------------
//----2-axis linear interpolation control (data setting, execution, and reset)----

IF G_bInterpolationCMD THEN;
    wAxes[0]    := 1;
    wAxes[1]    := 2;
    lePosition1[0] := 100000.0;
    lePosition1[1] := 200000.0;
    lePosition2[0] := -100000.0;
    lePosition2[1] := -200000.0;
    leVelocity     := 50000.0;
    leAcceleration := 50000.0;
    leDeceleration := 50000.0;
    leJerk         := 0.0;
    bExcute_LP := TRUE;
ELSE
    bExcute_LP := FALSE;
END_IF;


//----直線補間制御+バッファモード-------------------
//----Linear interpolation control + Buffer mode----

//軸グループ有効, Axes group enable
MC_GroupEnable_1(
    AxesGroup := AxesGroup001.AxesGroupRef ,
    Execute   := bExcute_LP ,
    Done      => bGroupEnableDone
);

//直線補間制御1, Linear interpolation control 1
MCv_MoveLinearInterpolateRelative_1(
    AxesGroup    := AxesGroup001.AxesGroupRef ,
    Execute      := bGroupEnableDone ,
    LinearAxes   := wAxes ,
    Distance     := lePosition1 ,
    Velocity     := leVelocity ,
    Acceleration := leAcceleration ,
    Deceleration := leDeceleration ,
    Jerk         := leJerk ,
    VelocityMode := MC_INTERPOLATE_SPEED_MODE__VectorSpeed ,
    Busy         => bBusy1 ,
    Active       => bActive1
);

//直線補間制御2, Linear interpolation control 2
MCv_MoveLinearInterpolateRelative_2(
    AxesGroup  := AxesGroup001.AxesGroupRef ,
    Execute    := bActive1 ,
    LinearAxes := wAxes ,
    Distance   := lePosition2 ,
    BufferMode := MC_BUFFER_MODE__mcBuffered ,
    Done       => bDone2 ,
    Busy       => bBusy2
);

//FB実行中断出力, FB CommandAborted output
bCommandAborted := MCv_MoveLinearInterpolateRelative_1.CommandAborted OR
                   MCv_MoveLinearInterpolateRelative_2.CommandAborted;

//FBエラー出力, FB error output
bError := MCv_MoveLinearInterpolateRelative_1.Error OR
          MCv_MoveLinearInterpolateRelative_2.Error;


//直線補間制御2完了信号またはFB実行中断でタイマ入力をセット
//Sets timer input with linear interpolation control 2 complete signal or FB CommandAborted output

SET( bDone2 OR bCommandAborted , bDwell_in );

//ドウェル, Dwell
TON_1( IN:=bDwell_in, PT:= T#100ms, Q=> bDwell_out );

//軸グループ無効, Axes group disable
MC_GroupDisable_1(
    AxesGroup := AxesGroup001.AxesGroupRef,
    Execute   := bDwell_out OR bError,
    Done      => bGroupDisableDone
);

//シーケンサに送る <= ドウェル時間経過・FBエラーのいずれかの出力がONする ＊制御終了信号
//Transmits linear interpolation control complete signal to the PLC CPU
G_bInterpolationDone := bDwell_out OR bError;

//位置決め完了時、補間位置決め2完了信号保持(タイマ入力)をリセット
//Resets timer input when positioning is completed
RST( bDwell_out, bDwell_in );
```

(Note): When the values for the Velocity, Acceleration, and Deceleration inputs are omitted, those of the previous FB are applied.

5.11 Synchronous Control (Program Name: Synchronous)

5.11.1 Procedures for executing synchronous control

The following shows the procedure flow for executing synchronous control.

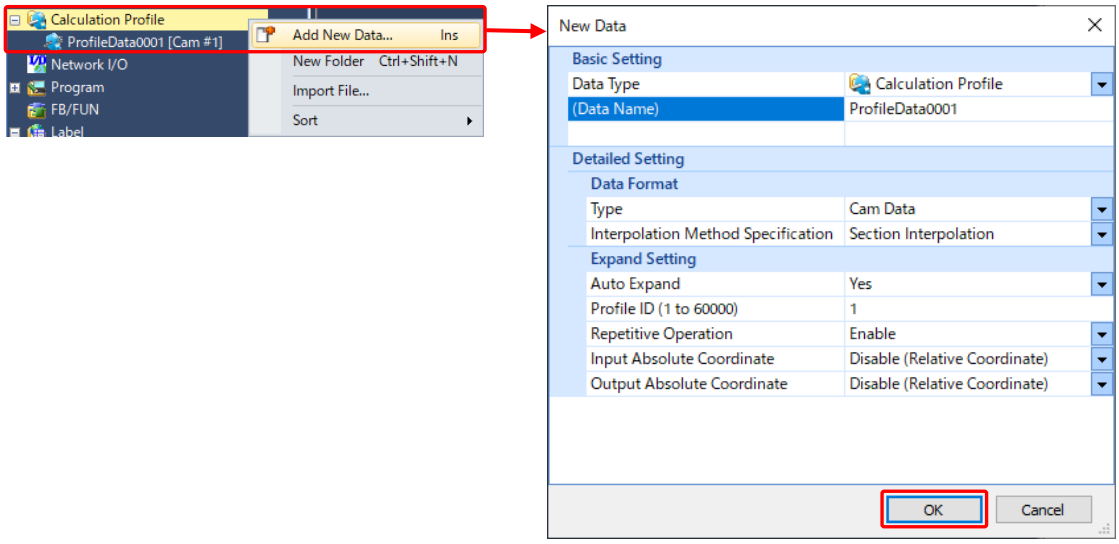


5.11.2 Calculation profile

Waveform data used for control is collectively called calculation profile data.
This section explains how to create a cam data.

(1) Creating a new calculation profile

In the navigation window of the motion control setting function, right-click "Calculation Profile" and select "Add New Data".



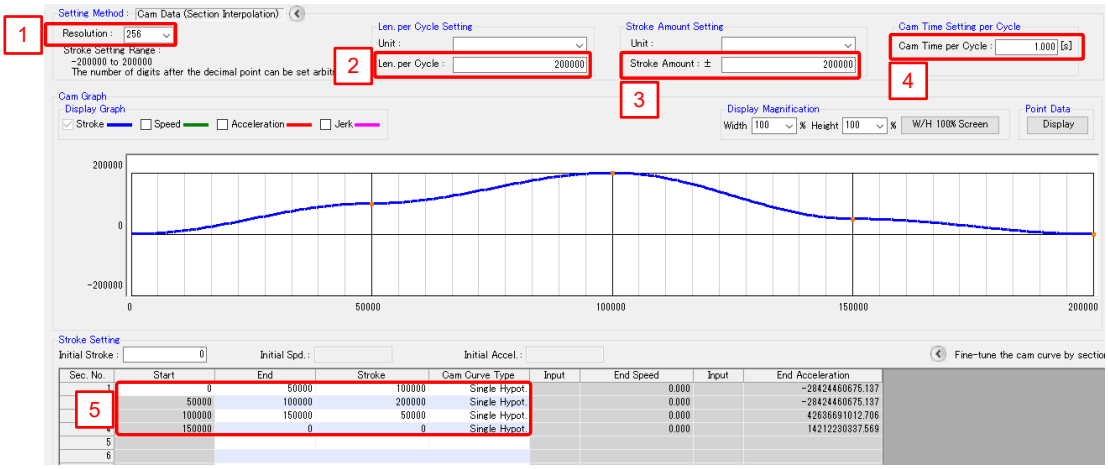
The following shows the setting items on the "New data" screen.

Item	Description
Automatic open	Yes: The calculation profile data is automatically opened at power-on. No: The open FB for calculation profile data needs to be executed.
Periodic	Invalid: The control ends when it executes until the end of calculation profile data. Valid: The execution of calculation profile data is continuously repeated.
Output absolute coordinate	Invalid (relative): When the calculation profile (cam) is started, an output value is calculated based on the current value. When executing a feed cam operation, select this setting. Valid (absolute): The output value at the time the calculation profile (cam) is started is calculated to be always the start point for one cycle of the calculation profile data. When the start point and the end point of the calculation profile data are different, the command is output in one operation cycle in order to return to the first output value at the next one cycle start.

The example in this document uses the initial value. Click the [OK] button.

(2) Creating a cam data

Set the calculation profile waveform.



The following shows the necessary setting items.

No.	Item	Description
1	Resolution	Set the resolution of the cam data.
2	Length per Cycle	Set the length per cycle and its unit. (Set the movement amount of the master axis for one cam cycle)
3	Stroke Amount	Set the stroke amount and its unit. (Set the movement amount of the slave axis for one cam cycle)
4	Cam Time per Cycle	Set the time for one cam cycle. This setting is used when velocity, acceleration, and jerk are calculated.
5	Stroke Setting	Set the stroke.

The following shows the example settings.

No.	Item	Setting value
1	Resolution	256
2	Length per Cycle	200000 (blank for the unit setting)
3	Stroke Amount	200000 (blank for the unit setting)
4	Cam Time per Cycle	1.000
5	Stroke Setting	Refer to the following table.

Section No.	Start point	End point	Stroke	Cam curve type
1	0	50000	100000	Single hypotenuse
2	50000	100000	200000	Single hypotenuse
3	100000	150000	50000	Single hypotenuse
4	150000	0	0	Single hypotenuse

[POINTS]

When executing a linear cam operation (the same operation as the master axis, or the operation that changes the master axis speed based on a specified speed ratio), create a calculation profile for the linear cam, or use MC_GearIn.

The calculation profile data for the linear cam is not provided in the system.

5.11.3 Single axis synchronous control FBs

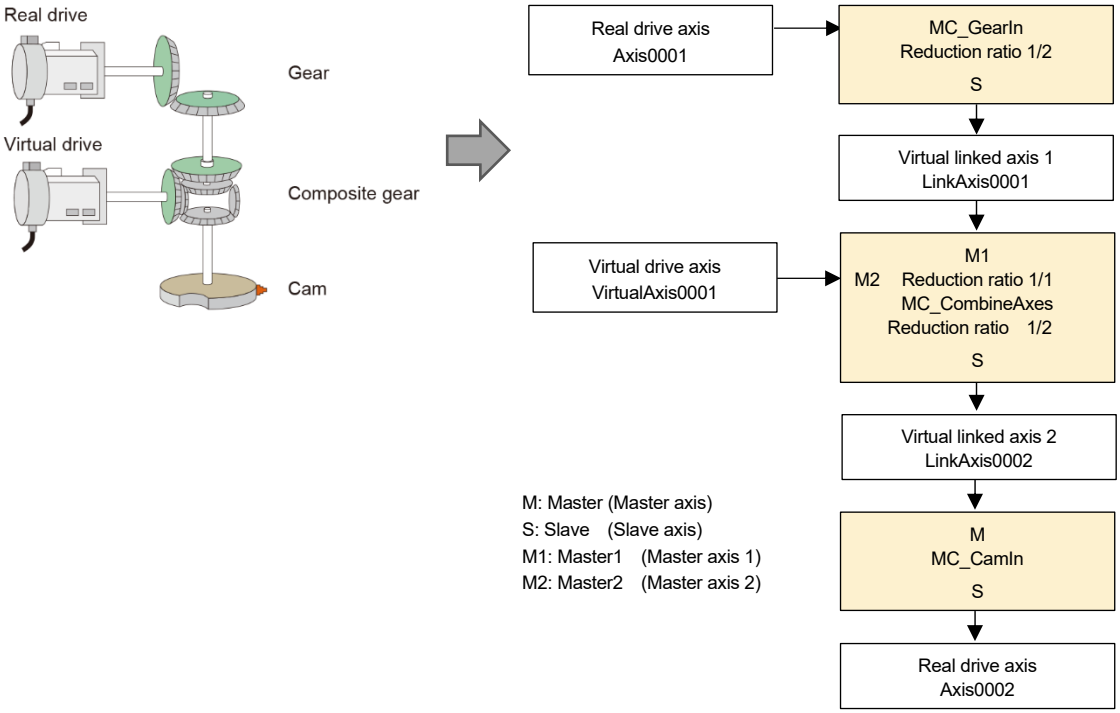
The single axis synchronous control FBs operate as software-based mechanical modules such as gears, speed change gears, and cams. These FBs transmit the position information (command) of Slave that is synchronized with Master.

(1) FBs

Type	Command	Description
MCFB (motion)	MC_CamIn	Executes cam operation.
	MC_GearIn	Executes gear operation based on the specified speed ratio between the master axis and the slave axis.
	MC_CombineAxes	Combines motion of two axes by a selectable combination method, and outputs the result to the third axis.
	MCv_ChangeCycle	Changes the cam current value per cycle to the specified value during MC_CamIn control. It is used to compensate the cam current value per cycle into an arbitrary value.
	MCv_******Filter	Executes the specific filter processing to the input of Master, and outputs the result to Slave.
	MC_Stop	Stops the synchronous control.

5.11.4 Axes configuration

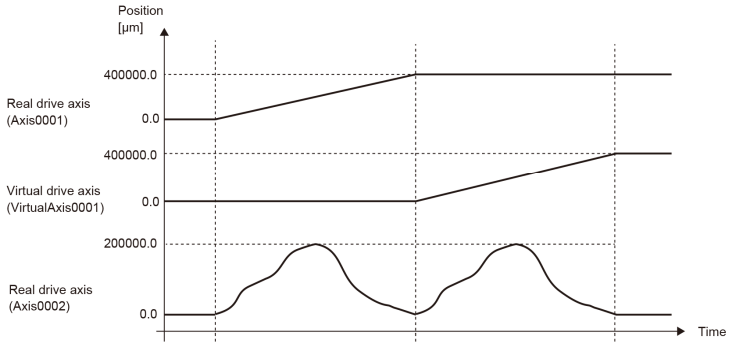
This chapter explains the following cam system.



5.11.5 Program example

(1) Operation pattern

The X-axis moves from the origin point for 400000.0 [um] while the Y-axis (Axis0002) is operated according to the cam pattern created in Section 5.11.2. When the X-axis reaches the target position (400000.0 [um]), the virtual drive axis starts operation and the Y-axis repeats the cam operation. At this time, the X-axis is stopped and only the Y-axis is operated.



(2) Local labels

	Label Name	Data Type	Class	Initial	Const	Comment
1	MC_GearIn_1	MC_GearIn	VAR			ギア動作FB / Gear operation FB
2	MC_CombineAxes_1	MC_CombineAxes	VAR			加減算位置決めFB / FB combining the motion of two master axes
3	MC_CamIn_1	MC_CamIn	VAR			カム動作FB / Cam operation FB
4	bGearInBusy	Bit	VAR			ギア動作FB Busy出力 / Gear operation FB Busy output
5	bCombineAxesBusy	Bit	VAR			加減算位置決め動作FB Busy出力 / FB combining the motion of two master axes Busy output
6	bCamInBusy	Bit	VAR			カム動作FB Busy出力 / Cam operation FB Busy output
7	MC_MoveRelative_1	MC_MoveRelative	VAR			相対値位置決めFB / Relative value positioning FB
8	MC_MoveRelative_2	MC_MoveRelative	VAR			相対値位置決めFB / Relative value positioning FB
9	MC_Stop_1	MC_Stop	VAR			軸停止FB1 / Axis Stop FB1
10	MC_Stop_2	MC_Stop	VAR			軸停止FB2 / Axis Stop FB2
11	MC_Stop_3	MC_Stop	VAR			軸停止FB3 / Axis Stop FB3
12	leAcceleration	FLOAT [Double Pr…	VAR			加速度 / Acceleration
13	leDeceleration	FLOAT [Double Pr…	VAR			減速度 / Deceleration
14	leJerk	FLOAT [Double Pr…	VAR			ジャーク / Jerk
15	lePosition1	FLOAT [Double Pr…	VAR			移動量 / Distance
16	leVelocity	FLOAT [Double Pr…	VAR			速度 / Velocity
17	bInSync	Bit	VAR			カム動作FB inSync出力 / Cam operation FB inSync output
18	bDone1	Bit	VAR			相対値位置決めFB1 Done出力 / Relative value positioning FB1 Done output
19	bBusy1	Bit	VAR			相対値位置決めFB1 Busy出力 / Relative value positioning FB1 Busy output
20	bDone2	Bit	VAR			相対値位置決めFB2 Done出力 / Relative value positioning FB2 Done output
21	bBusy2	Bit	VAR			相対値位置決めFB2 Busy出力 / Relative value positioning FB2 Busy output
22	bStopReq1	Bit	VAR			軸停止要求1 / Axis Stop request 1
23	bStopDone1	Bit	VAR			軸停止完了1 / Axis Stop complete 1
24	bStopDone2	Bit	VAR			軸停止完了2 / Axis Stop complete 2
25	bStopDone3	Bit	VAR			軸停止完了3 / Axis Stop complete 3
26	bSyncMoveCMD	Bit	VAR			相対位置決め始動 / Relative value positioning start
27	bExecute_S	Bit	VAR			実行指令 / Start
28	bCommandAborted	Bit	VAR			相対値位置決めFB 実行中断出力 / Relative value positioning FB CommandAborted output
29	bError	Bit	VAR			相対値位置決めFB Error出力 / Relative value positioning FB Error output
30						

1) These labels are automatically added when the user drags and drops the corresponding FB to the program editor.

2) These labels are registered manually.

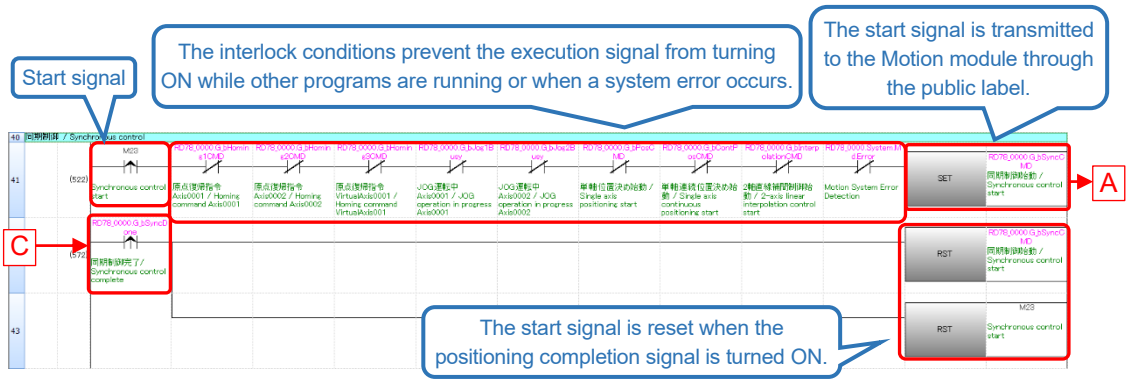
(3) Program example

	Program description	Module
1)	The start signal (the rising edge of the synchronous control start device) is transmitted to the Motion module through the public label.	PLC CPU
2)	• Responding to the start signal, the Motion module stores the positioning data to the specified labels. When all the data are stored, the execution signals of the single axis synchronous control FBs (MC_GearIn, MC_CombineAxes, and MC_CamIn) are turned ON. • When the Axis0002 status is turned to "7: SynchronizedMotion", Axis0001 (Master axis) starts positioning. At this time, Axis0002 is operated according to the specified calculation profile. • When positioning of Axis0001 is completed, the virtual drive axis (VirtualAxis0001) starts positioning. At this time, Axis0001 is stopped, but Axis0002 is operated according to the specified calculation profile. • The synchronous control is stopped by executing MC_Stop on the real drive axis (Axis0002) and the virtual linked axes (LinkAxis0001 and LinkAxis0002) when any of the following conditions is satisfied: the positioning of the virtual drive axis (VirtualAxis0001) is completed; an error occurs; or the execution is aborted. • The start signal is reset when the real drive axis (Axis0002) status is out of synchronization.	Motion module
3)	The interlock conditions are added to prevent execution of the synchronous control while other programs are running or when a system error occurs.	PLC CPU

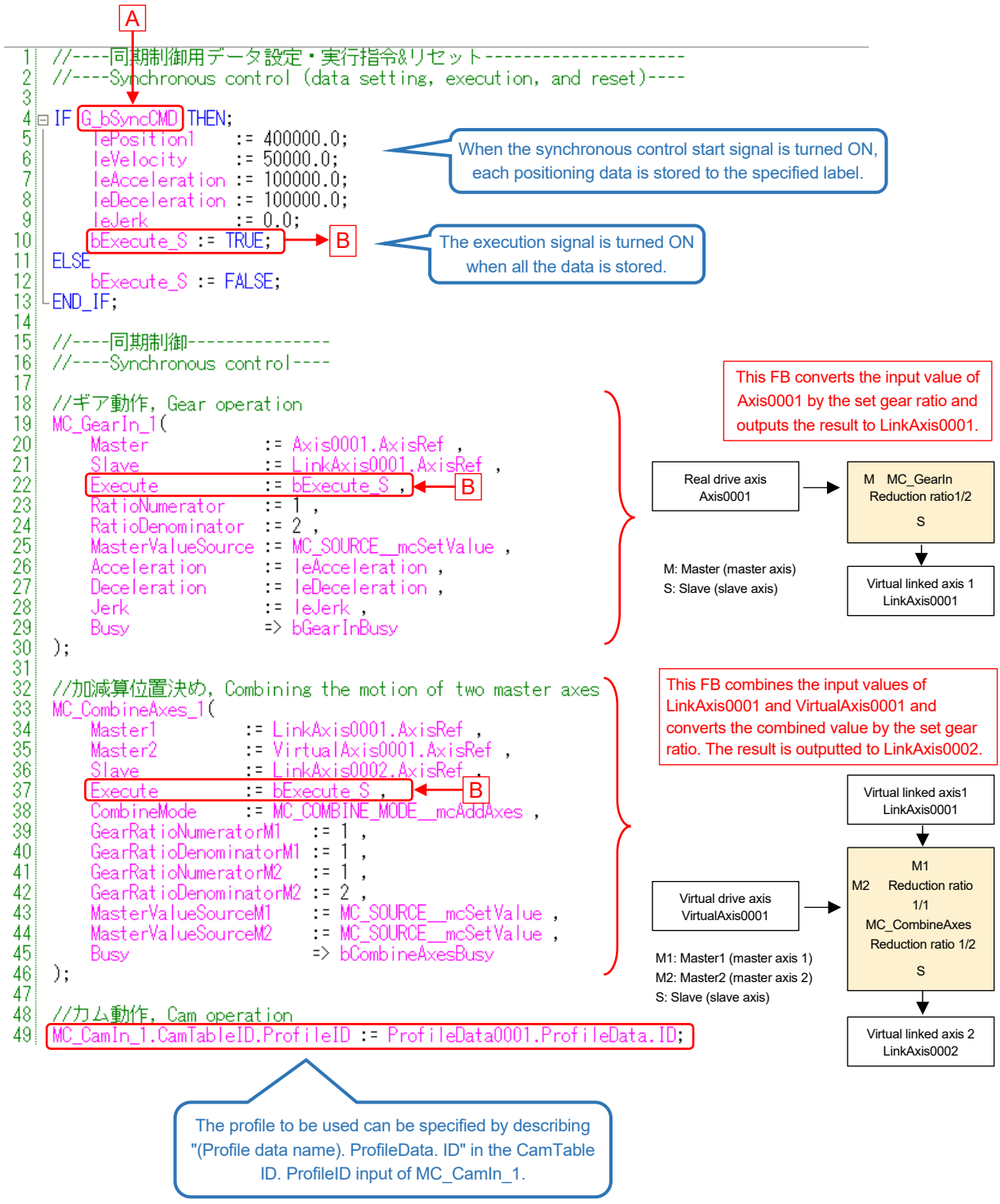
[Start signal]

Synchronous control start	M23

[PLC CPU]



[Motion module]




```
//----同期制御用データ設定・実行指令&リセット----------
//----Synchronous control (data setting, execution, and reset)----

IF G_bSyncCMD THEN;
    fePosition1     := 400000.0;
    feVelocity      := 50000.0;
    feAcceleration  := 100000.0;
    feDeceleration  := 100000.0;
    feJerk          := 0.0;
    bExecute_S := TRUE;
ELSE
    bExecute_S := FALSE;
END_IF;

//----同期制御---------------
//----Synchronous control----

//ギア動作, Gear operation
MC_GearIn_1(
    Master           := Axis0001.AxisRef ,
    Slave            := LinkAxis0001.AxisRef ,
    Execute          := bExecute_S ,
    RatioNumerator   := 1 ,
    RatioDenominator := 2 ,
    MasterValueSource := MC_SOURCE__mcSetValue ,
    Acceleration     := feAcceleration ,
    Deceleration     := feDeceleration ,
    Jerk             := feJerk ,
    Busy             => bGearInBusy
);

//加減算位置決め, Combining the motion of two master axes
MC_CombineAxes_1(
    Master1          := LinkAxis0001.AxisRef ,
    Master2          := VirtualAxis0001.AxisRef ,
    Slave            := LinkAxis0002.AxisRef ,
    Execute          := bExecute_S ,
    CombineMode      := MC_COMBINE_MODE__mcAddAxes ,
    GearRatioNumeratorM1   := 1 ,
    GearRatioDenominatorM1 := 1 ,
    GearRatioNumeratorM2   := 1 ,
    GearRatioDenominatorM2 := 2 ,
    MasterValueSourceM1    := MC_SOURCE__mcSetValue ,
    MasterValueSourceM2    := MC_SOURCE__mcSetValue ,
    Busy                   => bCombineAxesBusy
);

//カム動作, Cam operation
MC_CamIn_1.CamTableID.ProfileID := ProfileData0001.ProfileData.ID;
```

(To the next page)

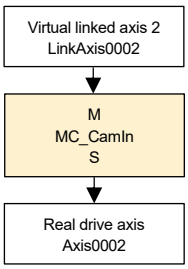
(Continued)

The CamTableID input is omitted since it is set in line 45.

```
51  MC_CamIn_1(
52      Master           := LinkAxis0002.AxisRef ,
53      Slave            := Axis0002.AxisRef ,
54      Execute          := bExecute_S ,
55      MasterOffset     := 0.0 ,
56      SlaveOffset      := 0.0 ,
57      MasterScaling    := 1.0 ,
58      SlaveScaling     := 1.0 ,
59      MasterStartDistance := 0.0 ,
60      MasterSyncPosition  := 0.0 ,
61      StartMode           := MC_START_MODE__mcImmediate ,
62      MasterValueSource   := MC_SOURCE__mcSetValue ,
63      InSync              => bInSync ,
64      Busy                => bCamInBusy
65  );
66
```

B

This FB drives Axis 0002 by the input value of LinkAxis0002 with the specified calculation profile.



M: Master (master axis)
S: Slave (slave axis)

Axis0001 executes positioning when the status of Axis0002 turns to "7: SynchronizedMotion".

```
67  //入力軸始動, Input axis start
68  bSyncMoveCMD := G_bSyncCMD & (Axis0002.Md.AxisStatus = MC_AXIS_STATUS__SynchronizedMotion) ;
69
70  //相対値位置決め 実ドライブ軸 Axis0001
71  //Relative value positioning real drive axis Axis0001
72  MC_MoveRelative_1(
73      Axis         := Axis0001.AxisRef ,
74      Execute      := bSyncMoveCMD ,
75      Distance     := lePosition1 ,
76      Velocity     := leVelocity ,
77      Acceleration := leAcceleration ,
78      Deceleration := leDeceleration ,
79      Jerk         := leJerk ,
80      Done         => bDone1 ,
81      Busy         => bBusy1
82  );
83
84  //相対値位置決め  仮想ドライブ軸 VirtualAxis0001
85  //Relative value positioning virtual drive axis VirtualAxis0001
86  MC_MoveRelative_2(
87      Axis         := VirtualAxis0001.AxisRef ,
88      Execute      := bDone1 & Axis0001.Md.CmdInPos ,
89      Distance     := lePosition1 ,
90      Velocity     := leVelocity ,
91      Acceleration := leAcceleration ,
92      Deceleration := leDeceleration ,
93      Jerk         := leJerk ,
94      Done         => bDone2 ,
95      Busy         => bBusy2
96  );
97
98  //FB実行中断出力, FB CommandAborted output
99  bCommandAborted := MC_GearIn_1.CommandAborted OR
100                     MC_CombineAxes_1.CommandAborted OR
101                     MC_CamIn_1.CommandAborted OR
102                     MC_MoveRelative_1.CommandAborted OR
103                     MC_MoveRelative_2.CommandAborted;
104
105 //FBエラー出力, FB error output
106 bError := MC_GearIn_1.Error OR MC_CombineAxes_1.Error OR
107           MC_CamIn_1.Error OR MC_MoveRelative_1.Error OR
108           MC_MoveRelative_2.Error;
109
110 //位置決め完了時、または実行中断時、エラー発生時に軸停止信号をON, Turns axis stop signal ON
111 bStopReq1 := ( bDone2 & VirtualAxis0001.Md.CmdInPos ) OR bError OR bCommandAborted;
112
```

Axis0001 positioning

VirtualAxis0001 starts positioning when the positioning of Axis0001 is completed.

VirtualAxis0001 positioning

The axis stop signal (bStopReq1) is turned ON when any of the following conditions is satisfied: the positioning of VirtualAxis0001 is completed; an error occurs; or the execution is aborted.

(To the next page)

(Continued)

```
113 //同期終了, Synchronization done
114 MC_Stop_1(
115     Axis    := Axis0002.AxisRef ,
116     Execute := bStopReq1 ,
117     Done    => bStopDone1
118 );
119
120 MC_Stop_2(
121     Axis    := LinkAxis0001.AxisRef ,
122     Execute := bStopReq1 ,
123     Done    => bStopDone2
124 );
125
126 MC_Stop_3(
127     Axis    := LinkAxis0002.AxisRef ,
128     Execute := bStopReq1 ,
129     Done    => bStopDone3
130 );
131
132 //同期終了後、始動要求信号をリセット
133 //Resets the start request signal when synchronization is completed
134 RST( bStopDone1 OR System.Md.Error, bSyncMoveCMD );
135 RST( bStopDone1 OR System.Md.Error, bStopReq1 );
136
137 //シーケンサに送る <= 位置決め完了信号・FB実行中断・FBエラーのいずれかの出力がONする ＊制御終了信号
138 //Transmits synchronous control complete signal to the PLC CPU
139 G_bSyncDone := ( bStopDone2 & VirtualAxis0001.Md.CmdInPos ) OR bCommandAborted OR bError ;
140
```

Axis0002, LinkAxis0001, and LinkAxis0002 are out of synchronization.

C

5.12 Error Reset (Program Name: ErrorReset)

This program resets the error on each axis.

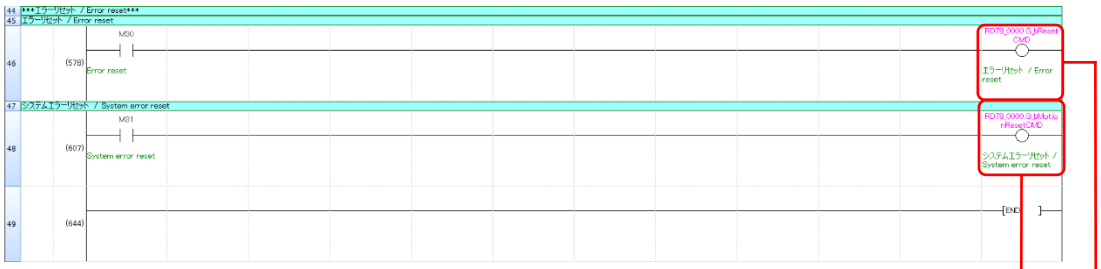
(1) FBs

Type	Command	Description
MCFB (administrative)	MC_Reset	Resets errors and warnings of the axis.
	MC_GroupReset	Resets errors and warnings of the axes group.
	MCv_MotionErrorReset	Reset all errors and warnings of the motion system.

(2) Program example

The following shows the program example for error reset.
The ON/OFF status of the device assigned as the start signal of the error reset is transmitted to the Motion module through the public label.

[PLC CPU]



[Motion module]

```
1
2  // エラーリセット, Error reset
3  MC_Reset_1(Axis:= Axis0001.AxisRef          ,Execute:= G_bResetCMD );
4  MC_Reset_2(Axis:= Axis0002.AxisRef          ,Execute:= G_bResetCMD );
5  MC_Reset_3(Axis:= VirtualAxis0001.AxisRef   ,Execute:= G_bResetCMD );
6  MC_Reset_4(Axis:= LinkAxis0001.AxisRef      ,Execute:= G_bResetCMD );
7  MC_Reset_5(Axis:= LinkAxis0002.AxisRef      ,Execute:= G_bResetCMD );
8
9  // 軸グループエラーリセット, Axes group error reset
10 MC_GroupReset_1(AxesGroup:= AxesGroup001.AxesGroupRef ,Execute:= G_bResetCMD );
11
12 // コントローラエラーリセット, Controller error reset
13 MCv_MotionErrorReset_1(Execute:= G_bMotionResetCMD );
14
```

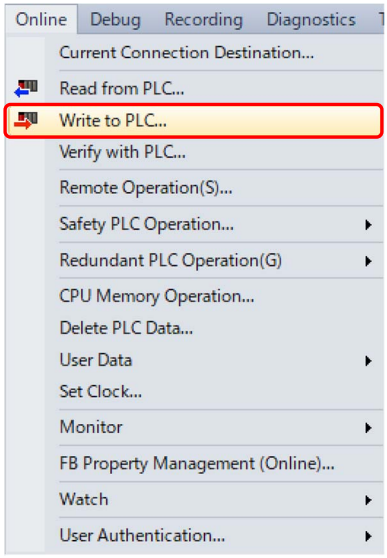
5.13 Checking Operation

5.13.1 Conversion and Writing of Programs

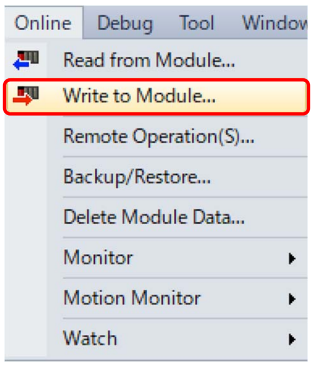
Write the programs to the PLC CPU and the Motion module.
After the programs are created in MELSOFT GX Works3 and the motion control setting function, convert all the programs from [Convert] → [Rebuild All].

Confirm that no error occurs after executing "Rebuild All", and write the program to the PLC CPU and the Motion module.

• MELSOFT GX Works3: [Online] → [Write to PLC]
• Motion control setting function: [Online] → [Write to Module]



[MELSOFT GX Works3]



[Motion control setting function]

(Note): When the parameters and the public labels of the Motion module are changed, convert and write the programs to the PLC CPU after reflecting the public labels.

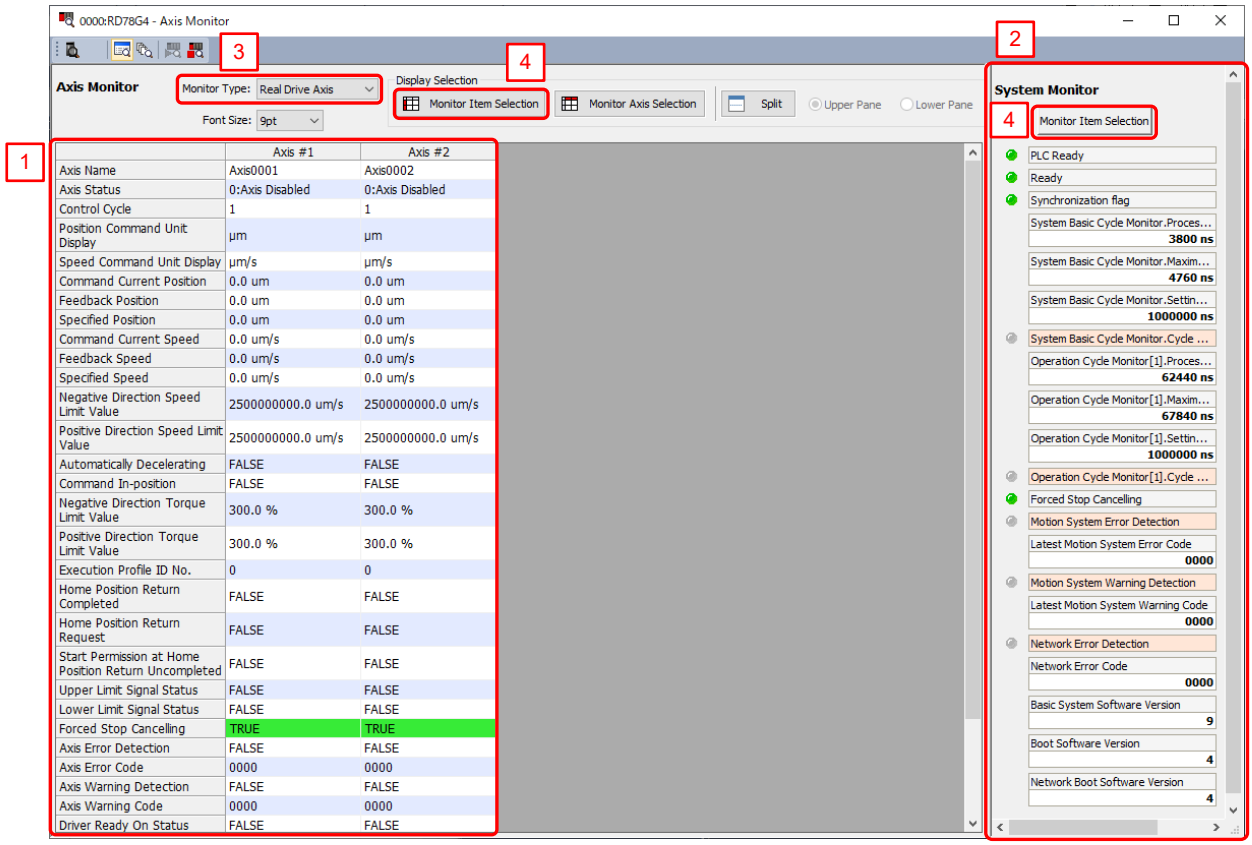
5.13.2 Axis monitor

The monitor screen displays the current values and the error codes of all axes in operation all at once. Users can check the current values and whether any error occurs during the operation.

(1) Displaying monitor screens

Select [Online] → [Motion Monitor] → [Axis Monitor] in the motion control setting function.

(2) Monitor screen



No.	Description
1	The monitor items for each axis are displayed.
2	The system monitor items are displayed.
3	The axis type to be monitored can be changed.
4	The monitor items can be added/deleted.

5.13.3 Program monitor

The program monitor enables the user to check the current status of the program in execution on the program editor.

(1) Displaying monitor screen

[PLC CPU]

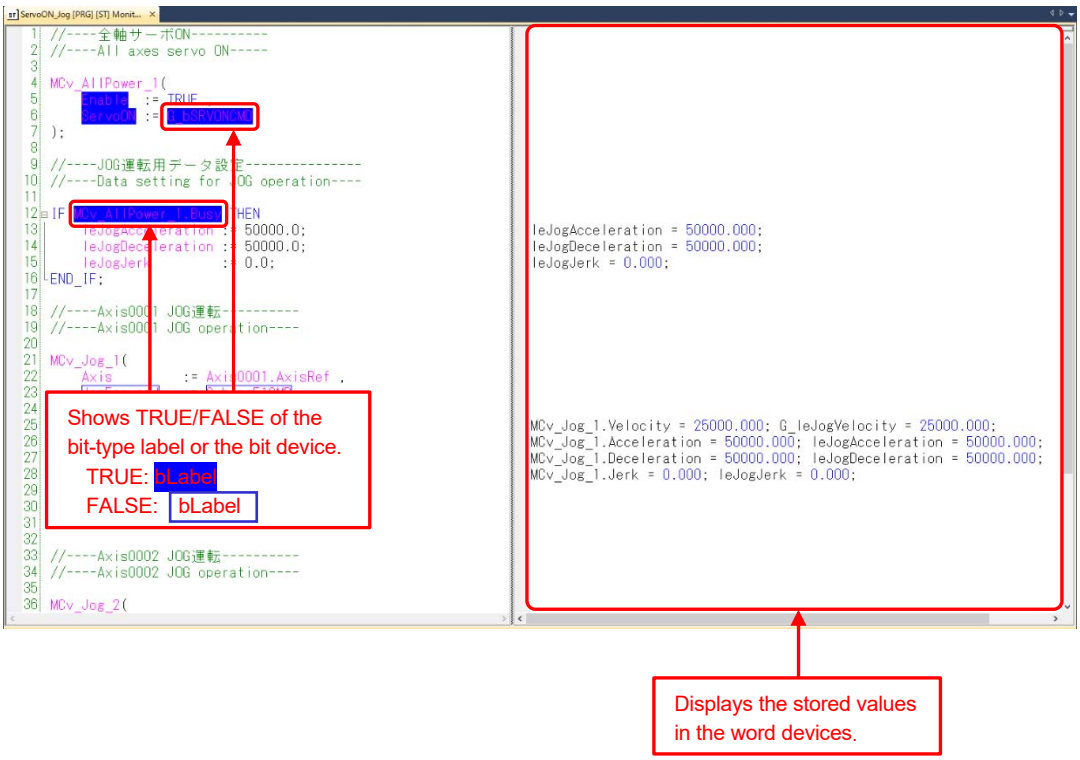
Select [Online] → [Monitor] → [Start Monitoring (All Windows)] in MELSOFT GX Works3, or click the icon [ ] on the tool bar.

[Motion module]

Select [Online] → [Monitor] → [Start Monitoring (All Windows)] in the motion control setting function, or click the icon [ ] on the tool bar.

(2) Monitor screen

The following shows the monitor screen of the motion control setting function.



5.13.4 Watch

The watch function enables the user to check the current values of the registered devices and labels. Register the devices and labels to be monitored to a watch window.

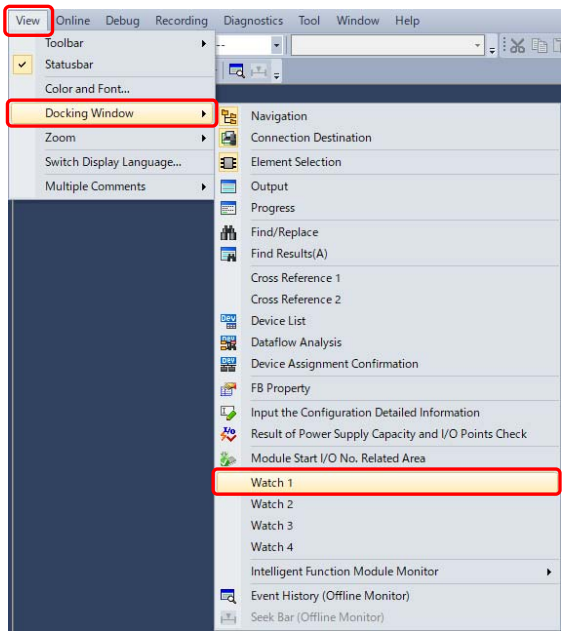
(1) Displaying monitor screens

[PLC CPU]

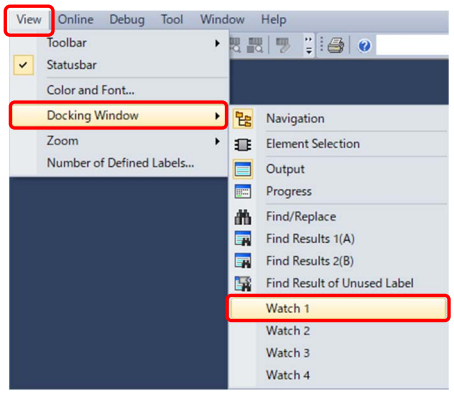
Select [View] → [Docking Window] → [Watch 1 to Watch 4] in MELSOFT GX Works3.

[Motion module]

Select [View] → [Docking Window] → [Watch 1 to Watch 4] in the motion control setting function.



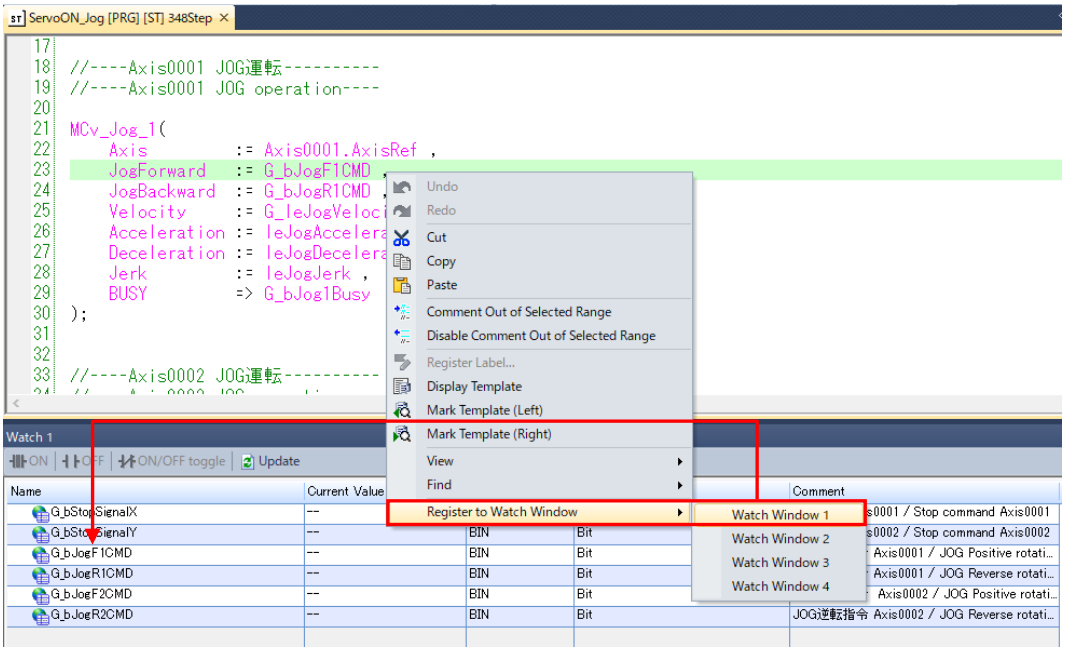
[MELSOFT GX Works3]



[Motion control setting function]

(2) Registration of labels and structures to a watch window

Enter the label or the structure to be registered in the "Name" column of a watch window or right-click a label or a structure to be registered on the program editor, and select [Register to Watch Window] → [Watch Window 1 to 4].



(3) Monitoring start

Select [Online] → [Watch] → [Start Watching] in MELSOFT GX Works3 or the motion control setting function.

(4) Changing current values

Directly enter a value in "Current Value" during monitoring.

The bit device can be switched to ON/OFF with double-click while holding the [Shift] key or with [Shift] + [Enter] after selecting the row.

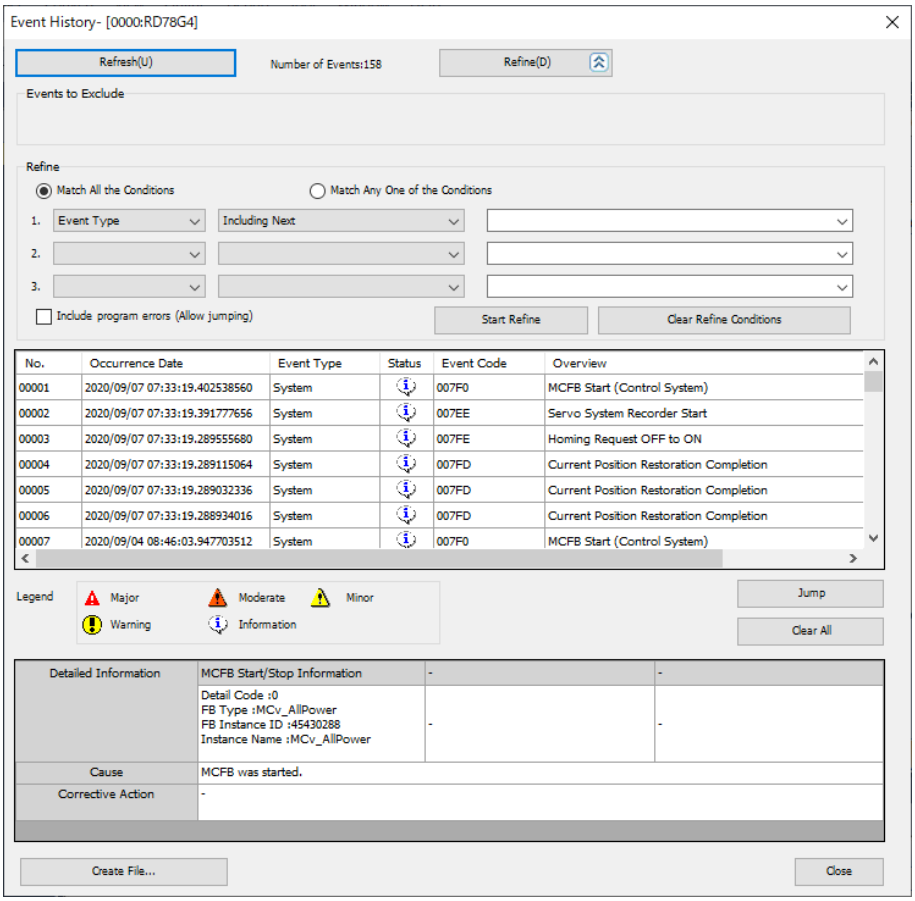
[POINTS]

In the sample program, "G_bStopSignalX" and "G_bStopSignalY" are registered in the watch window 1. Turning these signals ON stops the axis operation.

When "G_bStopSignalX" and "G_bStopSignalY" are operated from the PLC CPU, change the motion control attribute setting from "READ(Motion=>)" to "WRITE(=>Motion)", and reflect the public label.

5.13.5 Event History

The event history of the Motion module can be checked from [Online] → [Motion Monitor] → [Event History] in the motion control setting function. If an error occurs, check the details of the error.
The error occurrence time in the event history of the Motion module synchronizes with those recorded in the servo amplifiers. Check the error details of the event history together with the data in the servo amplifier.



To check the event history of the PLC CPU, click [Diagnosis] → [System monitor] in MELSOFT GX Works3, and click "Event history" on the "System monitor" screen.

MEMO

APPENDICES

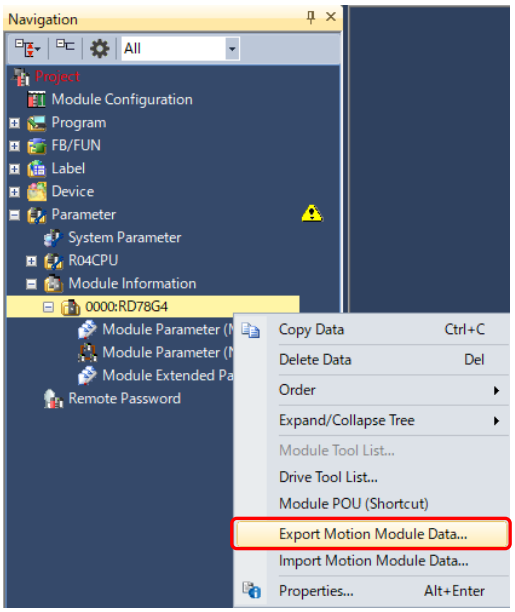
APPENDIX 1 Diverting Programs

This chapter explains how to divert the parameters and the programs of the Motion module from the sample program to another project. The following procedure can also be used when the Motion module is replaced. This chapter shows an example of when the RD78G4 is replaced with RD78GHV.

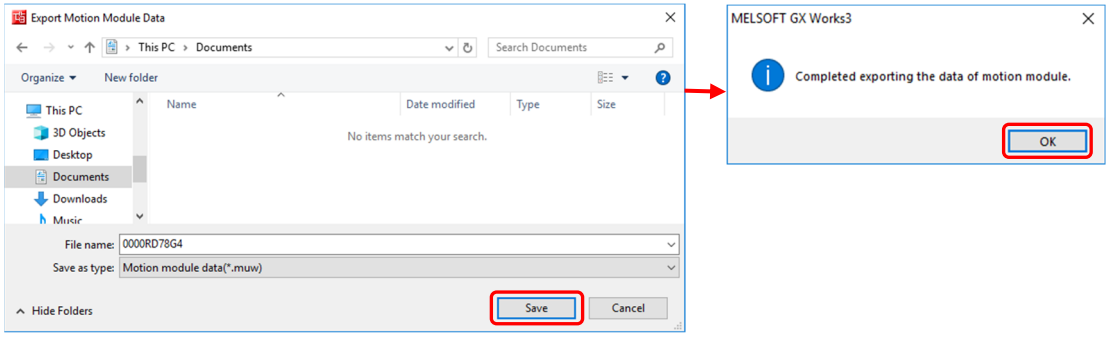
APPENDIX 1.1 Exporting the Motion module data

Output the parameters and the programs of the Motion module to one file.

1) Start MELSOFT GX Works3, and open the diverting data. In the navigation window, double-click "Parameter" → "Module information". Right-click "0000_RD78G4", and select [Export Motion Module Data].



2) On the "Export Motion Module Data" screen, select the folder to export, and click the [Save] button.
On the confirmation screen, click the [OK] button to output the Motion module data to an muw file.

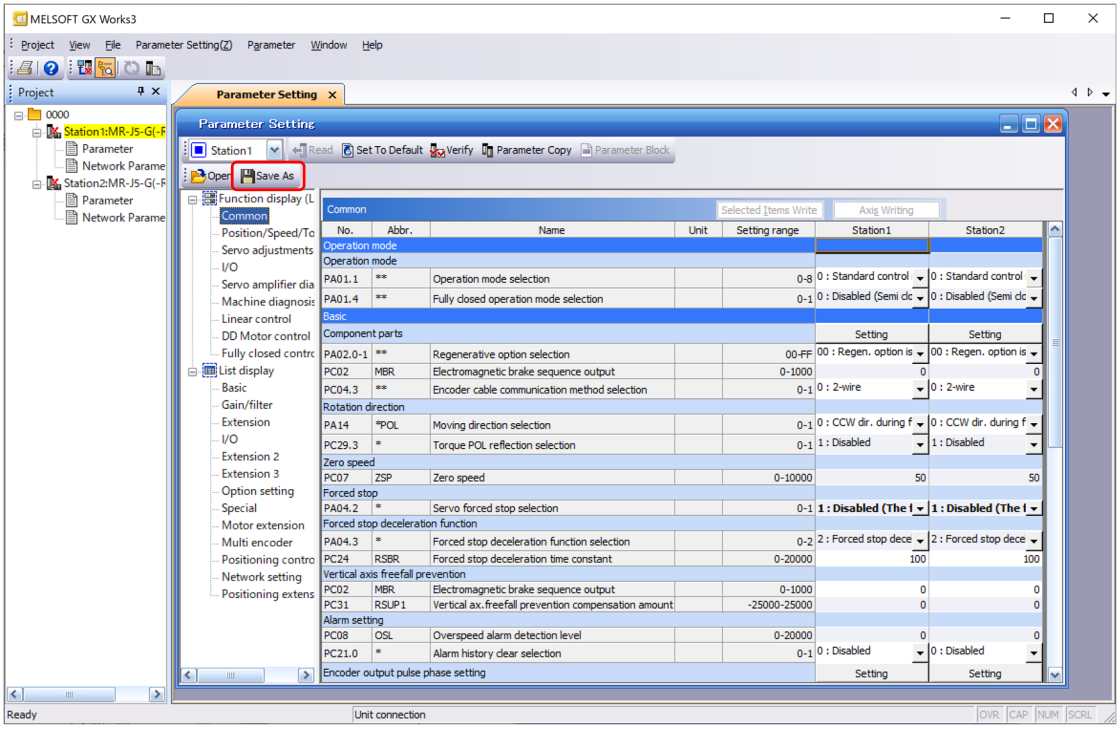


APP.

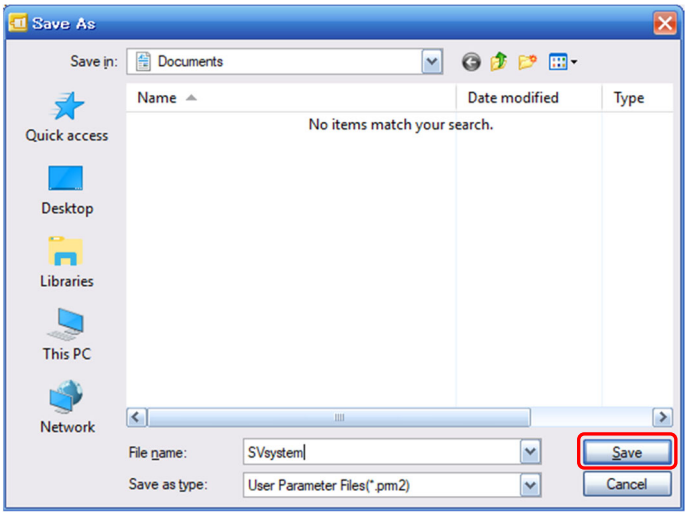
APPENDIX 1.2 Saving servo parameters

Servo parameters are not included in the muw file outputted in Appendix 1.1.
When writing the servo parameters from the Motion module to the servo amplifier, save the servo parameters of the diverting program in a separate file.

1) On the "Parameter Setting" screen, click the [Save As] button.



2) On the "Save As" screen, select a folder and click the [Save] button. The servo parameters are outputted to a prm2 file.

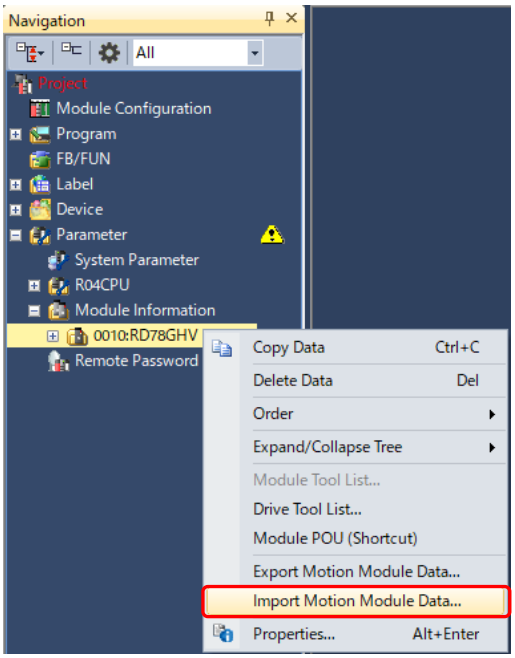


APPENDIX 1.3 Reflection to the editing program

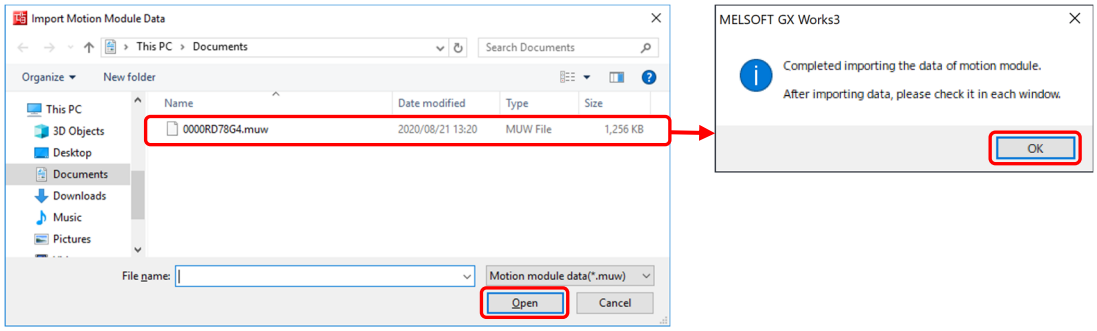
Start MELSOFT GX Works3, and create a module configuration diagram of the editing program.

(1) Importing the Motion module data

1) In the navigation window, double-click "Parameter" → "Module information".
Right-click "0010:RD78GHV", and select [Import Motion Module Data].



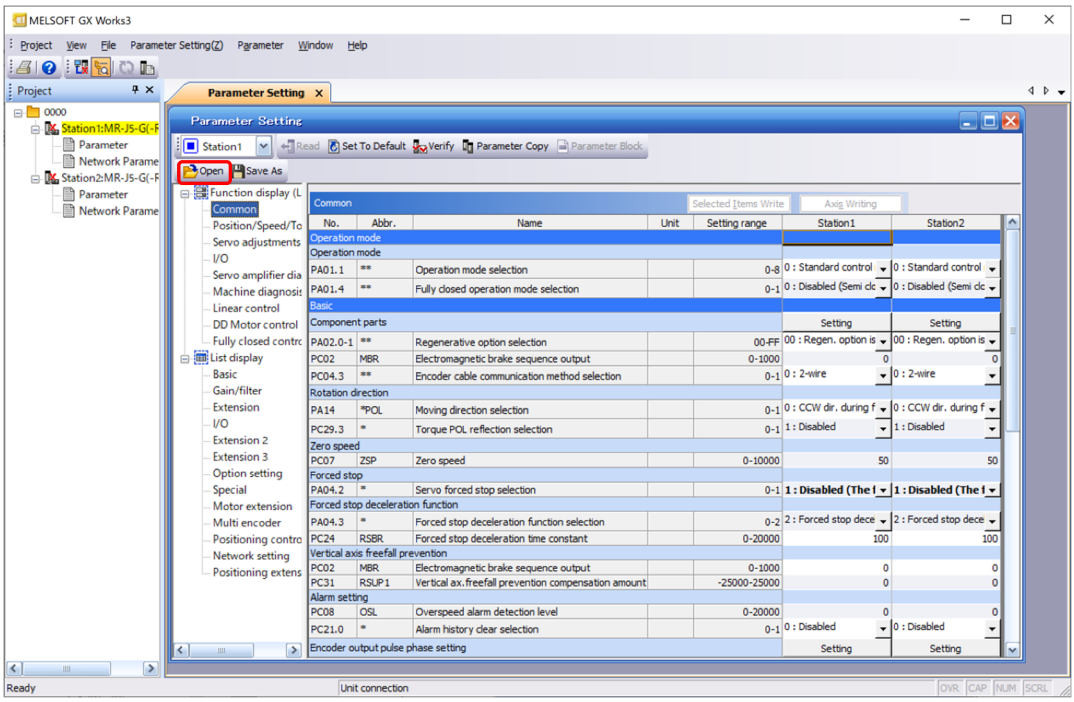
2) On the "Import Motion Module Data" screen, select the muw file outputted in Appendix 1.1, and click the [Open] button.
On the confirmation screen, click the [OK] button to complete the import of the Motion module data.



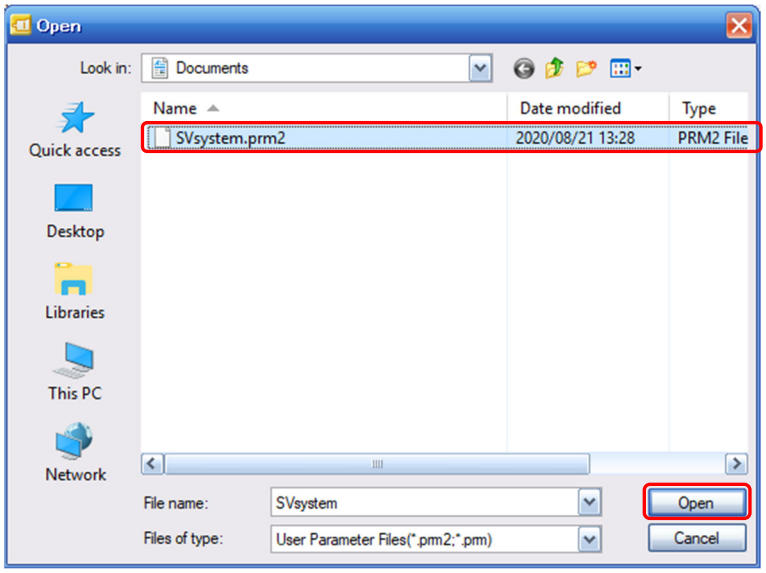
(2) Reading servo parameters

To read the servo parameters from the Motion module, read the servo parameter file outputted in Appendix 1.2.

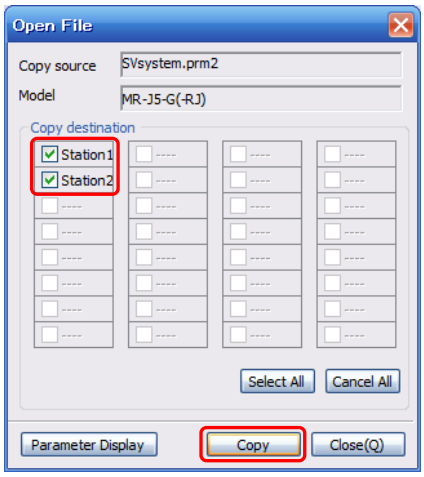
1) On the "Parameter Setting" screen of the editing program, click the [Open] button.



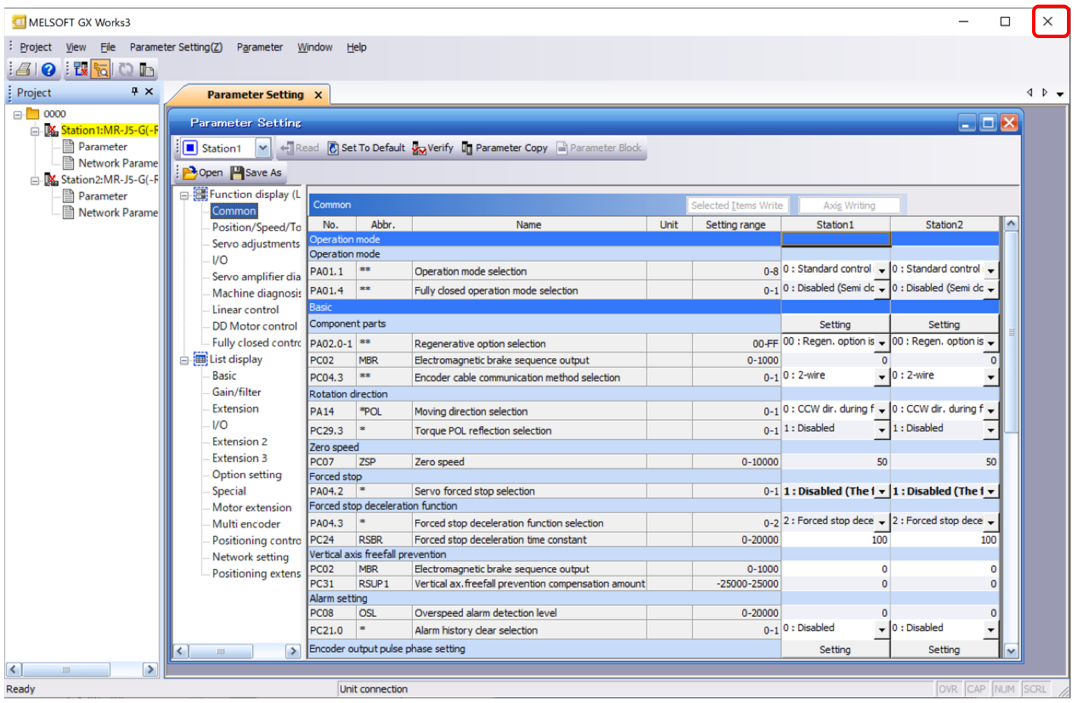
2) On the "Open" screen, select the prm2 file outputted in Appendix 1.2, and click the [Open] button.



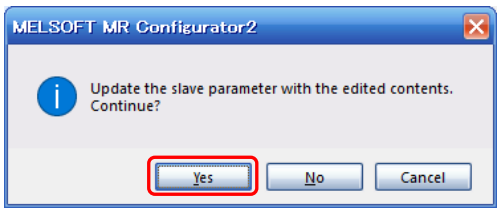
3) Select the stations to read the servo parameters, and click the [Copy] button.



4) Make sure that the servo parameters are read, and click the [×] button on the right top of the "Parameter Setting" screen.



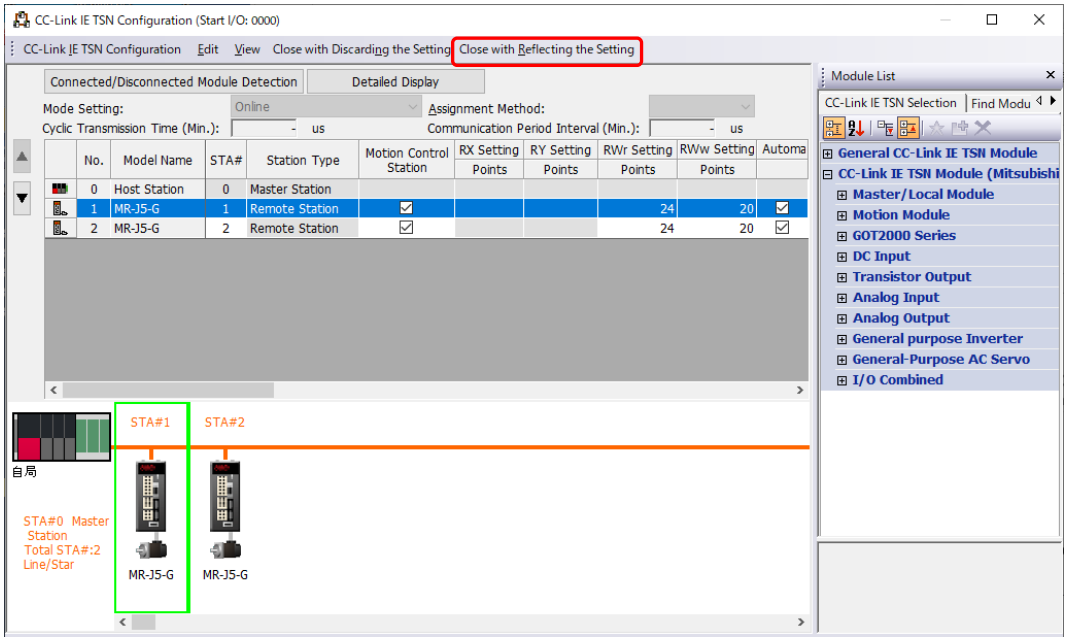
Click the [Yes] button on the confirmation screen to update the parameters.



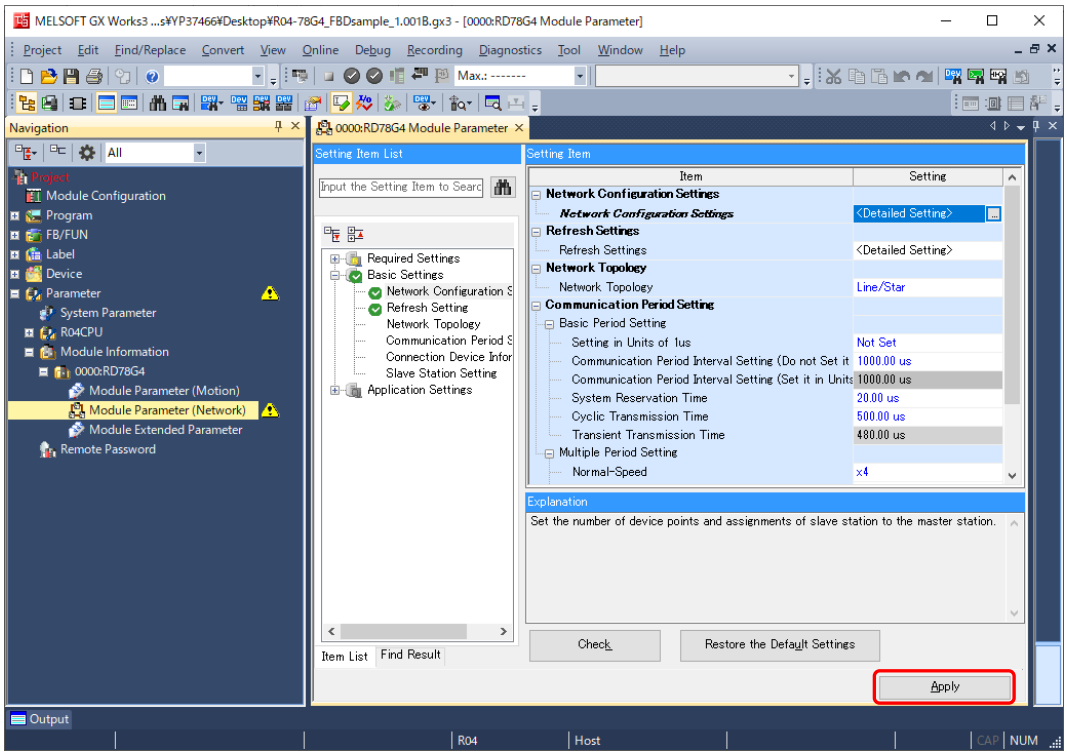
(3) Fixing parameters

Fix the CC-Link IE TSN configuration and the module parameter (network).

1) Click [Close with Reflecting the Setting] on the "CC-Link IE TSN Configuration" window to reflect the data.



2) In the parameter editor (Module Parameter), click the [Apply] button to reflect the parameters of the Motion module.



(4) Correcting programs

If the start I/O number of the diverting program and the editing program are different, change the following:

• PLC READY [Y0] of the sequence program
• READY [X0]
• The device No. of Synchronization flag [X1]
• The module No. of the buffer memory address (from U**\G)
• Public labels

Be sure to change the above especially when the module is changed to RD78GHV or RD78GHW. The RD78GHV or RD78GHW has reserved 16 points and available 32 points for input and output, and when the module is installed to the slot No. 0 to 1, the device of the PLC READY is [Y10].
Additionally, the start of the public label name changes from [RD78_0000.****] to [RD78_0010.****].
After reflecting the public labels as in Appendix 1.4, change all of the label names in the program using character string replacement.

APPENDIX 1.4 Conversion of All the Programs and Reflection of Public Labels

First convert all the programs of the Motion module, and then reflect public labels. After the public labels are reflected, convert all the programs of the PLC CPU.
If the programs of the PLC CPU are converted first, the public labels are not defined, and an error will occur.

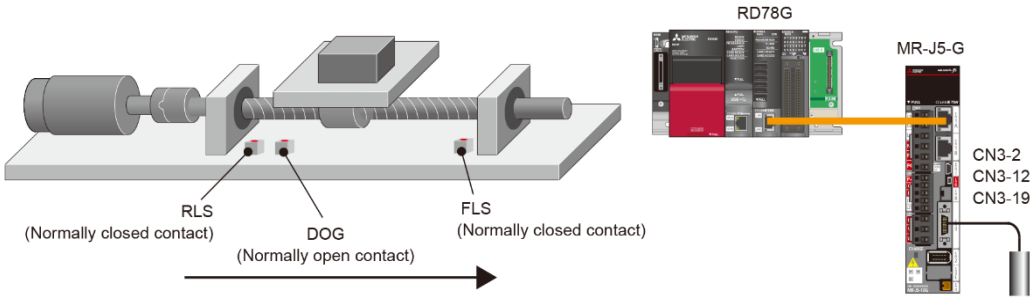
APPENDIX 2 Using External Input Signals

External input signals can be used for the hardware stroke limit signals and the proximity dog signal.

Usable external signals	Reference
DI signal of the servo amplifier	Appendix 2.1
Input signal of the PLC CPU	Appendix 2.2
Input signal of the remote input module	Appendix 2.3

APPENDIX 2.1 Using the DI signal of the servo amplifier

This section explains the case where the LSP/LSN signals of the servo amplifier are used for the hardware stroke limit signals and the case where homing is executed with the DOG signal of the servo amplifier.



(1) Hardware stroke limit

The negative logic (normally closed contact) is recommended for wiring of the hardware stroke limit.

[CAUTION]

If the positive logic (normally open contact) is used for wiring of the hardware stroke limit, it may cause a serious accident at sensor failure or disconnection.

(a) The connection destination of the servo amplifier signals and the location of the sensors

The setting location of the sensor varies according to the setting of the servo parameter [Pr. PA14 (Travel direction selection)]. Set the limit switches connected to the servo amplifier as follows:

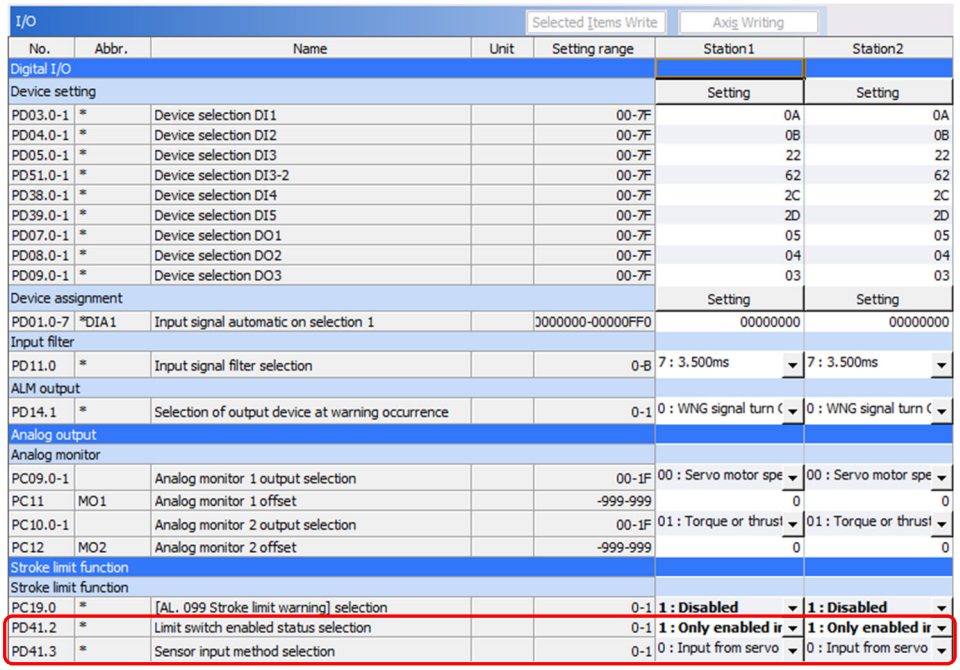
1) [Pr. PA14]: 0 (CCW or positive direction when the positioning address increases)
   - LSP signal: Upper stroke limit of the positioning address increasing side (FLS)
   - LSN signal: Lower stroke limit of the positioning address decreasing side (RLS)

2) [Pr. PA14]: 1 (CW or negative direction when the positioning address increases)
   - LSP signal: Lower stroke limit of the positioning address decreasing side (RLS)
   - LSN signal: Upper stroke limit of the positioning address increasing side (FLS)

(b) Servo parameters

Make sure that the servo parameters are set as follows:

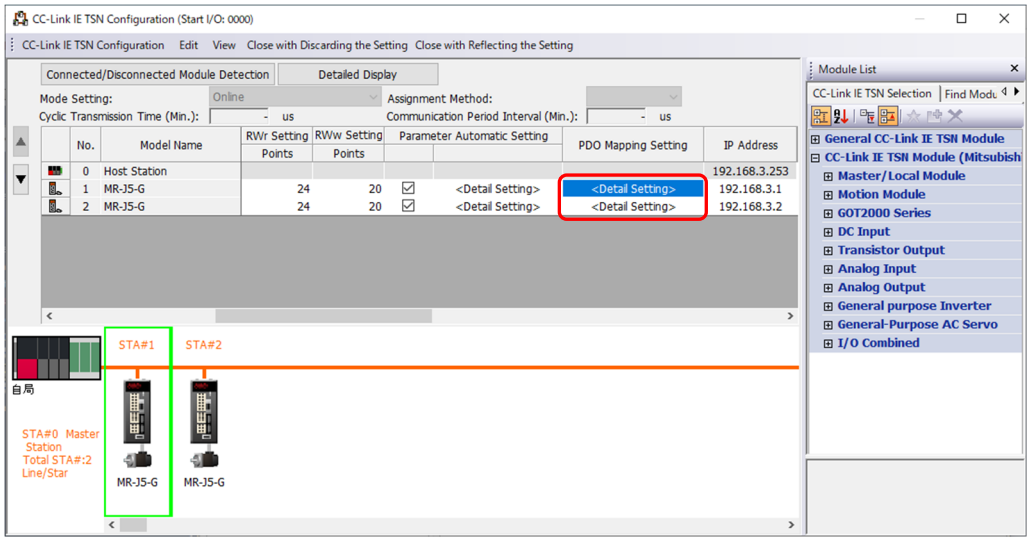
No.	Item	Setting value
Pr. PD41.2	Limit switch enabled status selection	1: Only enabled in home position return mode
Pr. PD41.3	Sensor input method selection	0: Input from servo amplifier



I/O					Selected Items Write	Axis Writing
No.	Abbr.	Name	Unit	Setting range	Station1	Station2
Digital I/O						
Device setting					Setting	Setting
PD03.0-1	*	Device selection DI1		00-7F	0A	0A
PD04.0-1	*	Device selection DI2		00-7F	0B	0B
PD05.0-1	*	Device selection DI3		00-7F	22	22
PD51.0-1	*	Device selection DI3-2		00-7F	62	62
PD38.0-1	*	Device selection DI4		00-7F	2C	2C
PD39.0-1	*	Device selection DI5		00-7F	2D	2D
PD07.0-1	*	Device selection DO1		00-7F	05	05
PD08.0-1	*	Device selection DO2		00-7F	04	04
PD09.0-1	*	Device selection DO3		00-7F	03	03
Device assignment					Setting	Setting
PD01.0-7	*DIA1	Input signal automatic on selection 1		0000000-00000FF0	00000000	00000000
Input filter						
PD11.0	*	Input signal filter selection		0-B	7 : 3.500ms	7 : 3.500ms
ALM output						
PD14.1	*	Selection of output device at warning occurrence		0-1	0 : WNG signal turn (	0 : WNG signal turn (
Analog output						
Analog monitor						
PC09.0-1		Analog monitor 1 output selection		00-1F	00 : Servo motor spe	00 : Servo motor spe
PC11	MO1	Analog monitor 1 offset		-999-999	0	0
PC10.0-1		Analog monitor 2 output selection		00-1F	01 : Torque or thrust	01 : Torque or thrust
PC12	MO2	Analog monitor 2 offset		-999-999	0	0
Stroke limit function						
Stroke limit function						
PC19.0	*	[AL. 099 Stroke limit warning] selection		0-1	**1 : Disabled**	**1 : Disabled**
PD41.2	*	Limit switch enabled status selection		0-1	**1 : Only enabled ir**	**1 : Only enabled ir**
PD41.3	*	Sensor input method selection		0-1	0 : Input from servo	0 : Input from servo

(c) Changing the PDO mapping

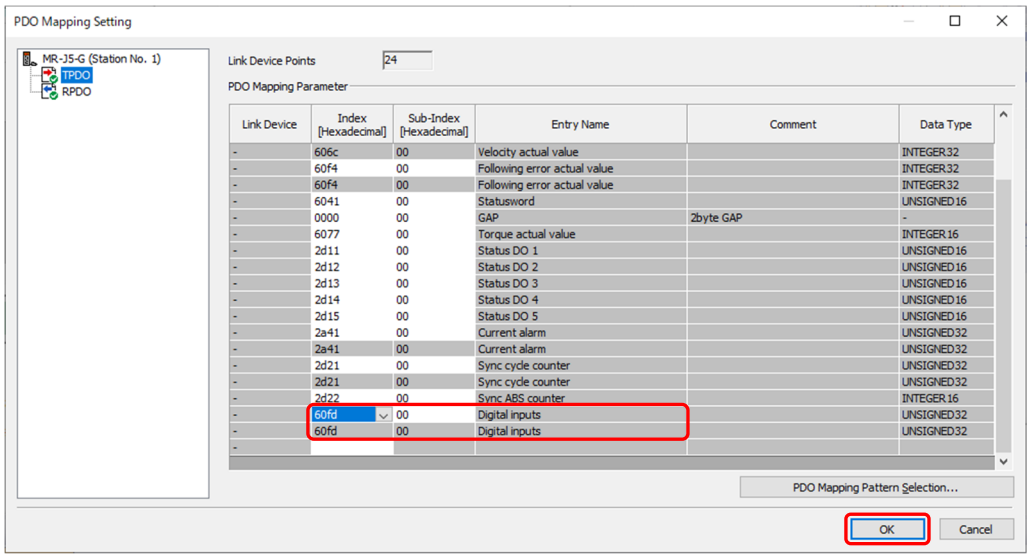
1) On the "CC-Link IE TSN Configuration" window, double-click "<Detail Setting>" in the "PDO Mapping Setting" to display the "PDO Mapping Setting" screen.



2) Set the end of the PDO mapping parameter as follows, and click the [OK] button.

Index (hexadecimal)	Sub-index (hexadecimal)
60fd	00

The entry name is displayed as "Digital inputs".
(The double word type data is displayed in two lines.)



Set as above for all axes that use hardware limit signals connected to the servo amplifier.

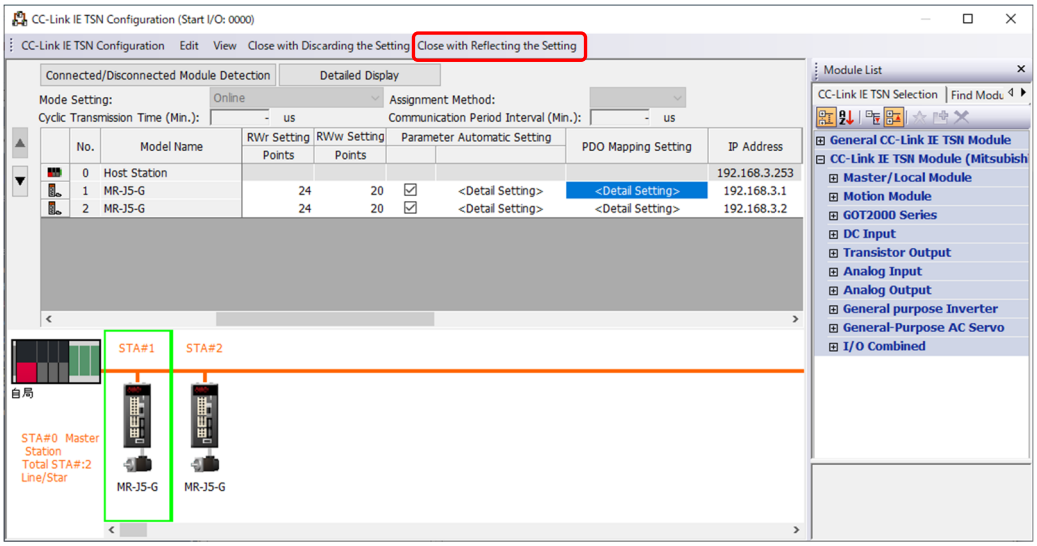
[POINTS]

"Digital inputs" is a label that contains the status of the servo amplifier input signal.

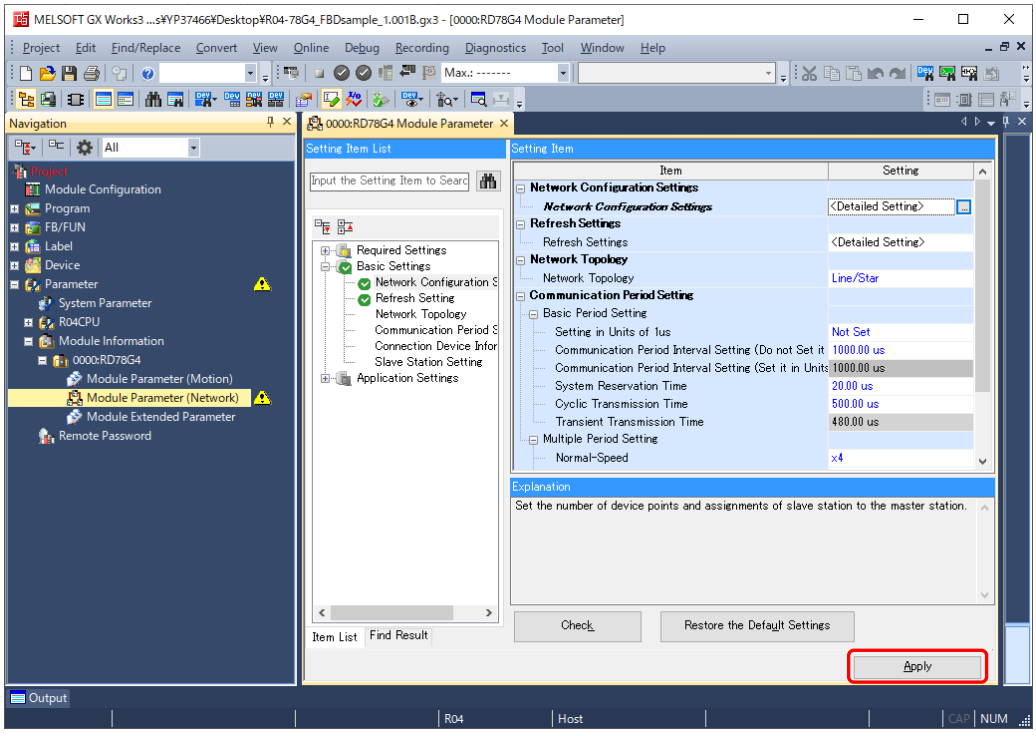
- bit0: Negative limit switch
- bit1: Positive limit switch

For details, refer to Section 17.1 in "MR-J5-G/MR-J5W-G User's Manual (Object Dictionary)".

3) After the setting is completed, click "Close with Reflecting the Setting" on the "CC-Link IE TSN Configuration" window to reflect the data.

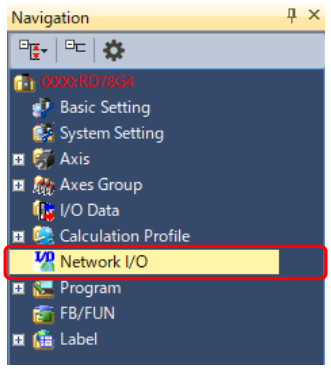


4) In the parameter editor (Module Parameter), click the [Apply] button to reflect the parameters of the Motion module.

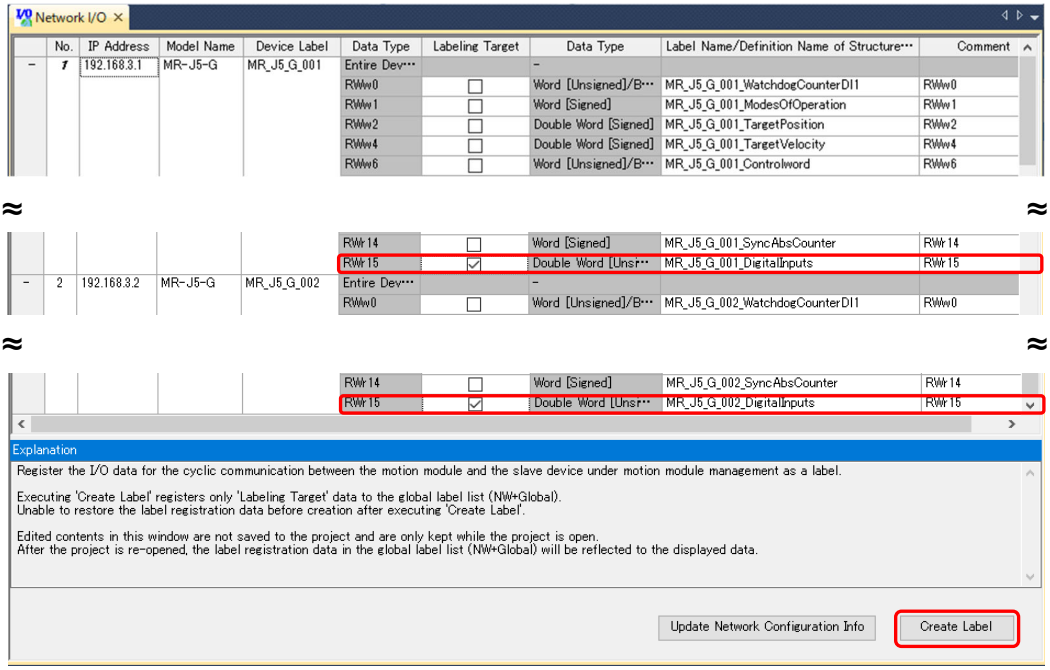


(d) Changing parameters of the Motion module

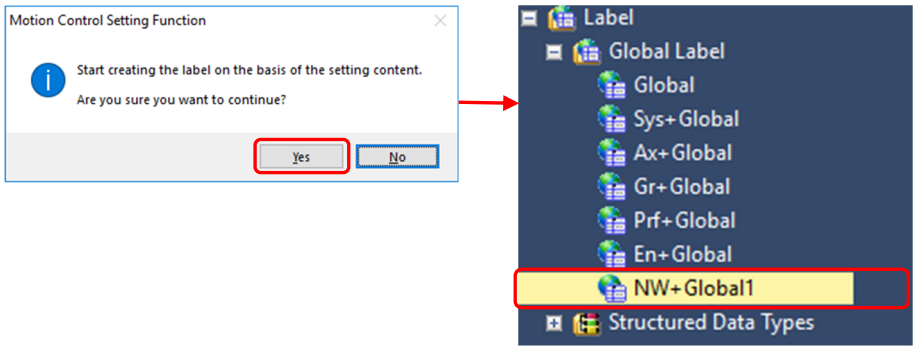
1) Double-click "Network I/O" in the navigation window of the motion control setting function to display the device controlled by the Motion module.



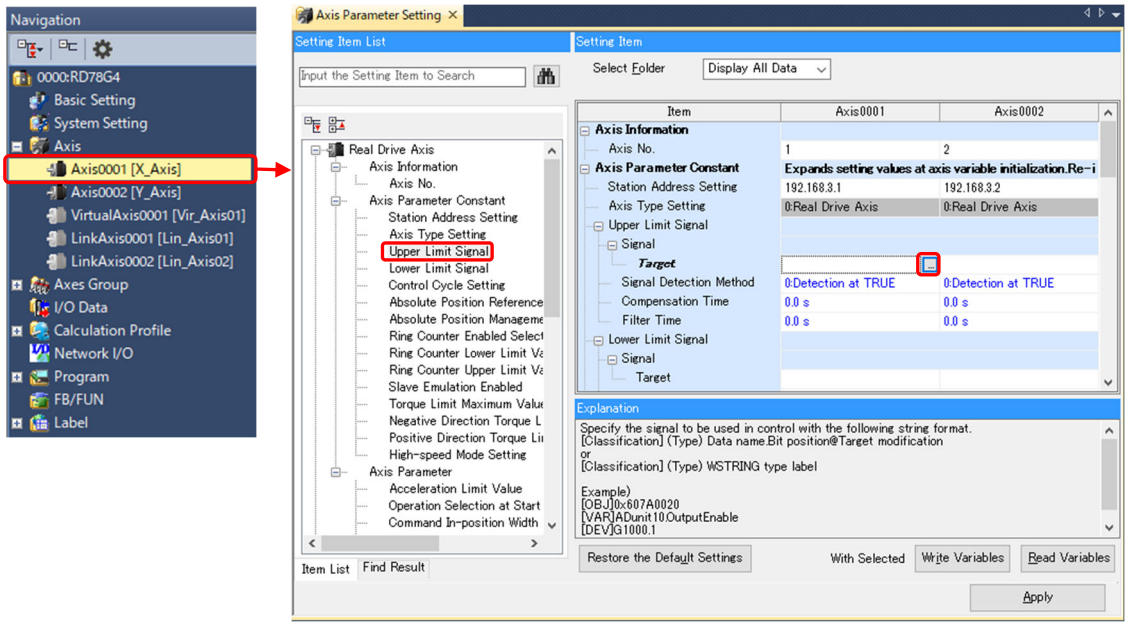
2) Check the "Labeling Target" checkbox of [MR_J5_G_***_DigitalInputs] (the label of the servo amplifier with the hardware stroke limit connected), and click the [Create Label] button.



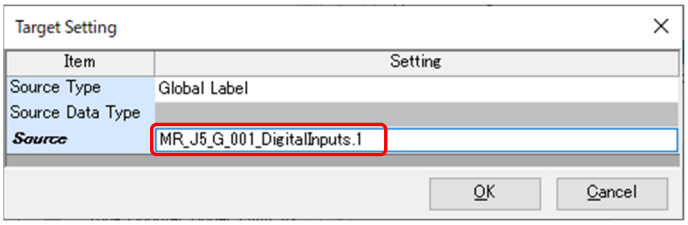
3) Click the [Yes] button on the confirmation screen.
The generated network I/O label will be stored below "NW+Global1" in "Global Label".



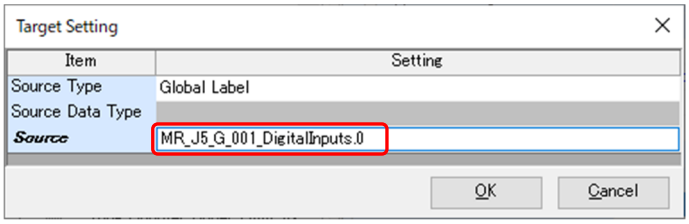
4) Open the axis parameter of the real drive axis.
From the navigation window, select "Axis", and double-click the name of the axis to be used (initial name: Axis0001).
From the "Axis Parameter Setting" window, select "Real Drive Axis" → "Upper Limit Signal", and click the [...] icon in "Target".



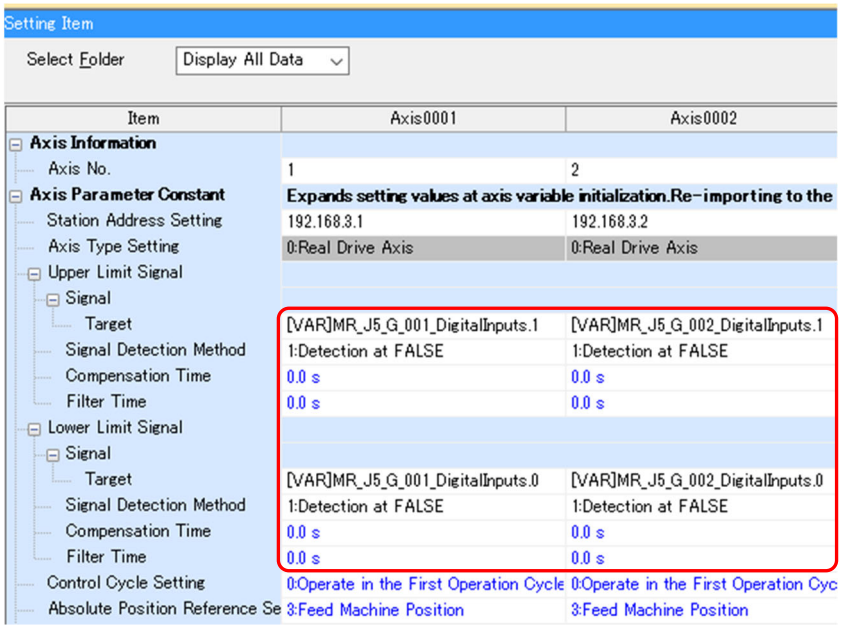
5) On the "Target Setting" screen, enter "MR_J5_G_***_DigitalInputs.1" in "Source".



On the "Target Setting" screen of the lower stroke limit, enter "MR_J5_G_***_DigitalInputs.0" in "Source".



Set as above for all axes that use hardware limit signals connected to the servo amplifier. Make sure that the relation between the axis label and the actual servo amplifier is correct. Change the signal detection method to "1: Detection at FALSE" because the normally closed contact is used.



The setting is completed. Convert all the program.

(e) Operation check

Write the program, and check whether ON/OFF of each signal can be monitored with "Upper limit signal status" and "Lower limit signal status" of the axis monitor.

To restore the value within the limit range, perform an error reset once, and then move the axis to the direction within the range with JOG operation, etc.

(2) Homing with the DOG signal

This section explains how to use the DOG signal of the servo amplifier for Homing Switch, such as using the proximity dog type homing.

(a) Servo parameters

Set the homing method and the polarity of the proximity dog signal with the following parameters.

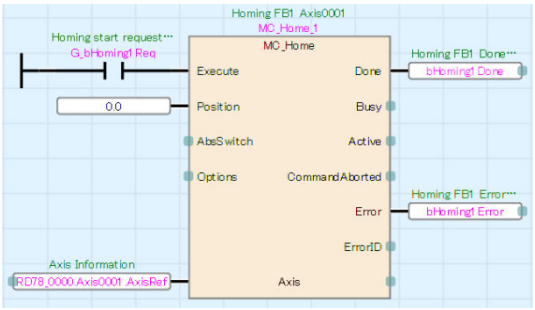
No.	Item
Pr. PT29.0	Device input polarity 1
Pr. PT45	Home position return method

Set the necessary parameters in the following table according to the homing method. (The parameters that require setting vary with the homing method.)

No.	Item
Pr. PT05	Homing speed
Pr. PT06	Creep speed
Pr. PT07	Home position shift distance
Pr. PT08	Homing position data
Pr. PT09	Travel distance after proximity dog
Pr. PT55.0	Homing deceleration time constant selection
Pr. PT56	Homing acceleration time constant
Pr. PT57	Homing deceleration time constant

(b) Program

For programs that use the DOG signal of the servo amplifier, the AbsSwitch input of MC_Home is omitted.



[FBD program]

```
//原点復帰, Homing
MC_Home_1(
    Axis      := Axis0001.AxisRef ,
    Execute   := G_bHoming1CMD ,
    Position  := 0.0 ,
    Done      => bHoming1Done,
    Error     => bHoming1Error
);
```

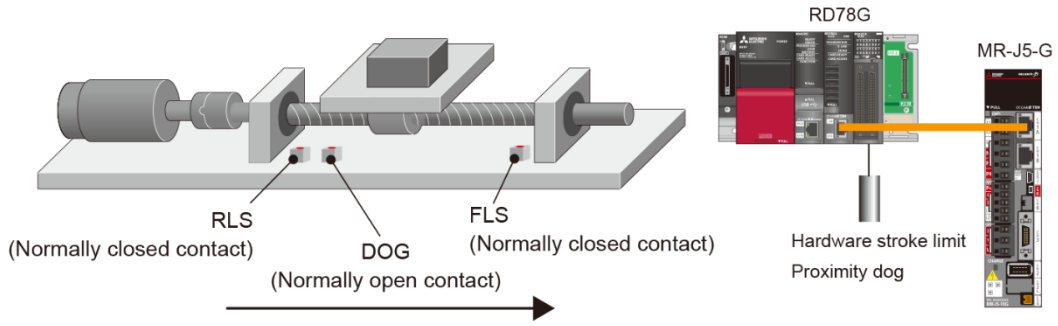
[ST program]

(c) Operation check

Write the program, and make sure that homing is correctly executed with the specified homing method.

APPENDIX 2.2 Using the input signal of the PLC CPU

This section explains the case where the PLC CPU input signal is used for the hardware stroke limit signals and the proximity dog signal.



(1) Hardware stroke limit

The negative logic (normally closed contact) is recommended for wiring of the hardware stroke limit.

[CAUTION]
If the positive logic (normally open contact) is used for the wiring of the hardware stroke limit, it may cause a serious accident at sensor failure or disconnection.

In this example, X20 is used as FLS, and X21 as RLS.

(a) The location of the sensors

Set the upper stroke limit (FLS) to the positioning address increasing side, and the lower stroke limit (RLS) to the positioning address decreasing side.

(b) Servo parameters

Set the following servo parameters as shown below:

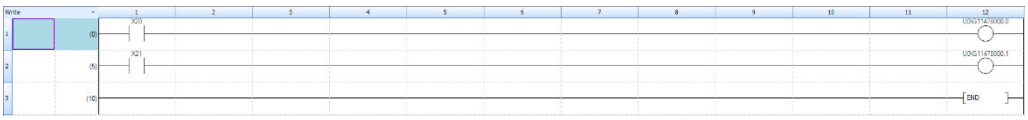
No.	Item	Setting value
Pr. PD41.2	Limit switch enabled status selection	1: Only enabled in home position return mode
Pr. PD41.3	Sensor input method selection	1: Input from controller

I/O — Selected Items Write — Axis Writing

No.	Abbr.	Name	Unit	Setting range	Station1
Digital I/O					
Device setting					Setting
PD03.0-1	*	Device selection DI1		00-7F	0A
PD04.0-1	*	Device selection DI2		00-7F	0B
PD05.0-1	*	Device selection DI3		00-7F	22
PD51.0-1	*	Device selection DI3-2		00-7F	62
PD38.0-1	*	Device selection DI4		00-7F	2C
PD39.0-1	*	Device selection DI5		00-7F	2D
PD07.0-1	*	Device selection DO1		00-7F	05
PD08.0-1	*	Device selection DO2		00-7F	04
PD09.0-1	*	Device selection DO3		00-7F	03
Device assignment					Setting
PD01.0-7	*DIA1	Input signal automatic on selection 1		0000000-00000FF0	00000000
Input filter					
PD11.0	*	Input signal filter selection		0-B	7 : 3.500ms
ALM output					
PD14.1	*	Selection of output device at warning occurrence		0-1	0 : WNG signal turn ON
Analog output					
Analog monitor					
PC09.0-1		Analog monitor 1 output selection		00-1F	00 : Servo motor speed (
PC11	MO1	Analog monitor 1 offset		-999-999	0
PC10.0-1		Analog monitor 2 output selection		00-1F	01 : Torque or thrust (±8
PC12	MO2	Analog monitor 2 offset		-999-999	0
Stroke limit function					
Stroke limit function					
PC19.0	*	[AL. 099 Stroke limit warning] selection		0-1	0 : Enabled
PD41.2	*	Limit switch enabled status selection		0-1	**1 : Only enabled in ho**
PD41.3	*	Sensor input method selection		0-1	**1 : Input from control**

(c) Program of the PLC CPU

The ON/OFF status of X20 and X21 is copied to the public label or the buffer memory and transmitted to the Motion module. The following shows a program when the ON/OFF status is copied to the buffer memory.



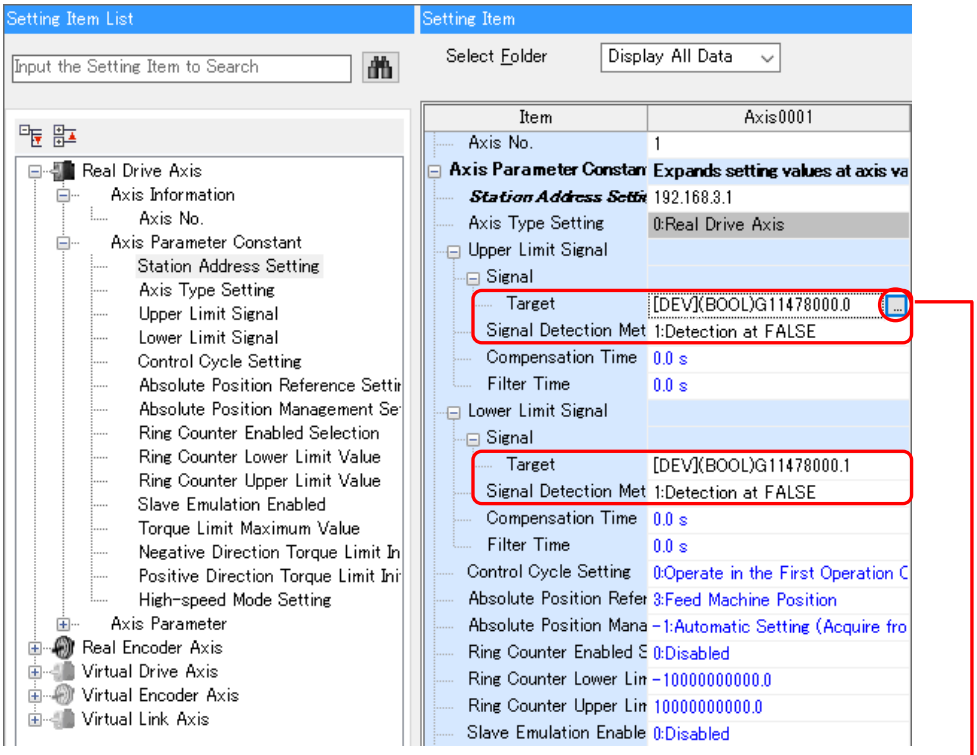
X20→U□\G11478000.0
X21→U□\G11478000.1

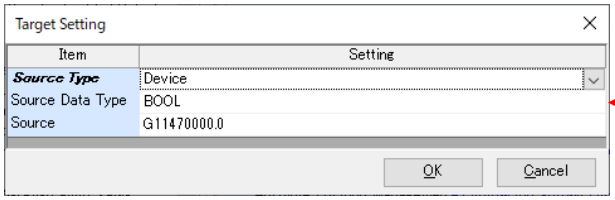
[POINTS]

The user setting area of the buffer memory is 11478000 to 11997999.

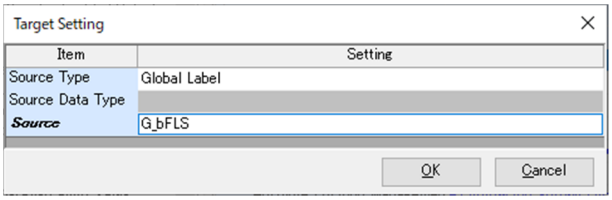
(d) Parameters of the Motion module

On the "Axis Parameter Setting" window, set "Upper Limit Signal" and "Lower Limit Signal" as follows:





(Note) When the public label is used, set as follows:



Set as above for all axes that use hardware limit signals.

Change the signal detection method to "1: Detection at FALSE" because the normally closed contact is used.

(e) Operation check

Write the program, and check whether ON/OFF of each signal can be monitored with "Upper limit signal status" and "Lower limit signal status" of the axis monitor.

To restore the value within the limit range, perform an error reset once, and then move the axis to the direction within the range with JOG operation, etc.

(2) Homing with the DOG signal

In this example, X22 is used as a DOG signal.

(a) Servo parameters

Set the homing method and the polarity of the proximity dog signal with the following parameters.

No.	Item
Pr. PT29.0	Device input polarity 1
Pr. PT45	Home position return method

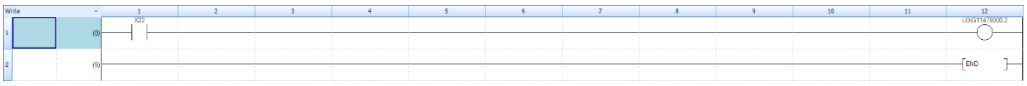
Set the necessary parameters in the following table according to the homing method. (The parameters that require setting vary with the homing method.)

No.	Item
Pr. PT05	Home position return speed
Pr. PT06	Creep speed
Pr. PT07	Home position shift distance
Pr. PT08	Home position return position data
Pr. PT09	Moving distance after proximity dog
Pr. PT55.0	Home position return deceleration time constant selection
Pr. PT56	Home position return acceleration time constant
Pr. PT57	Home position return deceleration time constant

(b) Programming with the Motion module

1) Program of the PLC CPU

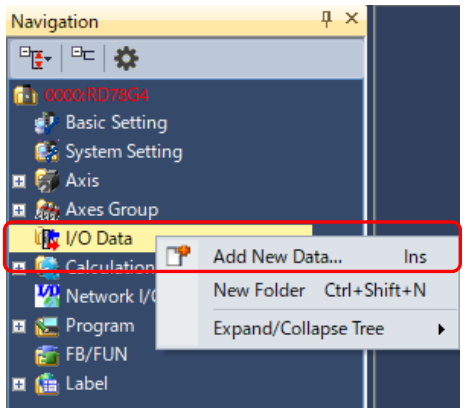
The ON/OFF status of X22 is copied to the public label or the buffer memory and transmitted to the Motion module. The following shows a program when the ON/OFF status is copied to the buffer memory.



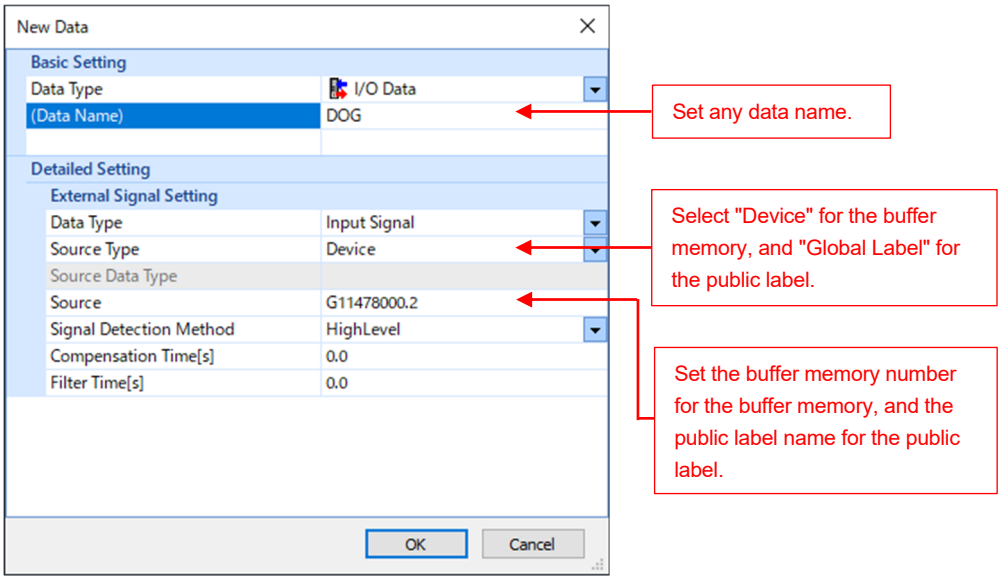
X22→U□\G11478000.2

2) Parameters of the Motion module

Right-click "I/O Data" in the navigation window of the motion control setting function, and select [Add New Data].



Set as follows:



3) Program of the Motion module

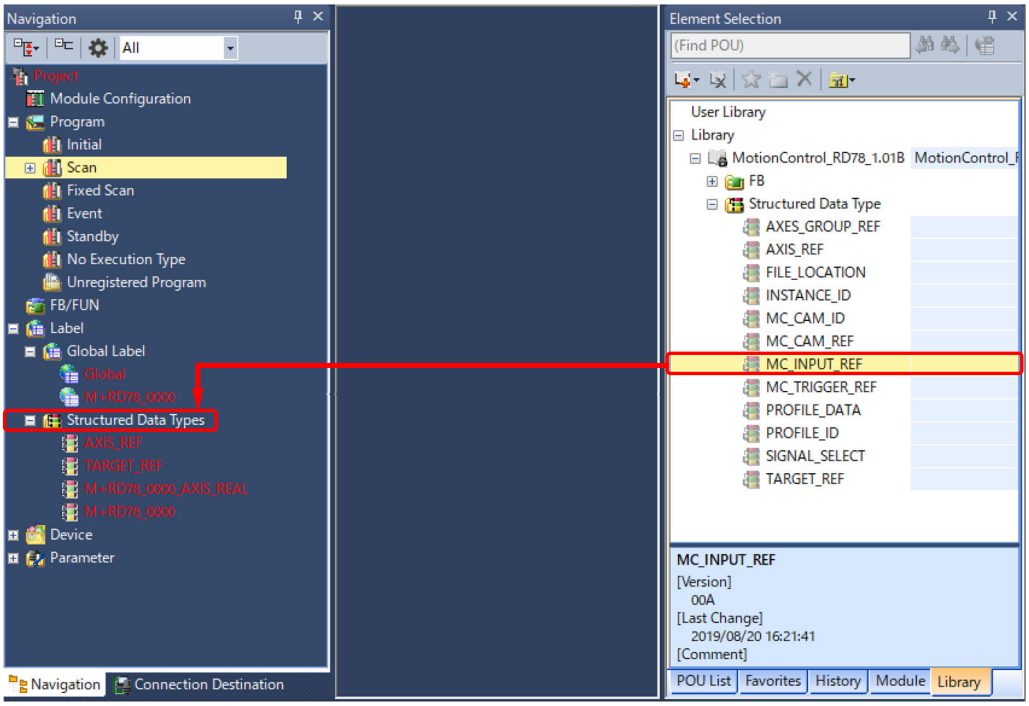
Enter the I/O data name set in 2) in the AbsSwitch input of MC_Home.

```
//Homing
MC_Home_1(
    Axis:= Axis0001.AxisRef,
    Execute:= bHoming ,
    Position:= 0.0 ,
    AbsSwitch:= DOG ,
    Options:= 0 ,
    Done=> bHomeDone ,
    Error=> bHomeError
);
```

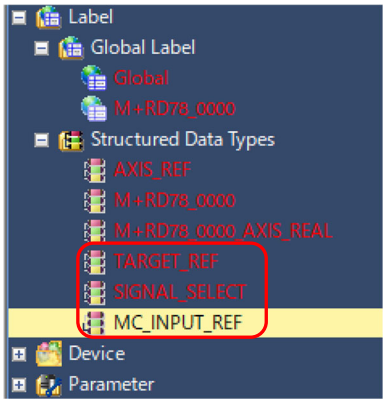
Enter the I/O data name.

(c) Programming with the PLC CPU

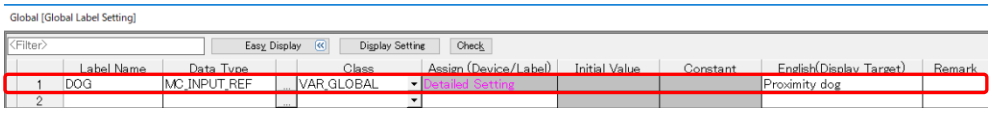
1) It is necessary to define an MC_INPUT_REF type structure to the PLC CPU.
From the [Library] tab in the element selection window on MELSOFT GX Works3, click "Library" → "MotionControl_RD78_**** → "Structured Data Type".
Select "MC_INPUT_REF", and drag and drop it onto "Structured Data Types" under "Label" in the navigation window.



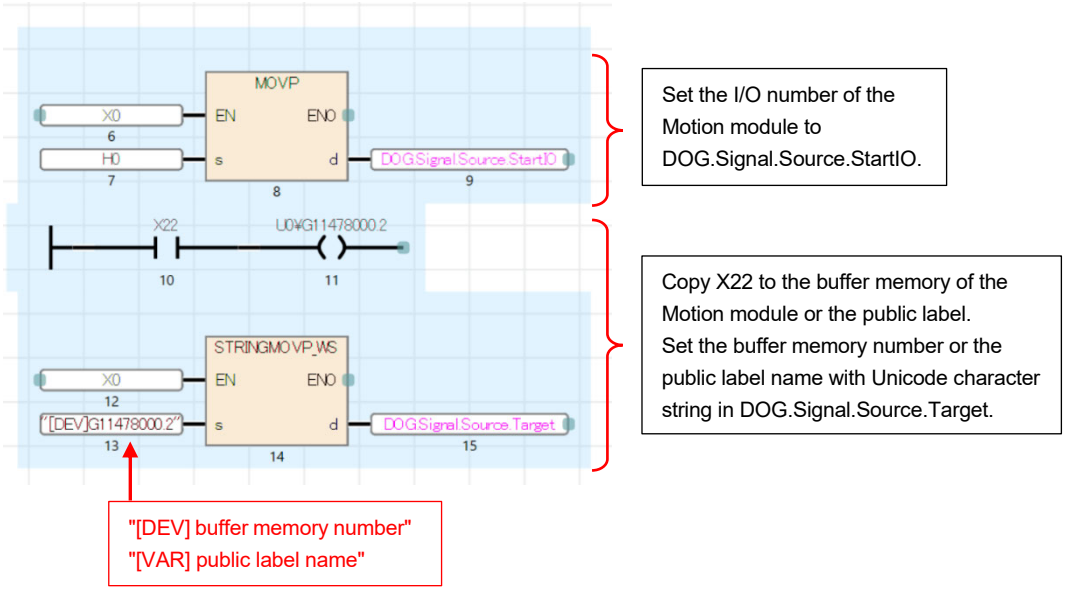
2) "MC_INPUT_REF" and the structures of the member ("TARGET_REF" and "SIGNAL_SELECT") are registered.



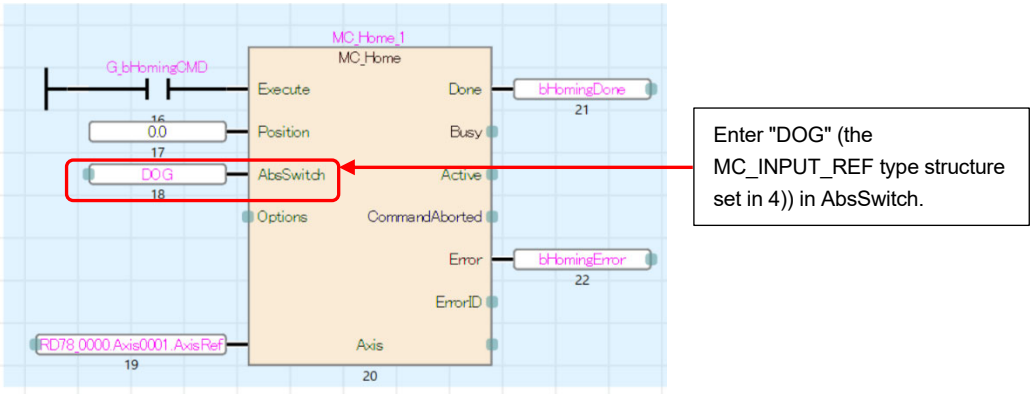
3) Prepare an MC_INPUT_REF type structure in the label of the PLC CPU.
The label can be registered either as a global or public label.
The label name is "DOG" in this example.



4) Set the member of the MC_INPUT_REF type structure with the program.



5) Write MC_Home to the program.

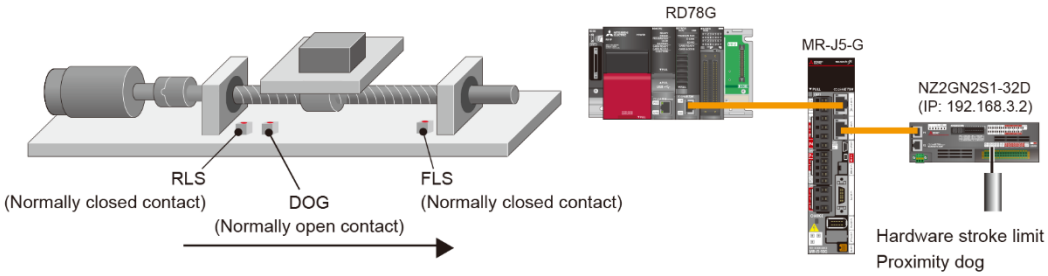


(d) Operation check

Write the program, and make sure that homing is correctly executed with the specified homing method.

APPENDIX 2.3 Using the input signal of the remote input module

This section explains the case where the remote input module of the motion control station is used for the hardware stroke limit signal and the proximity dog signal.



(1) Hardware stroke limit

The negative logic (normally closed contact) is recommended for the wiring of the hardware stroke limit.

[CAUTION]

If the positive logic (normally open contact) is used for the wiring of the hardware stroke limit, it may cause a serious accident at sensor failure or disconnection.

In this example, RX0 is used as FLS, and RX1 as RLS.

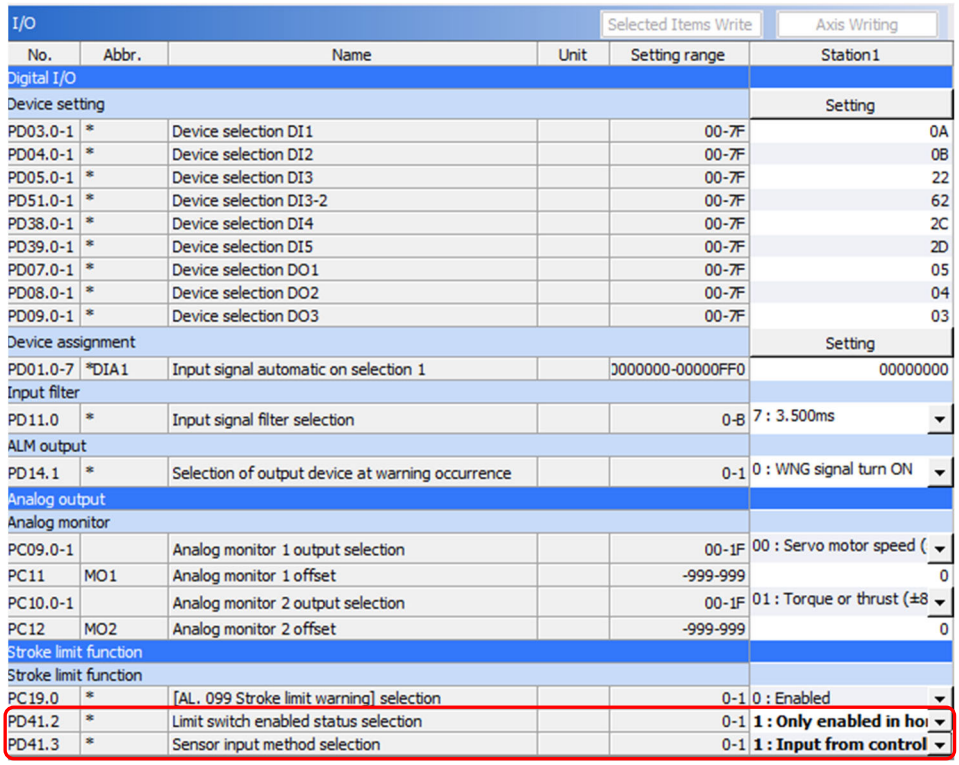
(a) The location of the sensors

Set the upper stroke limit (FLS) to the positioning address increasing side, and the lower stroke limit (RLS) to the positioning address decreasing side.

(b) Servo parameters

Set the following servo parameters as shown below:

No.	Item	Setting value
Pr. PD41.2	Limit switch enabled status selection	1: Only enabled in home position return mode
Pr. PD41.3	Sensor input method selection	1: Input from controller

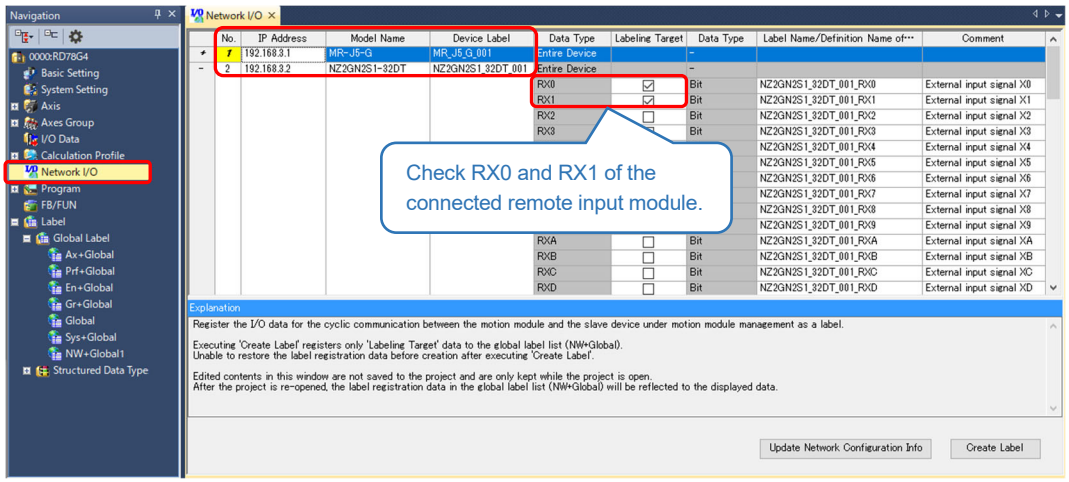


I/O				Selected Items Write	Axis Writing
No.	Abbr.	Name	Unit	Setting range	Station1
Digital I/O					
Device setting					Setting
PD03.0-1	*	Device selection DI1		00-7F	0A
PD04.0-1	*	Device selection DI2		00-7F	0B
PD05.0-1	*	Device selection DI3		00-7F	22
PD51.0-1	*	Device selection DI3-2		00-7F	62
PD38.0-1	*	Device selection DI4		00-7F	2C
PD39.0-1	*	Device selection DI5		00-7F	2D
PD07.0-1	*	Device selection DO1		00-7F	05
PD08.0-1	*	Device selection DO2		00-7F	04
PD09.0-1	*	Device selection DO3		00-7F	03
Device assignment					Setting
PD01.0-7	*DIA1	Input signal automatic on selection 1		0000000-00000FF0	00000000
Input filter					
PD11.0	*	Input signal filter selection		0-B	7 : 3.500ms
ALM output					
PD14.1	*	Selection of output device at warning occurrence		0-1	0 : WNG signal turn ON
Analog output					
Analog monitor					
PC09.0-1		Analog monitor 1 output selection		00-1F	00 : Servo motor speed (
PC11	MO1	Analog monitor 1 offset		-999-999	0
PC10.0-1		Analog monitor 2 output selection		00-1F	01 : Torque or thrust (±8
PC12	MO2	Analog monitor 2 offset		-999-999	0
Stroke limit function					
Stroke limit function					
PC19.0	*	[AL. 099 Stroke limit warning] selection		0-1	0 : Enabled
PD41.2	*	Limit switch enabled status selection		0-1	**1 : Only enabled in ho**
PD41.3	*	Sensor input method selection		0-1	**1 : Input from control**

(c) Parameters of the Motion module

1) Setting of the network I/O

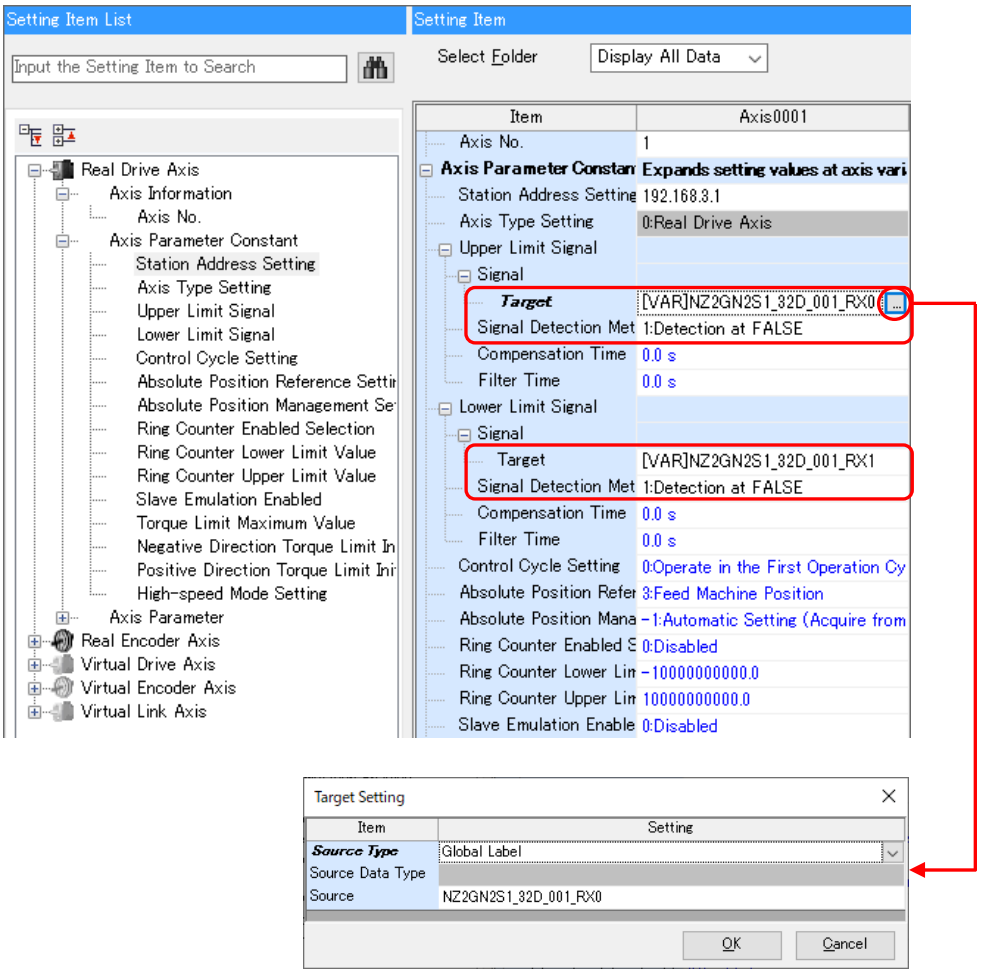
Double-click "Network I/O" in the navigation window of the motion control setting function. On this screen, label the input signal of the remote input module. Check the checkbox of RX0 and RX1, and click the [Create Label] button.



After the label is generated, the "NW+Global1" group will be added in "Global Label" in the navigation window.

2) Axis parameters

On the "Axis Parameter Setting" window, set "Target" of "Upper Limit Signal" and "Lower Limit Signal" as follows:



Set as above for all axes that use hardware limit signals.

Change the signal detection method to "1: Detection at FALSE" because the normally closed contact is used.

(d) Operation check

Write the program, and check whether ON/OFF of each signal can be monitored with "Upper limit signal status" and "Lower limit signal status" of the axis monitor.

To restore the value within the limit range, perform an error reset once, and then move the axis to the direction within the range with JOG operation, etc.

(2) Homing with the DOG signal

In this example, RX2 is used as a DOG signal.

(a) Servo parameters

Set the homing method and the polarity of the proximity dog signal with the following parameters.

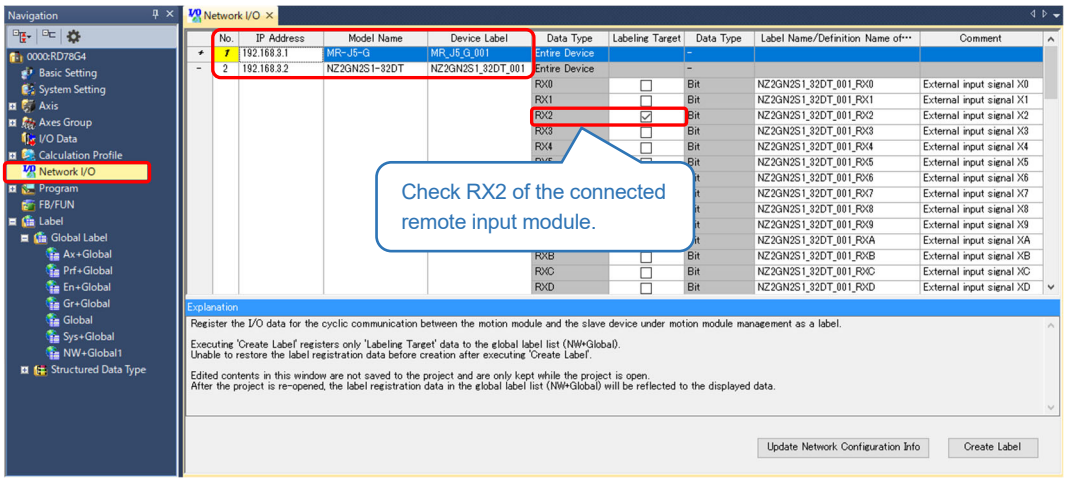
No.	Item
Pr. PT29.0	Device input polarity 1
Pr. PT45	Home position return method

Set the necessary parameters in the following table according to the homing method. (The parameters that require setting vary with the homing method.)

No.	Item
Pr. PT05	Home position return speed
Pr. PT06	Creep speed
Pr. PT07	Home position shift distance
Pr. PT08	Home position return position data
Pr. PT09	Moving distance after proximity dog
Pr. PT55.0	Home position return deceleration time constant selection
Pr. PT56	Home position return acceleration time constant
Pr. PT57	Home position return deceleration time constant

(b) Parameters of the Motion module (setting of the network I/O)

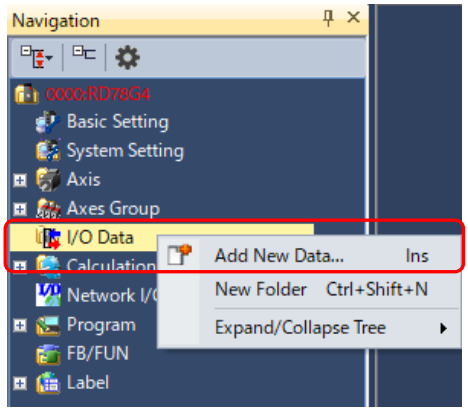
Double-click "Network I/O" in the navigation window of the motion control setting function. On this screen, label the input signal of the remote input module. Check the checkbox of RX2, and click the [Create Label] button.



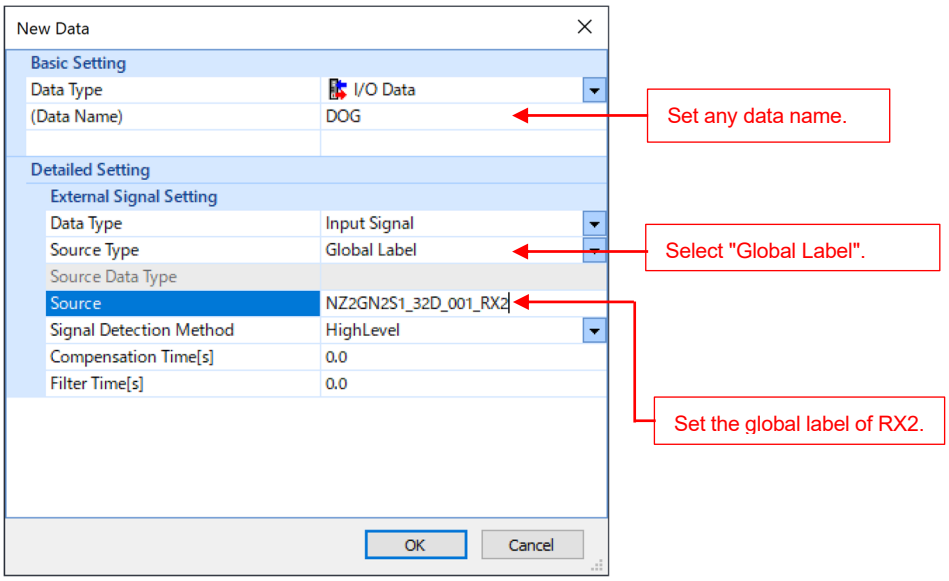
After the label is generated, the "NW+Global1" group will be added in "Global Label" in the navigation window.

(c) Programming with the Motion module

1) Right-click "I/O Data" in the navigation window of the motion control setting function, and select [Add New Data].



Set as follows:



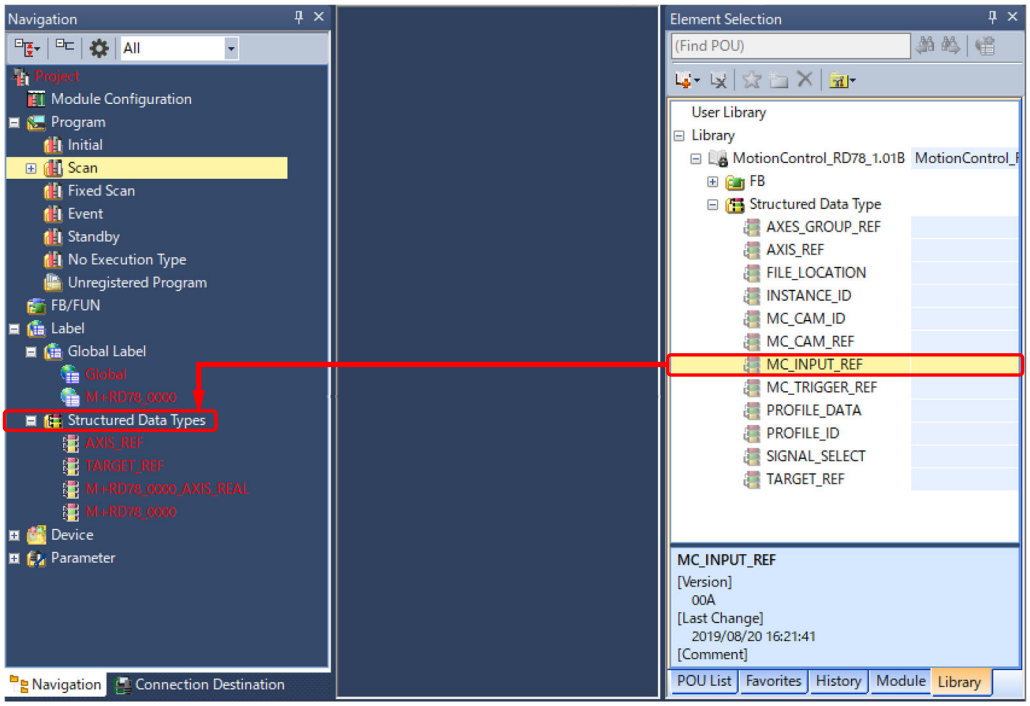
2) Enter the I/O data name set in (2) in the AbsSwitch input of MC_Home.

```
//Homing
MC_Home_1(
    Axis:= Axis0001.AxisRef,
    Execute:= bHoming ,
    Position:= 0.0 ,
    AbsSwitch:= DOG ,
    Options:= 0 ,
    Done=> bHomeDone ,
    Error=> bHomeError
);
```

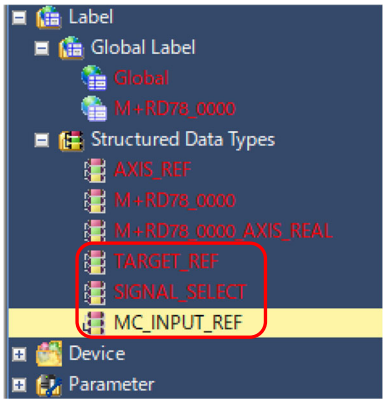
Enter the I/O data name.

(d) Programming with the PLC CPU

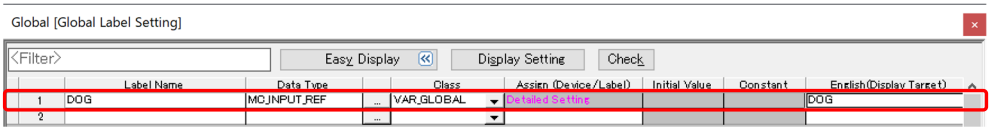
1) It is necessary to define an MC_INPUT_REF type structure to the PLC CPU.
From the [Library] tab in the element selection window on MELSOFT GX Works3, click "Library" → "MotionControl_RD78_****" → "Structured Data Type".
Select "MC_INPUT_REF", and drag and drop it onto "Structured Data Types" under "Label" in the navigation window.



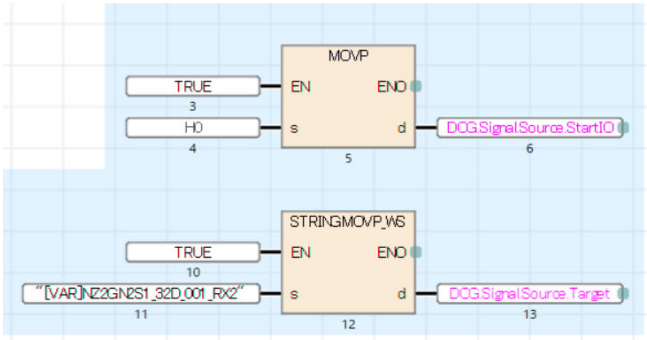
2) "MC_INPUT_REF" and the structures of the member ("TARGET_REF" and "SIGNAL_SELECT") are registered.



3) Prepare an MC_INPUT_REF type structure in the label of the PLC CPU.
The label can be registered in either the global or public label.
The label name is "DOG" in this example.



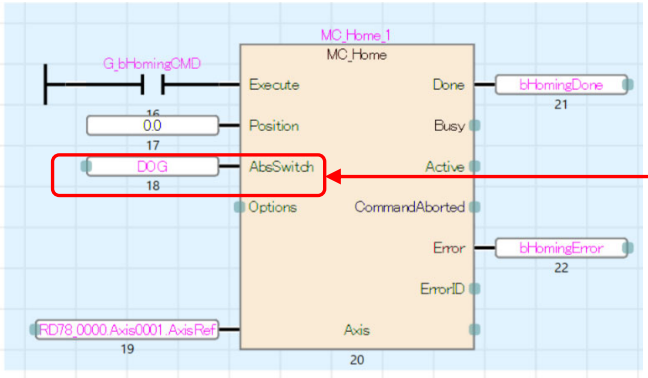
4) Set the member of the MC_INPUT_REF type structure with the program.



Set the I/O number of the Motion module to DOG.Signal.Source.StartIO.

Set the label name of RX2 with Unicode character string in DOG.Signal.Source.Target.

5) Write MC_Home to the program.



Enter "DOG" (the MC_INPUT_REF type structure set in 4)) in AbsSwitch.

(e) Operation check

Write the program, and make sure that homing is correctly executed with the specified homing method.

APPENDIX 3 Precautions When the Absolute Position Detection is Used

When the absolute position detection system is used, be careful of the following items.

(1) Setting of the servo parameters

Change the following two servo parameters:

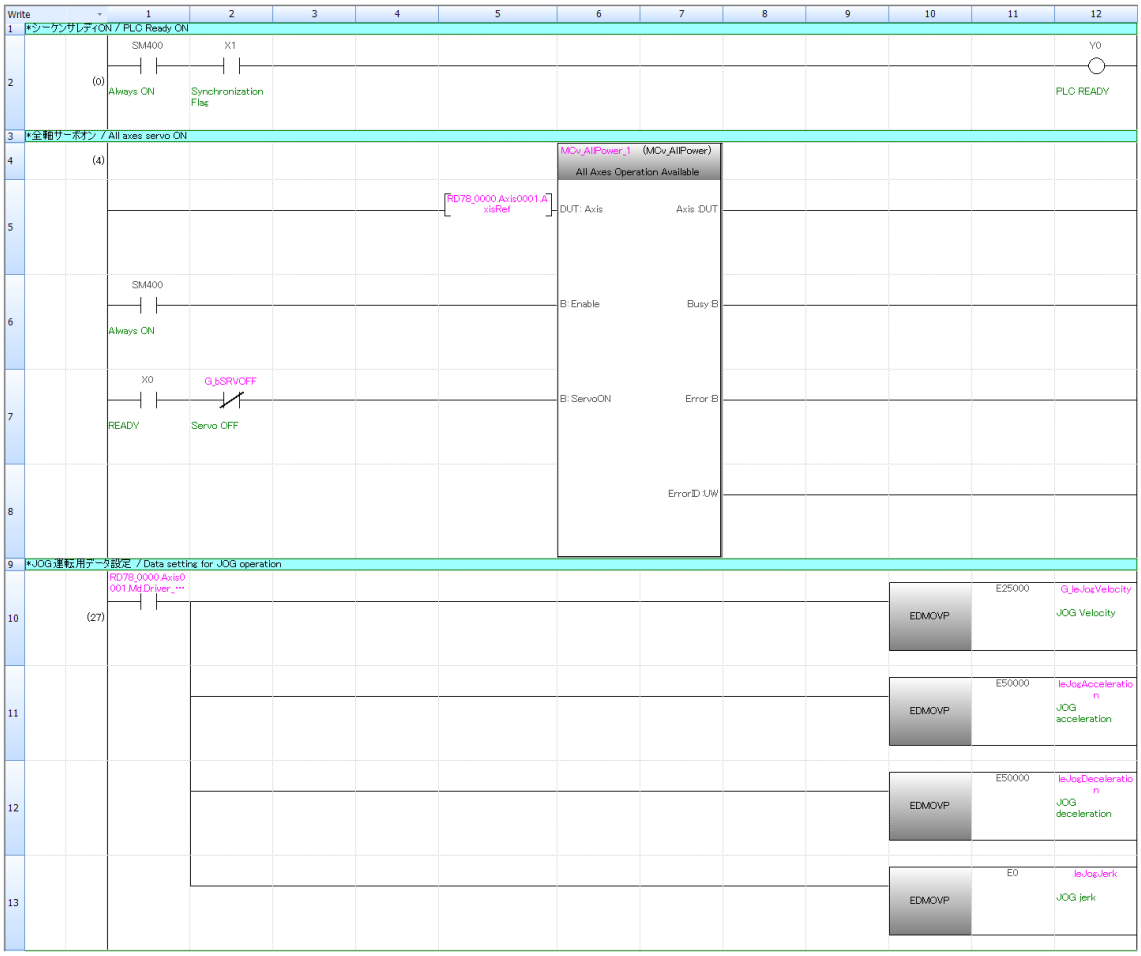
No.	Item	Setting value
Pr. PA03.0	Absolute position detection system selection	0: Disabled → 1: Enabled
Pr. PC29.5	[AL. 0E3 Absolute position counter warning] selection	1: Enabled → 0: Disabled

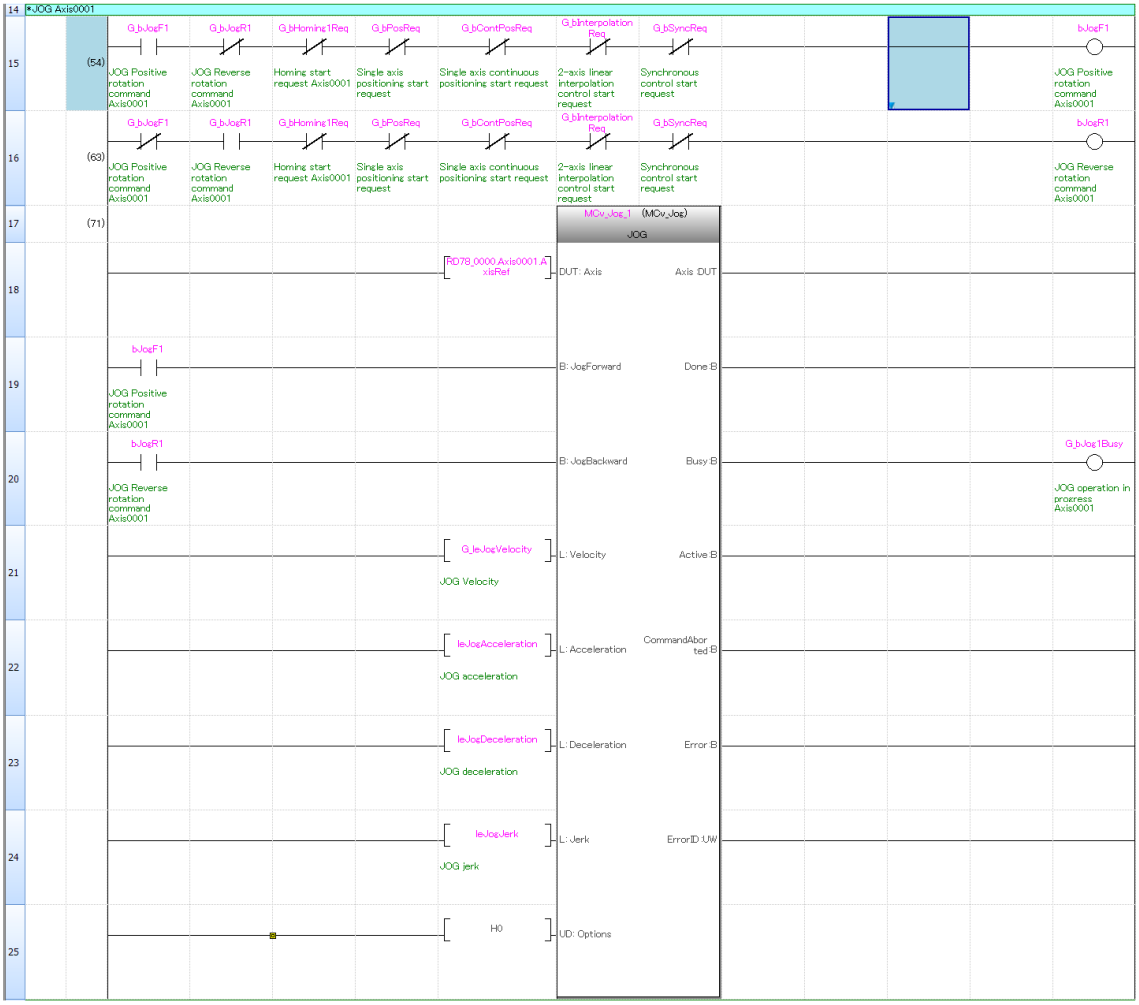
(2) At the first power-on

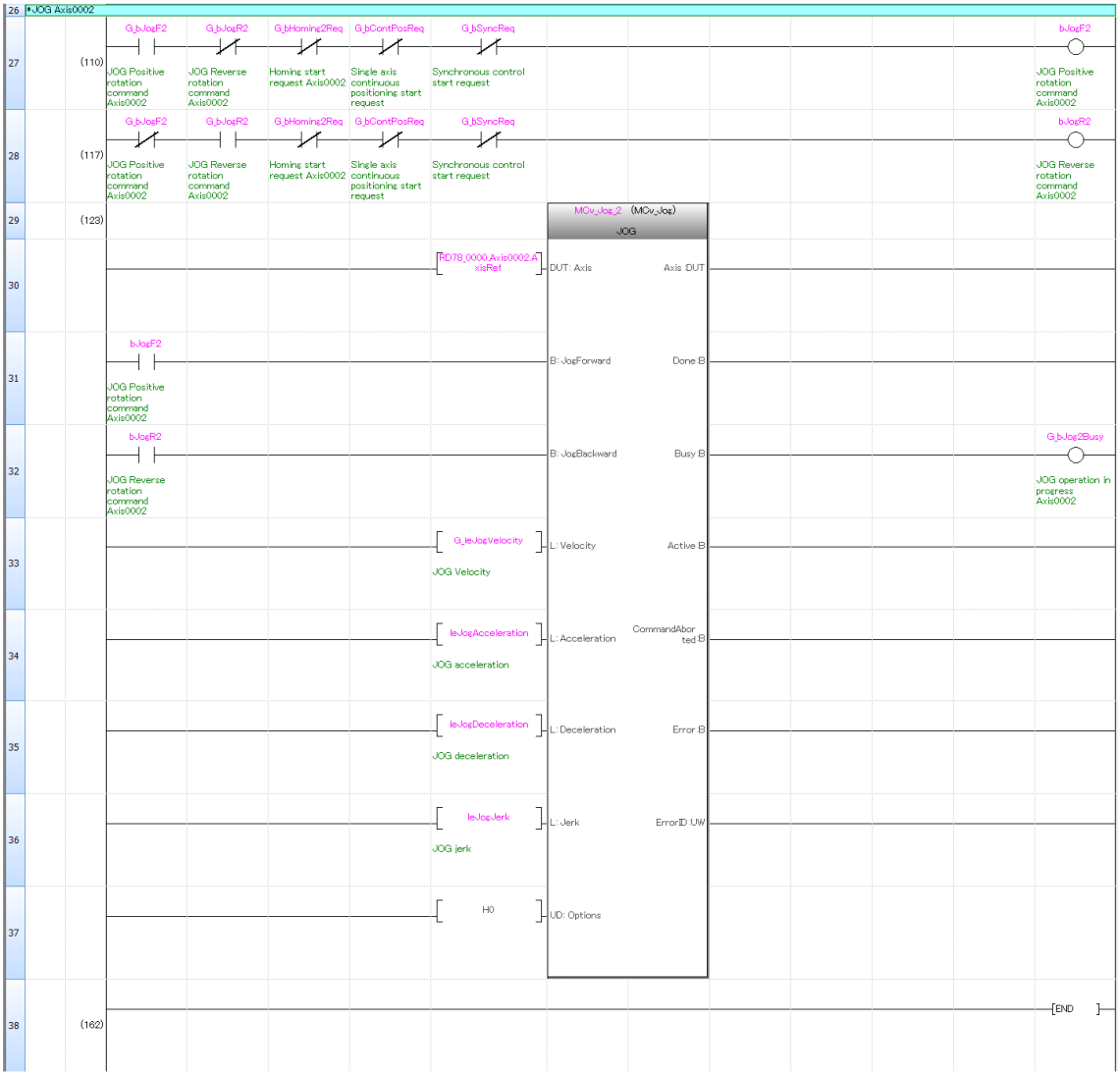
After the absolute position detection system is enabled, [AL. 025.1 Servo motor encoder – Absolute position erased] occurs at the first power-on of the servo amplifier.
Wait for 5 s as the alarm occurs, and then cycle the power.
After that, execute homing.

APPENDIX 4 Program Example in Ladder

The following shows a [ServoON_Jog] program example created in ladder instead of FBD in Chapter 4.




*JOG Axis0001
G_bJogF1
G_bJogR1
G_bHoming1Req
G_bPosReq
G_bContPosReq
G_bInterpolation Req
G_bSyncReq
bJogF1
(54)
JOG Positive rotation command Axis0001
JOG Reverse rotation command Axis0001
Homing start request Axis0001
Single axis positioning start request
Single axis continuous positioning start request
2-axis linear interpolation control start request
Synchronous control start request
JOG Positive rotation command Axis0001
bJogR1
(63)
JOG Reverse rotation command Axis0001
(71)
MCv_Jog_1
(MCv_Jog)
JOG
RD78_0000.Axis0001.AxisRef
DUT: Axis
Axis: DUT
B: JogForward
Done: B
B: JogBackward
Busy: B
G_bJog1Busy
JOG operation in progress Axis0001
G_leJogVelocity
L: Velocity
Active: B
JOG Velocity
leJogAcceleration
L: Acceleration
CommandAborted: B
JOG acceleration
leJogDeceleration
L: Deceleration
Error: B
JOG deceleration
leJogJerk
L: Jerk
ErrorID: UW
JOG jerk
H0
UD: Options


*JOG Axis0002
G_bJogF2
G_bJogR2
G_bHoming2Req
G_bContPosReq
G_bSyncReq
bJogF2
(110)
JOG Positive rotation command Axis0002
JOG Reverse rotation command Axis0002
Homing start request Axis0002
Single axis continuous positioning start request
Synchronous control start request
JOG Positive rotation command Axis0002
bJogR2
(117)
JOG Reverse rotation command Axis0002
(123)
MCv_Jog_2 (MCv_Jog)
JOG
RD78_0000.Axis0002.AxisRef
DUT: Axis
Axis :DUT
B: JogForward
Done :B
B: JogBackward
Busy :B
G_bJog2Busy
JOG operation in progress Axis0002
G_leJogVelocity
L: Velocity
Active :B
JOG Velocity
leJogAcceleration
L: Acceleration
CommandAborted :B
JOG acceleration
leJogDeceleration
L: Deceleration
Error :B
JOG deceleration
leJogJerk
L: Jerk
ErrorID :UW
JOG jerk
H0
UD: Options
(162)
END

MEMO

Mistubishi Electric Servo System Motion Module Quick Start Guide

Country/Region	Sales office	
USA	Mitsubishi Electric Automation, Inc. 500 Corporate Woods Parkway, Vernon Hills, IL 60061, U.S.A.	Tel : +1-847-478-2100
Mexico	Mitsubishi Electric Automation, Inc. Mexico Branch Boulevard Miguel de Cervantes Saavedra 301, Torre Norte Piso 5, Ampliacion Granada, Miguel Hidalgo, Ciudad de Mexico, Mexico, C.P.11520	Tel : +52-55-3067-7512
Brazil	Mitsubishi Electric do Brasil Comercio e Servicos Ltda. Avenida Adelino Cardana, 293, 21 andar, Bethaville, Barueri SP, Brazil	Tel : +55-11-4689-3000
Germany	Mitsubishi Electric Europe B.V. German Branch Mitsubishi-Electric-Platz 1, 40882 Ratingen, Germany	Tel : +49-2102-486-0
UK	Mitsubishi Electric Europe B.V. UK Branch Travellers Lane, UK-Hatfield, Hertfordshire, AL10 8XB, U.K.	Tel : +44-1707-28-8780
Italy	Mitsubishi Electric Europe B.V. Italian Branch Centro Direzionale Colleoni - Palazzo Sirio, Viale Colleoni 7, 20864 Agrate Brianza (MB), Italy	Tel : +39-039-60531
Spain	Mitsubishi Electric Europe B.V. Spanish Branch Carretera de Rubi, 76-80-Apdo. 420, E-08190 Sant Cugat del Valles (Barcelona), Spain	Tel : +34-935-65-3131
France	Mitsubishi Electric Europe B.V. French Branch 25, Boulevard des Bouvets, 92741 Nanterre Cedex, France	Tel : +33-1-55-68-55-68
Czech Republic	Mitsubishi Electric Europe B.V. Czech Branch, Prague Office Pekarska 621/7, 155 00 Praha 5, Czech Republic	Tel : +420-255-719-200
Poland	Mitsubishi Electric Europe B.V. Polish Branch ul. Krakowska 48, 32-083 Balice, Poland	Tel : +48-12-347-65-00
Russia	Mitsubishi Electric (Russia) LLC St. Petersburg Branch Piskarevsky pr. 2, bld 2, lit "Sch", BC "Benua", office 720; 195027 St. Petersburg, Russia	Tel : +7-812-633-3497
Sweden	Mitsubishi Electric Europe B.V. (Scandinavia) Hedvig Mollersgata 6, 223 55 Lund, Sweden	Tel : +46-8-625-10-00
Turkey	Mitsubishi Electric Turkey A.S. Umraniye Branch Serifali Mahallesi Nutuk Sokak No:5, TR-34775 Umraniye / Istanbul, Turkey	Tel : +90-216-526-3990
UAE	Mitsubishi Electric Europe B.V. Dubai Branch Dubai Silicon Oasis, P.O.BOX 341241, Dubai, U.A.E.	Tel : +971-4-3724716
South Africa	Adroit Technologies 20 Waterford Office Park, 189 Witkoppen Road, Fourways, South Africa	Tel : +27-11-658-8100
China	Mitsubishi Electric Automation (China) Ltd. Mitsubishi Electric Automation Center, No.1386 Hongqiao Road, Shanghai, China	Tel : +86-21-2322-3030
Taiwan	SETSUYO ENTERPRISE CO., LTD. 6F, No.105, Wugong 3rd Road, Wugu District, New Taipei City 24889, Taiwan	Tel : +886-2-2299-2499
Korea	Mitsubishi Electric Automation Korea Co., Ltd. 7F to 9F, Gangseo Hangang Xi-tower A, 401, Yangcheon-ro, Gangseo-Gu, Seoul 07528, Korea	Tel : +82-2-3660-9529
Singapore	Mitsubishi Electric Asia Pte. Ltd. 307 Alexandra Road, Mitsubishi Electric Building, Singapore 159943	Tel : +65-6473-2308
Thailand	Mitsubishi Electric Factory Automation (Thailand) Co., Ltd. 101, True Digital Park Office, 5th Floor, Sukhumvit Road, Bangchak, Phra Khanong, Bangkok	Tel : +66-2092-8600
Indonesia	PT. Mitsubishi Electric Indonesia Gedung Jaya 8th Floor, JL. MH. Thamrin No.12, Jakarta Pusat 10340, Indonesia	Tel : +62-21-3192-6461
Vietnam	Mitsubishi Electric Vietnam Company Limited 11th & 12th Floor, Viettel Tower B, 285 Cach Mang Thang 8 Street, Ward 12, District 10, Ho Chi Minh City, Vietnam	Tel : +84-28-3910-5945
India	Mitsubishi Electric India Pvt. Ltd. Pune Branch Emerald House, EL-3, J Block, M.I.D.C., Bhosari, Pune - 411026, Maharashtra, India	Tel : +91-20-2710-2000
Australia	Mitsubishi Electric Australia Pty. Ltd. 348 Victoria Road, P.O. Box 11, Rydalmere, N.S.W 2116, Australia	Tel : +61-2-9684-7777

Mitsubishi Electric Corporation Nagoya Works is a factory certified for ISO14001 (standards for environmental management systems) and ISO9001(standards for quality assurance management systems)

ISO 14001 JACO JAPAN ACCREDITATION ORGANIZATION FOR ENVIRONMENT AND QUALITY
EC 98 J 2017
UKAS MANAGEMENT SYSTEMS 0051

ISO 9001
BUREAU VERITAS Certification
BUREAU VERITAS 1828
UKAS MANAGEMENT SYSTEMS 0008

MITSUBISHI ELECTRIC CORPORATION

HEAD OFFICE: TOKYO BUILDING, 2-7-3, MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN
NAGOYA WORKS: 1-14, YADA-MINAMI 5, HIGASHI-KU, NAGOYA, JAPAN

L(NA)03191ENG-A 2010 (IP)

New publication, effective October 2020