Programmable Automation Controller

MELSEC MX Controller Basics (Motion Control Startup)

This course is for participants who will establish a motion control system using the MELSEC MX Controller MX-F Model for the first time.

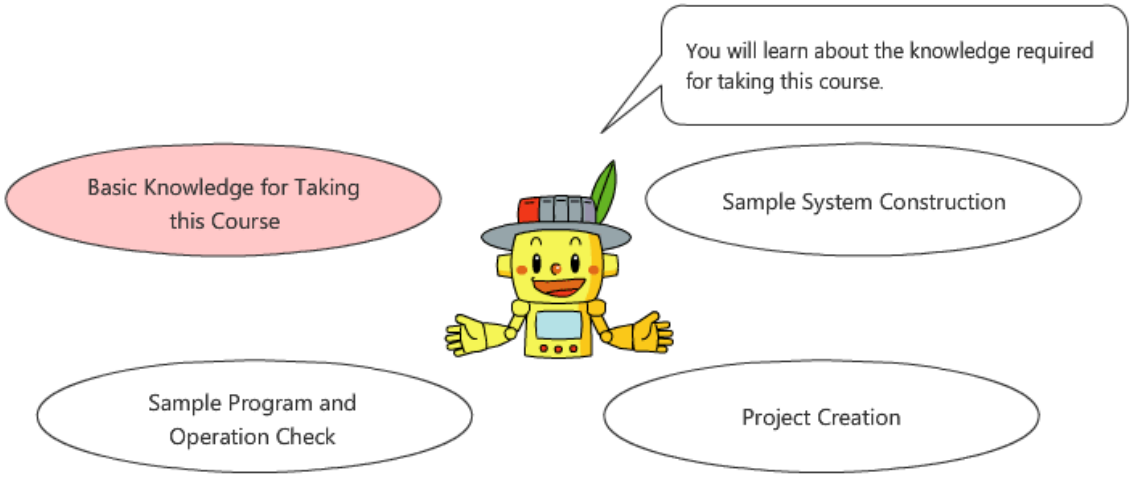
* This course is also suitable for MX-R Model users, as it offers the equivalent explanations.

Click the Forward button at the upper right of the screen to proceed to the next page.

L(NA)0810007ENG

Introduction Purpose of the Course

This course is intended for those who will establish a motion control system using the MELSEC MX Controller MX-F Model for the first time and provides basic knowledge from system design to installation, wiring, setting, and programming.



This course requires the basic knowledge of MELSEC programmable controllers, AC servos, and the positioning control.

Introduction Course Structure

The contents of this course are as follows.

Chapter 1 - Basic Knowledge for Taking this Course

This chapter describes the knowledge required for taking this course.

Chapter 2 - Sample System Construction

This chapter describes the hardware configuration of the sample system.

Chapter 3 - Project Creation

This chapter describes the settings using engineering software.

Chapter 4 - Sample Program and Operation Check

This chapter describes the program contents and the operation of the sample system.

Final Test

Passing grade: 60% or higher

Introduction

How to Use This e-Learning Tool

Go to the next page	>	Go to the next page.
Back to the previous page	<	Back to the previous page.
Move to the desired page	TOC	"Table of Contents" will be displayed, enabling you to navigate to the desired page.
Exit the learning	X	Exit the learning. Window such as "Contents" screen and the learning will be closed.

Cautions for Use

Safety precautions

When you learn by using actual products, please carefully read the safety precautions in the corresponding manuals.

Precautions in this course

The displayed screens of the software version that you use may differ from those in this course.
This course uses the following software version.
For the latest version of each software, check the Mitsubishi Electric FA Website.

MELSOFT GX Works3	Ver.1.120A
MELSOFT MR Configurator2	Ver.1.165X

The following table lists the titles and document numbers of manuals related to this course.
Download the manuals from the Mitsubishi Electric FA Website as needed and refer to them.
Each chapter indicates its relevant manuals.

Manual	Manual No.
MELSEC MX Controller MX-R Model User's Manual	SH-082641ENG
MELSEC MX Controller MX-F Model User's Manual	SH-082634ENG
MELSEC MX Controller MX-R Model Programming Manual	SH-082644ENG
MELSEC MX Controller MX-F Model Programming Manual	SH-082691ENG
MELSEC iQ-R/MELSEC iQ-F Structured Text (ST) Programming Guide Book	SH-081483ENG
MR-J5-G/MR-J5W-G User's Manual (Introduction)	SH-030294ENG
MR-J5 User's Manual (Hardware)	SH-030298ENG
MR-J5 User's Manual (Function)	SH-030300ENG
MR-J5-G/MR-J5W-G User's Manual (Parameters)	SH-030308ENG

Chapter 1 Basic Knowledge for Taking this Course

This chapter describes the knowledge required for taking this course.
The relevant manuals are listed as follows.

Syntax of the ST language

- MELSEC iQ-R/MELSEC iQ-F Structured Text (ST) Programming Guide Book
- MELSEC MX Controller MX-F Model Programming Manual

Instructions supported by the ST language

- MELSEC MX Controller MX-F Model Programming Manual

Labels, structures, and program examples

- MELSEC MX Controller MX-F Model User's Manual*
- MELSEC MX Controller MX-F Model Programming Manual*

*: For users of the MELSEC MX Controller MX-R Model, refer to the user's manual and programming manual for the MX-R Model. This also applies to the following chapters.

1.1 Subject of this Course

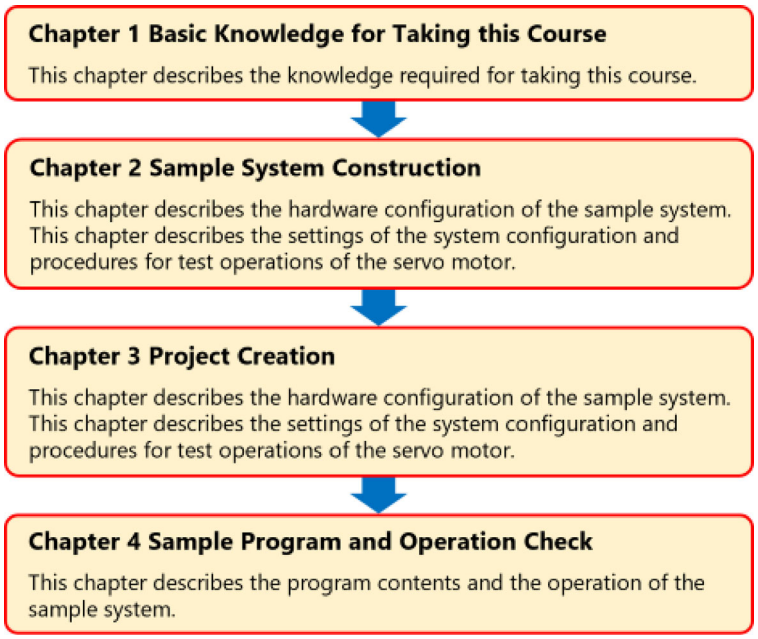
In this course, you will learn how to control the mechanism of a one-axis ball screw by using the MELSEC MX Controller MX-F Model and MELSERVO-J5 series AC Servo.

The following PTP (Point-To-Point) motion*1 is the subject of this course.

*1: Movement between one point to another.

1.2 Flow of this Course

The following shows a flow of this course.



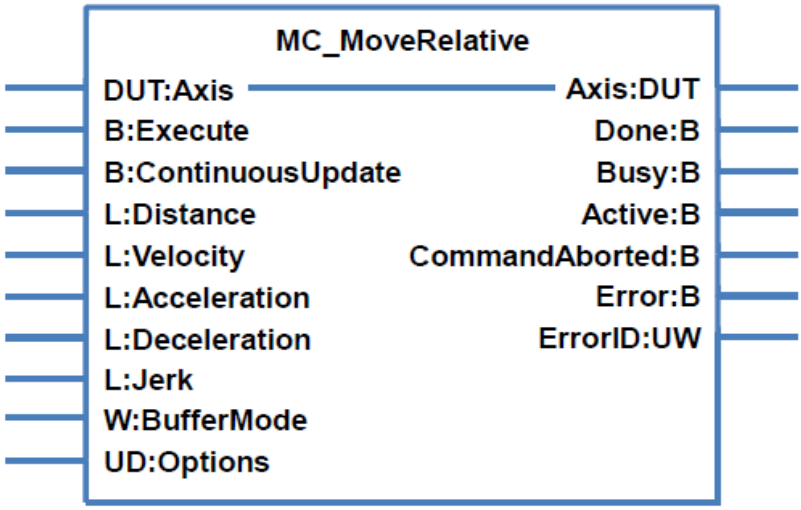
1.3 PLCopen® Motion Control FB

PLCopen® is a third-party organization, aiming at improving the development efficiency of PLC applications, promoting the international standard IEC 61131-3 for PLC programming, and creating and certificating the standard function block (FB) specifications that are independent of the vendor.

Using the FB specified by PLCopen® enables the programming independent of PLC manufacturers since the I/O and operation specifications of the FB are standardized.
This makes the program structured and improves the reusability, resulting in the reduction of engineering cost.
The motion control is defined as Motion Control FB.

The MELSEC MX Controller is compatible with this Motion Control FB (hereafter, referred to as MCFB) and uses this FB for programming.
(For details, refer to Chapter 4.)

Example) MC_MoveRelative (Relative value positioning control)



Basic rules of the ST language (excerpt)

The following shows a part of the sample program.

```
25  //-----Jog Operation---------------------------
26  bJogEnable := (G_bHomeBusy=FALSE) & (G_bPositioningReq=FALSE);
27
28① MCv_Jog_1(
29  Axis         := Axis0001.AxisRef ,
30  JogForward   := G_bJogForwardCMD & (G_bJogBackwardCMD=FALSE) & bJogEnable ,//Input X1
31  JogBackward  := (G_bJogForwardCMD=FALSE) & G_bJogBackwardCMD & bJogEnable ,//Input X2
32  Velocity     := G_leJogVelocity,
33② Acceleration:= G_leJogAcc ,
34  Deceleration:= G_leJogDec ,
35  Jerk         := G_leJogJerk ,
36  Options      := H0,//0:mcAccDec
37  //Done               => ?BOOL? ,
38  Busy               => G_bJogBusy //,
39  //Active             => ?BOOL? ,
40③ //CommandAborted     => ?BOOL? ,
41  //Error              => ?BOOL? ,
42  //ErrorID            => ?WORD?
43  );
```

Comment
All the statements after // or statements between /* and */ or (* and *) are comments.

Add a semicolon ";" at the end of all the statements.

The format of "<variable> := <expression>;" is an assignment statement. Store the result of the formula on the right to the variable on the left.

Function block
Start the function block.
1) FB name
2) Indicated by input variables ":=".
3) Indicated by output variables "=>".

1.5 Label, Array, and Structure

(1) Label, array, and structure

In MELSEC MX Controller programs, labels are used instead of devices and buffer memory numbers.
A label is a variable (a string) used for I/O data or internal processing.
By using labels, you can write programs without considering device and buffer-memory sizes.
Therefore, programs that use labels can be easily reused in systems with different module configurations.
An array is a data type representing an aggregation of labels of the same data type as a single name.
A structure is a data type representing an aggregation of labels of different types as a single name.

(2) Label type

- Local label ····· A local label can be used only in the declared POU and cannot be used in other POUs.
  Each local label is defined by a name, class, and data type.
- Global label ····· A global label provides the same data within a single project.
  It can be used in all programs, including program blocks and function blocks.
  Each global label is assigned a name, class, and data type.
  A device can also be assigned.
- Module label ····· A module label is automatically generated by the engineering tool for each module and is unique to that module. It can be used as a global label.
- Network label ····· A network label enables access to other modules on the network. It can be used as a global label.

(3) Label data type

The following table lists the main label data types.

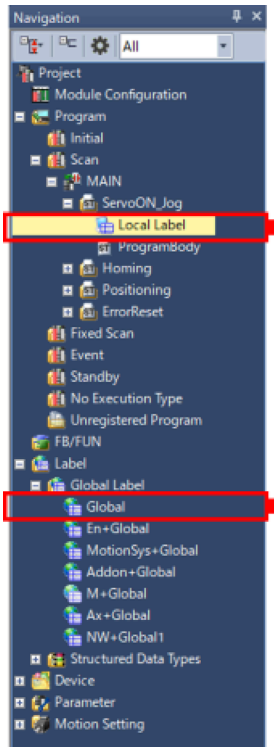
The sample program in this course uses label prefixes to indicate data types.

Data type		Range	Prefix
Bit	BOOL	FALSE(0), TRUE(1)	b
Word (unsigned)/bit string (16 bits)	WORD (UINT)	0 to 65535	u
Double word (unsigned)/bit string (32 bits)	DWORD (UDINT)	0 to 4294967295	ud
Word (signed)	INT	-32768 to 32767	w
Double word (signed)	DINT	-2147483648 to 2147483647	d
Single-precision real number	REAL	-2^{128} to -2^{-126}, 0, 2^{-126} to 2^{128}	e
Double-precision real number	LREAL	-2^{1024} to -2^{-1022}, 0, 2^{-1022} to 2^{1024}	le
Time	TIME	T#-24d20h31m23s648ms to T#24d20h31m23s647ms	tm
Timer	TIMER	TIMER is a structure. S (contact): BOOL C (coil): BOOL N (current value): WORD	td

For global labels, "G_" is added to the beginning of the label name.

(4) Label registration method

- Local label
  Double-click [Program] → [Program name] → [Local Label] in the project tree to open the local label editor. Each program has its own local label settings.
- Global label
  Double-click [Label] → [Global Label] → [Global] in the project tree to open the global label editor.



	Label Name	Data Type	Class	Initial Val	Constant	English(Display Target)
1	MC_Power_1	MC_Power	VAR			
2	bPowerStatus	Bit	VAR			Servo ON/OFF status
3	bReadyStatus	Bit	VAR			Ready ON/OFF status
4	bPowerBusy	Bit	VAR			MC_Power Busy
5	bPowerError	Bit	VAR			MC_Power Error
6	uPowerErrorID	Word [Unsigned]/Bit String [16-bit]	VAR			MC_Power Error ID
7	bJogEnable	Bit	VAR			Jog Operation Enable
8	MCv_Jog_1	MCv_Jog	VAR			
9						

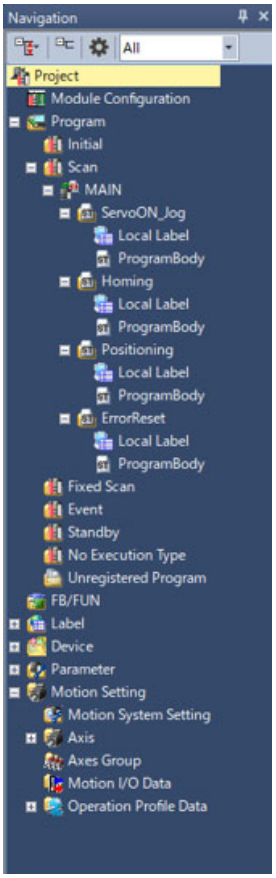
Example of the local label editor

	Label Name	Data Type	Class	Initial Value	Constant	English(Display Target)
1	G_leJogVelocity	FLOAT [Double Precision]	VAR_GLOBAL			JOG Velocity
2	G_leJogAcc	FLOAT [Double Precision]	VAR_GLOBAL			JOG Acceleration
3	G_leJogDec	FLOAT [Double Precision]	VAR_GLOBAL			JOG Deceleration
4	G_leJogJerk	FLOAT [Double Precision]	VAR_GLOBAL			JOG Jerk
5	G_bJogBusy	Bit	VAR_GLOBAL			MC_Jog Busy
6	G_bPositioningReq	Bit	VAR_GLOBAL			Positioning Request
7	G_lePoint0Address	FLOAT [Double Precision]	VAR_GLOBAL			Home Position Address
8	G_lePoint1 Address	FLOAT [Double Precision]	VAR_GLOBAL			Positioning Address
9	G_bHomeBusy	Bit	VAR_GLOBAL			MC_Home Busy
10	G_bSRVONCMD	Bit	VAR_GLOBAL			Servo ON

Example of the global label editor

1.6 Program Type

Sequence and motion control are combined into a single program, making it simpler.



Initial execution type program
This program is executed only once when the MELSEC MX Controller is powered on or when its state is changed from STOP to RUN.

Scan execution type program
This 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 execution type program
This is an interrupt program which is executed at a specified time interval. Different from a normal interrupt program, this type of program does not require the interrupt pointer (I) or IRET instruction and is executed for each program file unit.

Event execution type program
This program starts execution when triggered by a specified event. The program is executed in the order specified by the program settings of the CPU parameters. If the execution conditions of the specified trigger are met when the execution turn of the event execution type program comes, the program is executed.

Standby type program
This program is executed only when its execution is requested.

No execution type, unregistered program
This program is not executed on the MELSEC MX Controller. Programs with no execution type specified (if selected) are written to the controller.
Unregistered programs are not written.

1.7 Summary

In this chapter, you have learned:

- PLCopen® Motion Control FB
- Programming Using ST Language
- Label, Array, and Structure
- Program Type

Important points

PLCopen® Motion Control FB	• PLCopen®, a third-party organization, develops the standard FB specifications that are independent of the vendor. • The motion control is defined as Motion Control FB.
Programming Using ST Language	• All statements end with a semicolon ";". • The assignment statement is represented by <variable> := <expression>;. • The input variable of the FB is indicated by ":=", and the output variable is indicated by "=>"
Label, Array, and Structure	• The label types include the local label, global label, module label, and network label. • An array is an aggregation of labels of the same variable type. • A structure is an aggregation of labels of different variable types.
Program Type	• The program execution types include the initial execution type, scan execution type, fixed scan execution type, event execution type, standby execution type, and no execution type/unregistered program.

Chapter 2 Sample System Construction

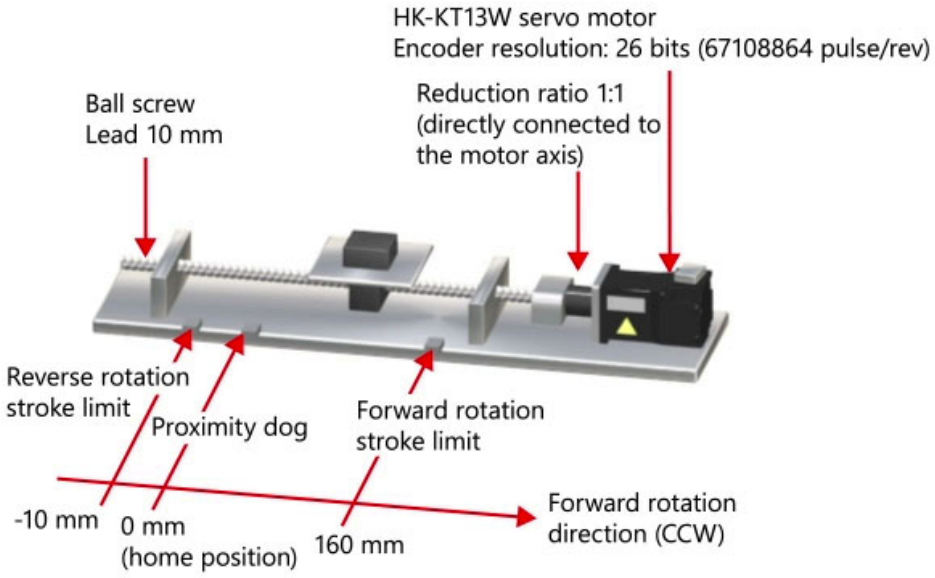
This chapter describes the hardware configuration of the sample system and the test operation procedure.
The relevant manuals are listed as follows.

System configuration and wiring

- MELSEC MX Controller MX-F Model User's Manual
- MELSEC MR-J5 User's Manual (Hardware)

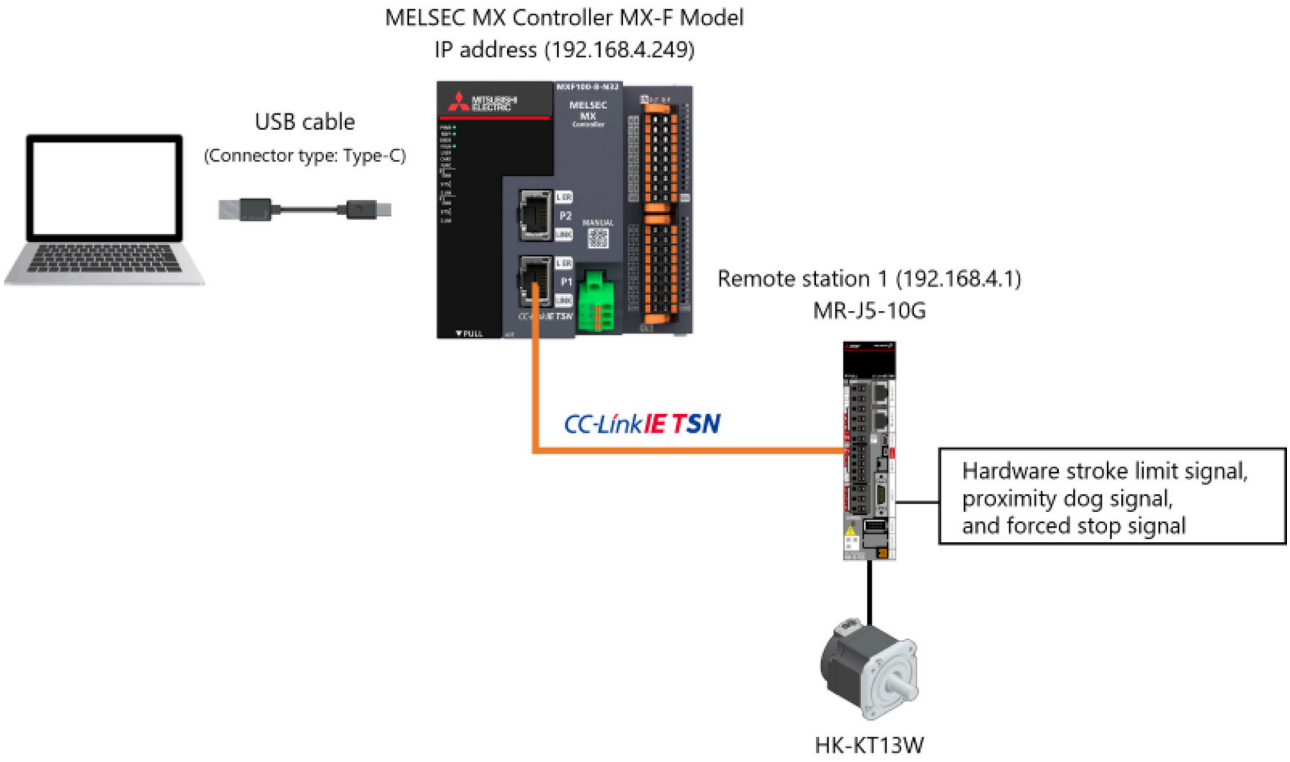
2.1 Machine Configuration

Use the mechanism of a one-axis ball screw. The machine specifications are as follows.



2.2 System Configuration

The system configuration is as follows.

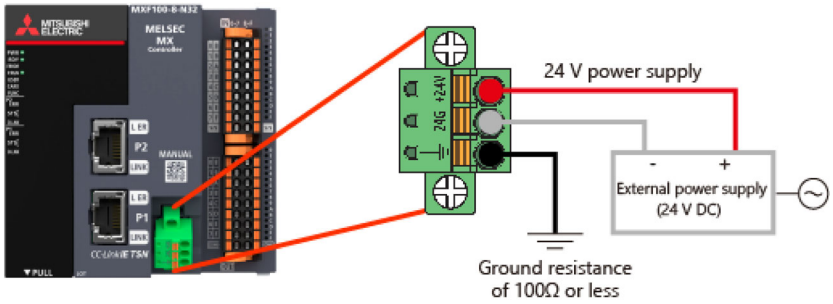


2.3 Wiring

2.3.1 Connecting a servo motor and a servo amplifier

(1) This section describes the power supply wiring for the MELSEC MX Controller MX-F Model.

- Connect the power cable to the +24V and 24G terminals on the power terminal block.
- Ground the FG terminal on the power terminal block with a ground resistance of 100 ohms or less.

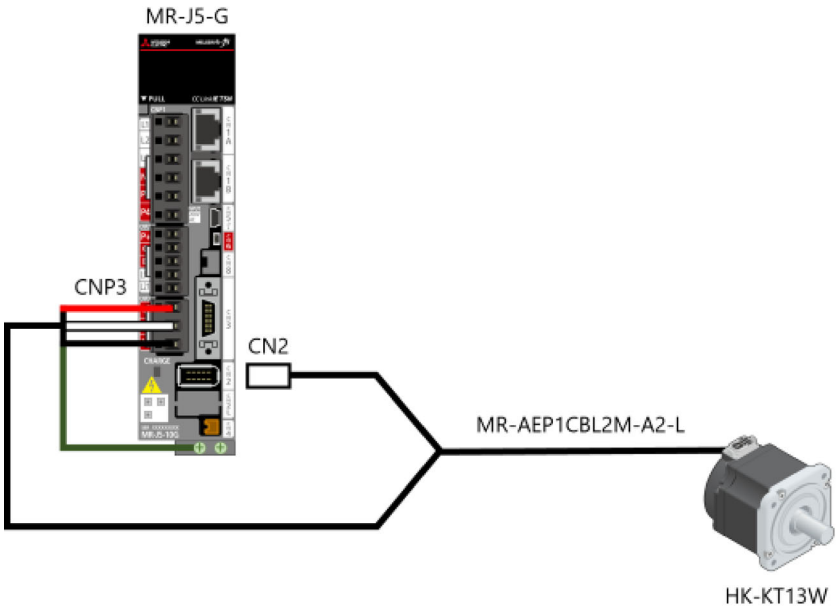


The following table describes the wire that can be connected to the power terminal block. It is recommended to use ferrules for reliable connections (when processing wires, use dedicated crimping tools).

Number of wires connected per terminal	Wire size		
	Solid wire, stranded wire	Ferrule with insulation sleeve	Ferrule without insulation sleeve
One wire	22 AWG (0.3 mm^2)	22 to 16 AWG (0.3 to 1.5 mm^2)	22 to 16 AWG (0.3 to 1.5 mm^2)

2.3.1 Connecting a servo motor and a servo amplifier

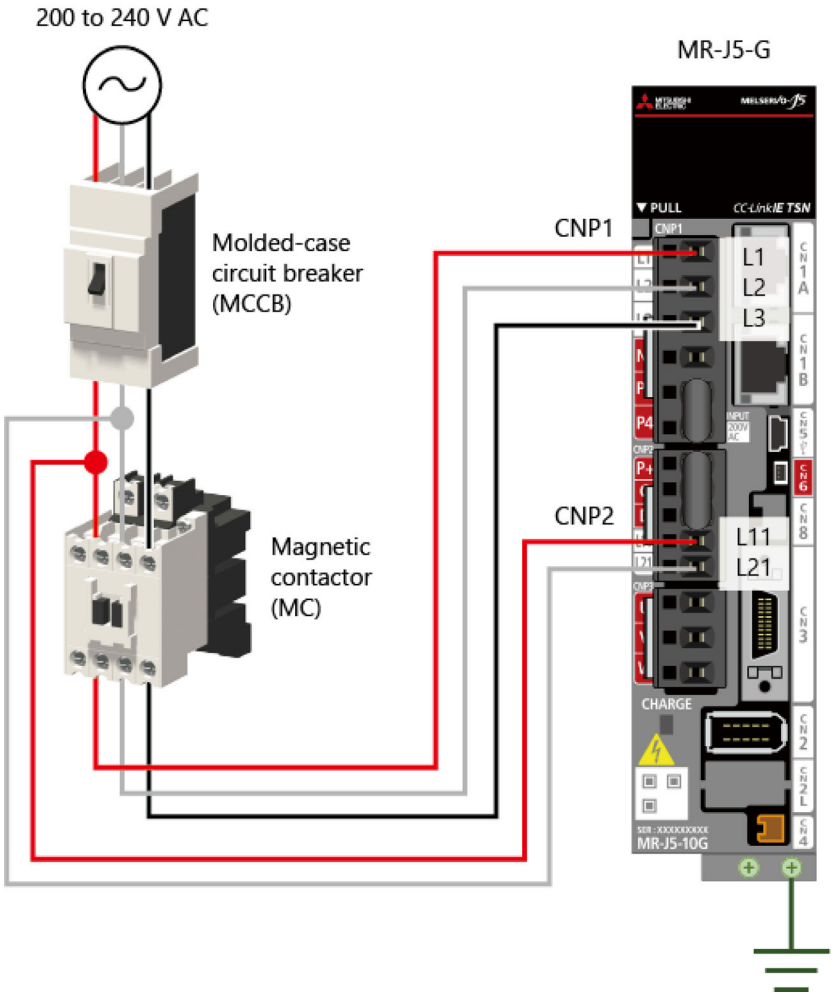
For the servo motor power cable and encoder cable, use the MR-AEP1CBL2M-A2-L option cable (single cable type).



2.3.2 Wiring a power supply and network cables

(2) Wiring the power supply of the servo amplifier
Wire the power supply to the main circuit power supply (L1, L2, and L3) and control circuit power supply (L11 and L21) of the servo amplifier.
The following shows the schematic diagram. Actual wiring and applicable wire size vary depending on the capacity. For details, refer to the user's manual for the servo amplifier used.

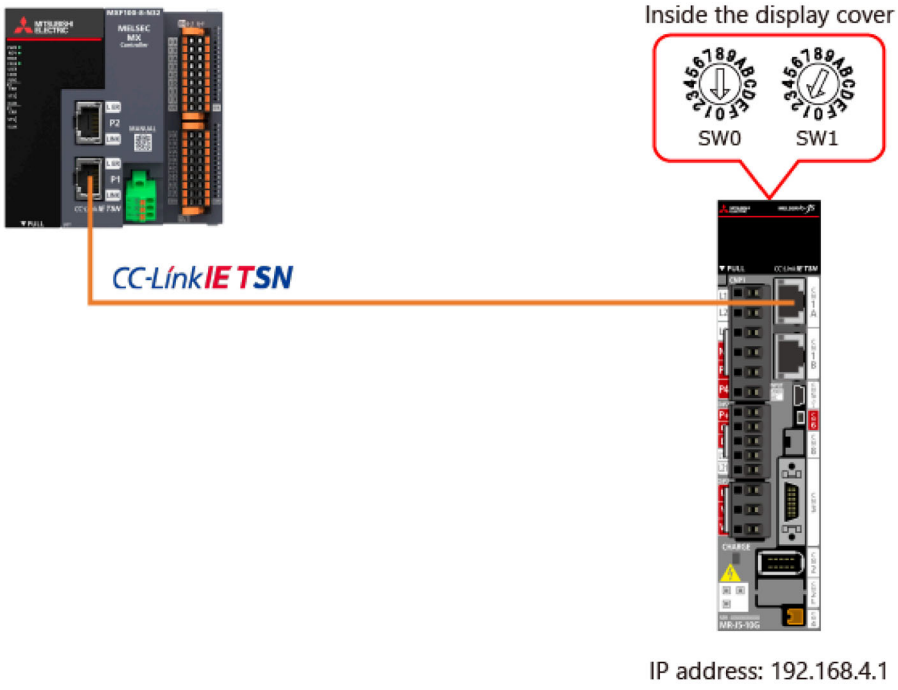
- Always use a molded-case circuit breaker (MCCB) with the input cables of the main circuit power supply.
- Always connect a magnetic contactor (MC) between the main circuit power supply and the L1, L2, and L3 terminals of the servo amplifier.



(3) Wiring a network cable

Wire a network cable (CC-Link IE TSN cable) that satisfies the following standards.

Communication speed	Ethernet cable	Connector	Standard
1 Gbps	Category 5e or higher, straight cable (double shielded, STP)	RJ45 connector	Cables that satisfy the following standards • IEEE 802.3 (1000BASE-T) • ANSI/TIA/EIA-568-B (Category 5e)



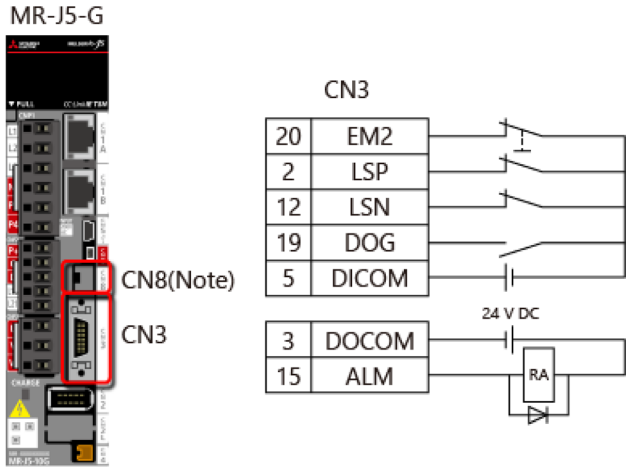
2.3.3 Wiring peripheral circuits

(1) I/O circuit of the servo amplifier

Wire the I/O circuit of the servo amplifier as follows.

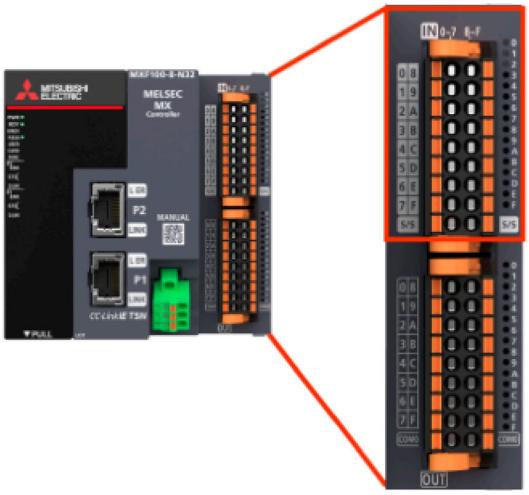
Wire the proximity dog signal, forward/reverse rotation stroke limit switches, and forced stop switch.

In addition, configure the circuit in which the magnetic contactor (MC) is turned off by ALM output.

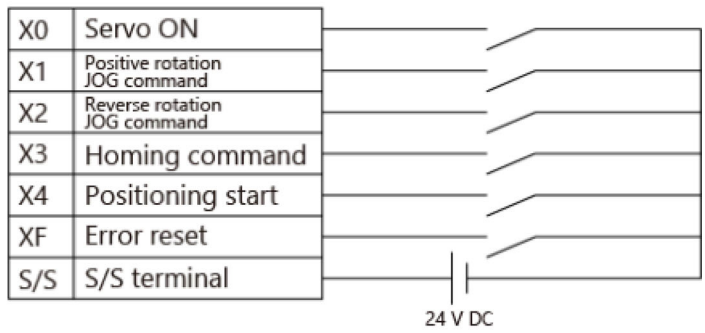


(Note) In this course, the STO function is not used. Thus, do not disconnect the short-circuit connector supplied with the servo amplifier from CN8.

(2) External circuit of the MELSEC MX Controller MX-F Model input terminal block
Wire the external input circuit of the controller as follows.



X0	Servo ON
X1	Positive rotation JOG command
X2	Reverse rotation JOG command
X3	Homing command
X4	Positioning start
XF	Error reset
S/S	S/S terminal

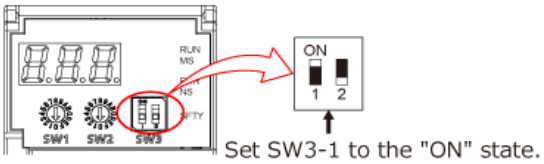


Use the alternate operation switch only for Servo ON (X0), and use the momentary operation switch for other signals.

2.4 Test Operation

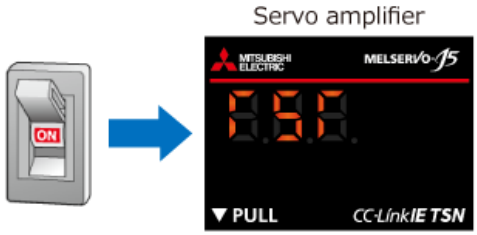
After the wiring is completed, perform a test operation with a single servo amplifier to check the rotation direction and others. Follow the procedures below to perform the test operation.

(1) Turn off the servo amplifier and programmable controller.
(2) Turn on the DIP switch (SW3-1) of the servo amplifier.

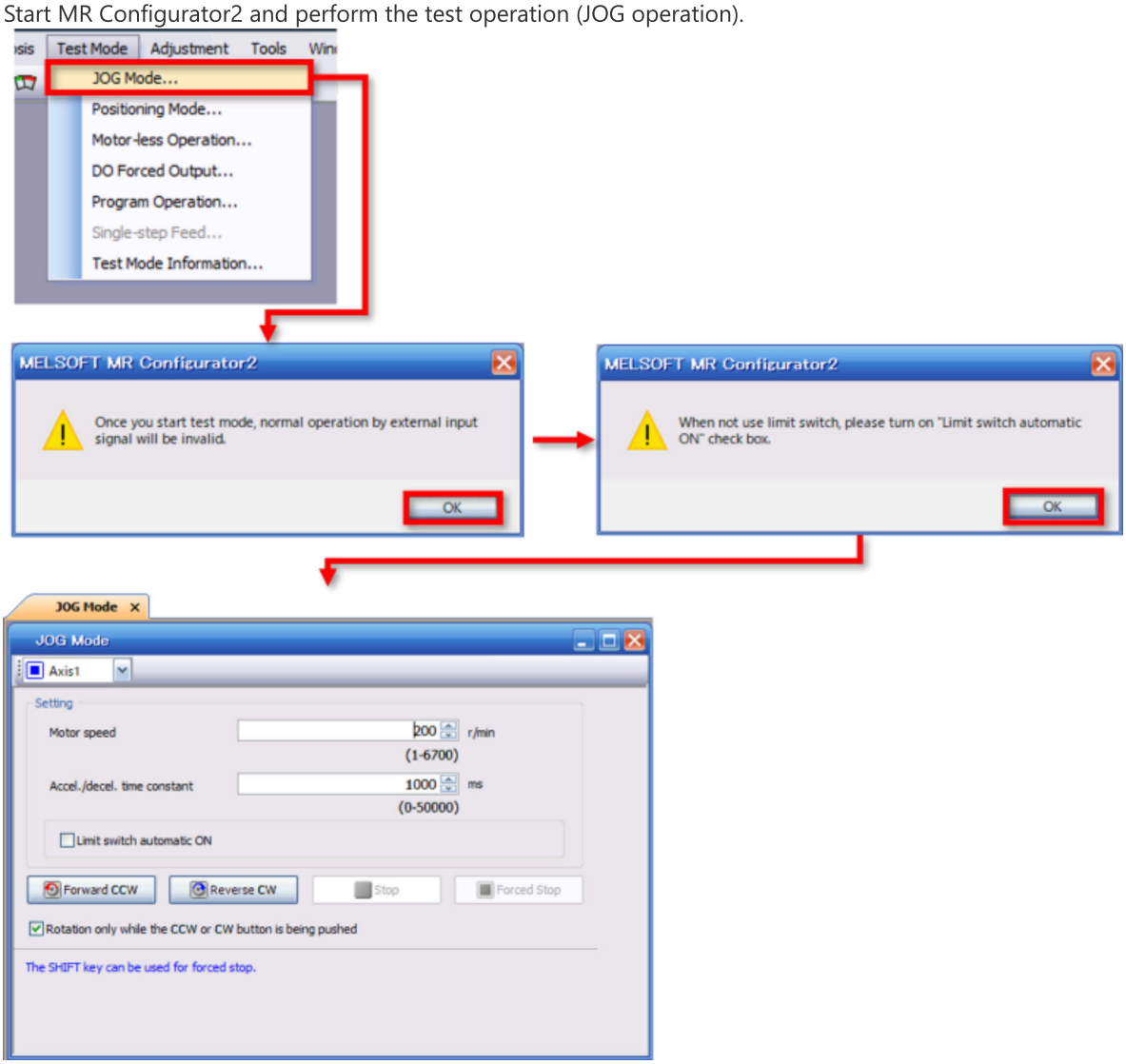


Set SW3-1 to the "ON" state.

(3) Connect the servo amplifier and a personal computer with a USB cable or Ethernet cable. (Note)
(4) Turn on the servo amplifier. "TST" is displayed on the display.



(5) Start MR Configurator2 and perform the test operation (JOG operation).



(6) Check the rotation direction and machine operation.
(7) After the test operation is completed, turn off the servo amplifier, and turn off the DIP switch (SW3-1).

(Note) When using an Ethernet cable, change the project of MR Configurator2 to a multi-axis project.

Tips
When using multiple servo amplifiers, connect them via Ethernet, eliminating the need to replace cables.

2.5 Summary

In this chapter, you have learned:

- Machine Configuration
- System Configuration
- Wiring
- Test Operation

Important points

Machine Configuration	• Use a one-axis ball screw in the sample system.
System Configuration	• Connect the MR-J5-G servo amplifier to the MELSEC MX Controller MX-F Model.
Wiring	• Use the option cable (single cable type) for the servo motor. • Set the IP address fourth octet of the servo amplifier with the rotary switches. • Connect the proximity dog signal, limit switches, and forced stop switch to the servo amplifier. • Connect operation command switches to the MELSEC MX Controller MX-F Model input terminal block.
Test Operation	• Change the DIP switch setting of the servo amplifier, and connect it to a personal computer. • Check the rotation direction of the servo motor and machine operation using the test operation function of MR Configurator2.

Chapter 3 Project Creation

In this chapter, you will learn how to create projects required for operating the MELSEC MX Controller by using the sample program.
Start GX Works3 and follow the instructions on the windows.
Or, download the following sample program and check the settings.

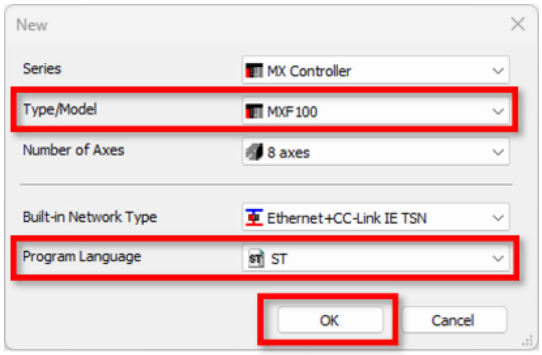
Sample_MXContBasic_en.zip GX Works3 Ver.1.116W or later is required.

The relevant manual is listed as follows.

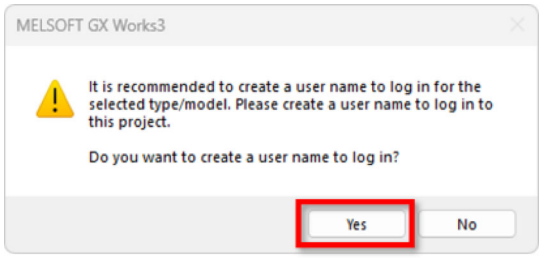
System configuration and wiring

MELSEC MX Controller MX-F Model User's Manual

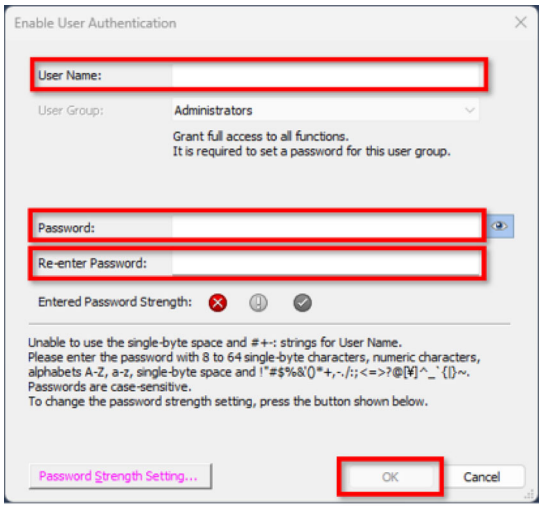
(1) Select [Project] → [New] in GX Works3.
In the following window, select the controller type/model and the programing language.
For the sample program, set the type/model to [MXF100] and the program language to [ST].
After completing the settings, click the [OK] button.



(2) Click the [Yes] button.

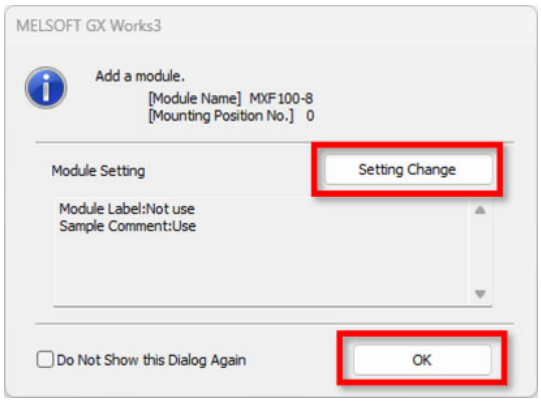


(3) Set each item*1 and click the [OK] button.
Save the project with a name.

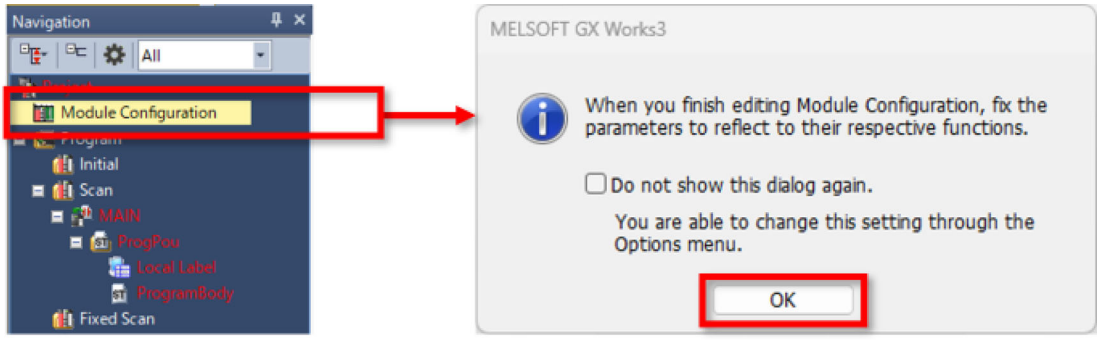


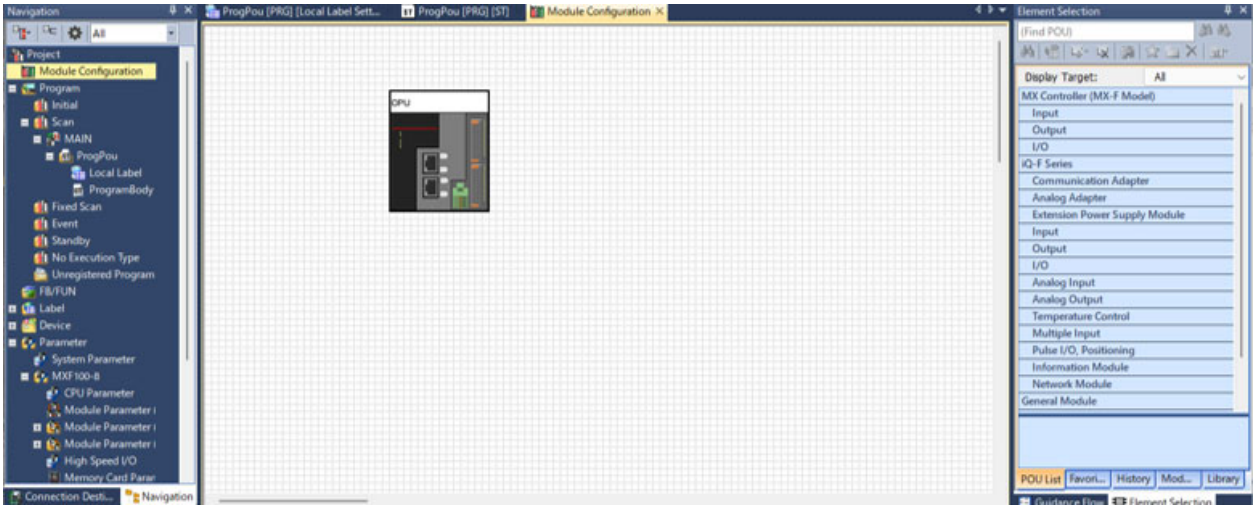
*1 You can set a password to prevent theft, tampering, and unauthorized execution resulting from unauthorized access.
For details, please refer to the manual.

(4) Set each item as follows. You can set whether to use the module label and sample comment.
To change these settings, click the [Setting Change] button and set "Use Module Label" to "Yes".
Click the [OK] button to create the project.

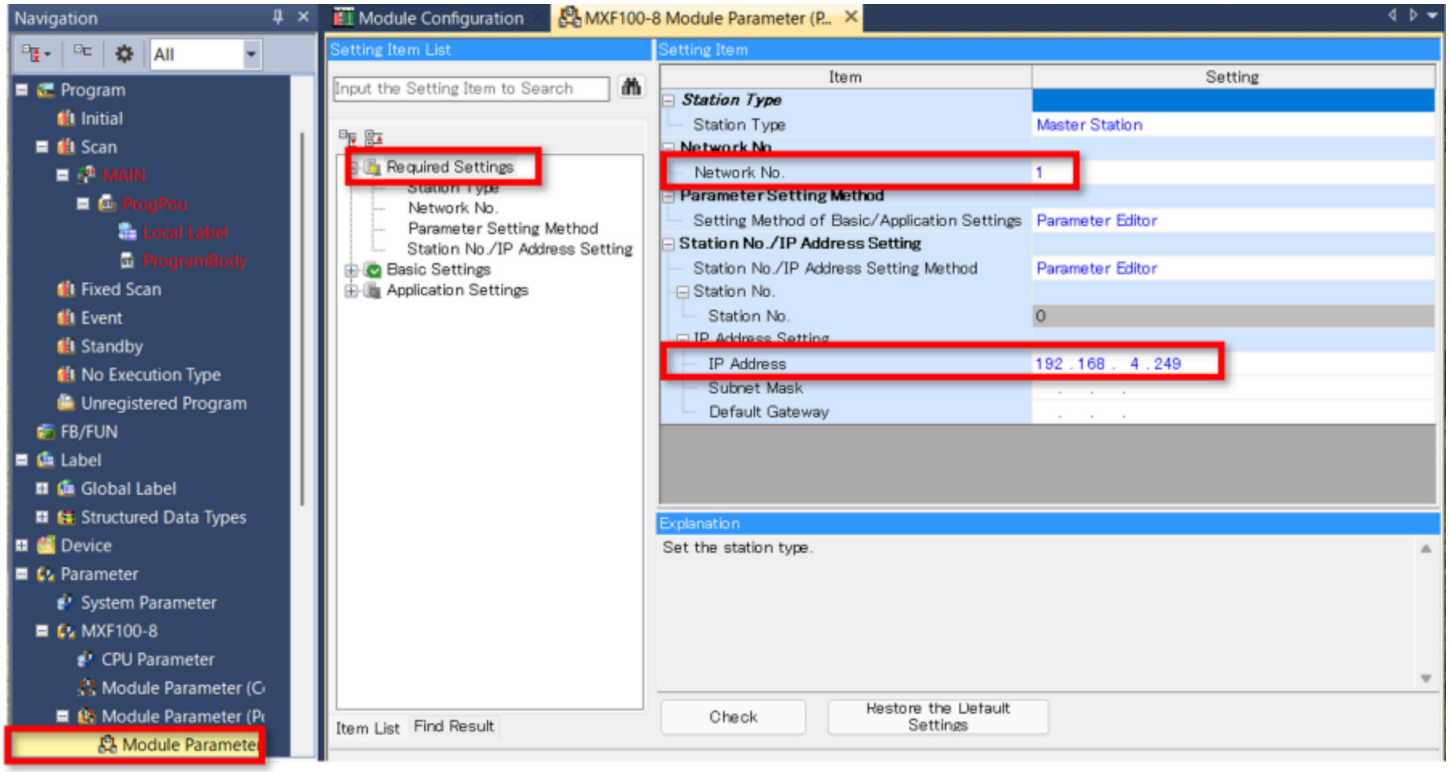


(1) Double-click [Module Configuration] in the project tree.
When the following window appears, click the [OK] button.





(2) In the Navigation window, go to [Parameter] → [MXF100-8] → [Module Parameter (Port 1: CC-Link IE TSN) → [Module Parameter (CC-Link IE TSN)] → [Required Settings], and set "Network No." and "IP Address".
(This sample program uses the initial value.)
Network No.: 1
IP Address: 192.168.4.249

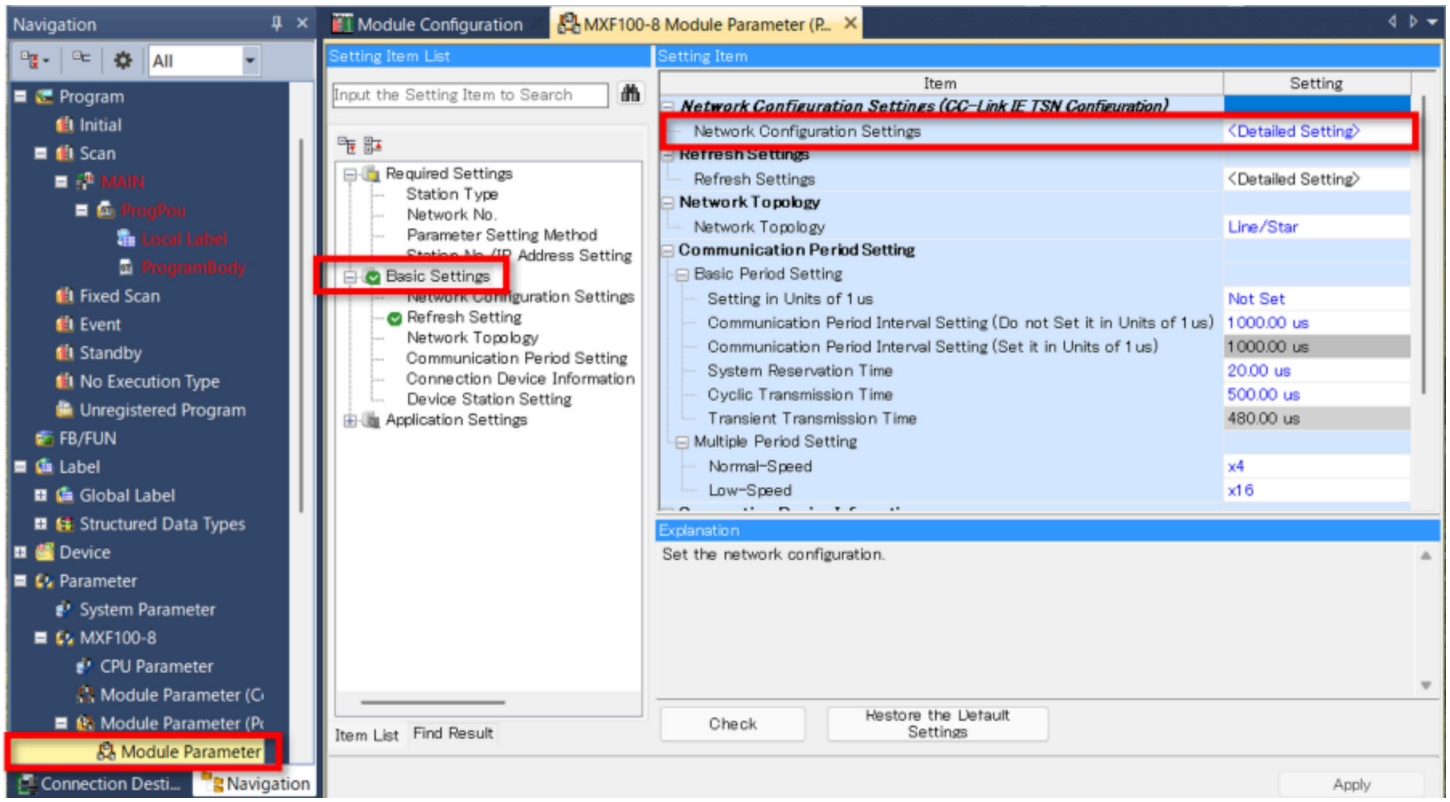


3.3 Network Configuration Setting

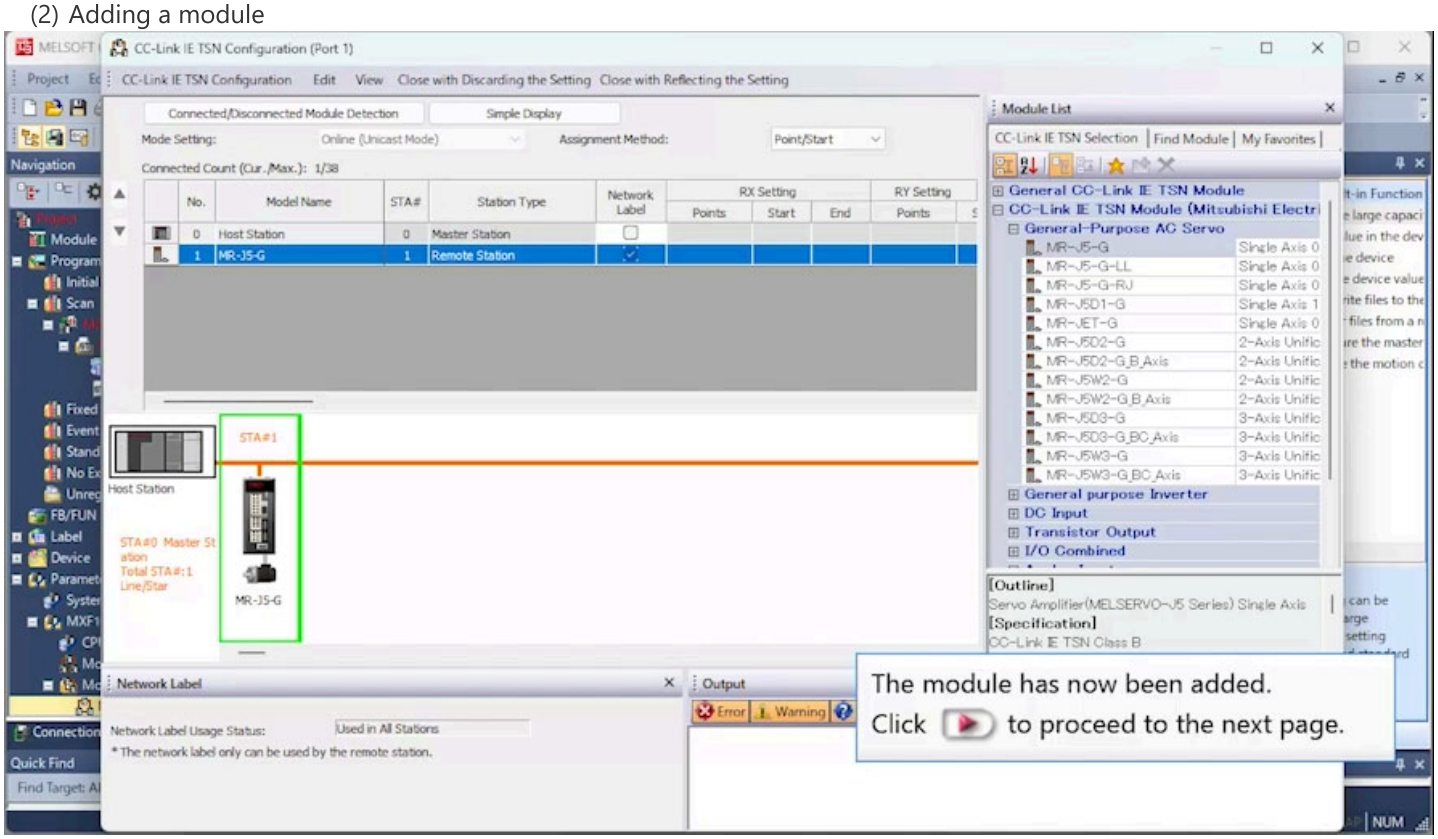
3.3.1 Network parameter (network configuration)

(1) Network configuration setting

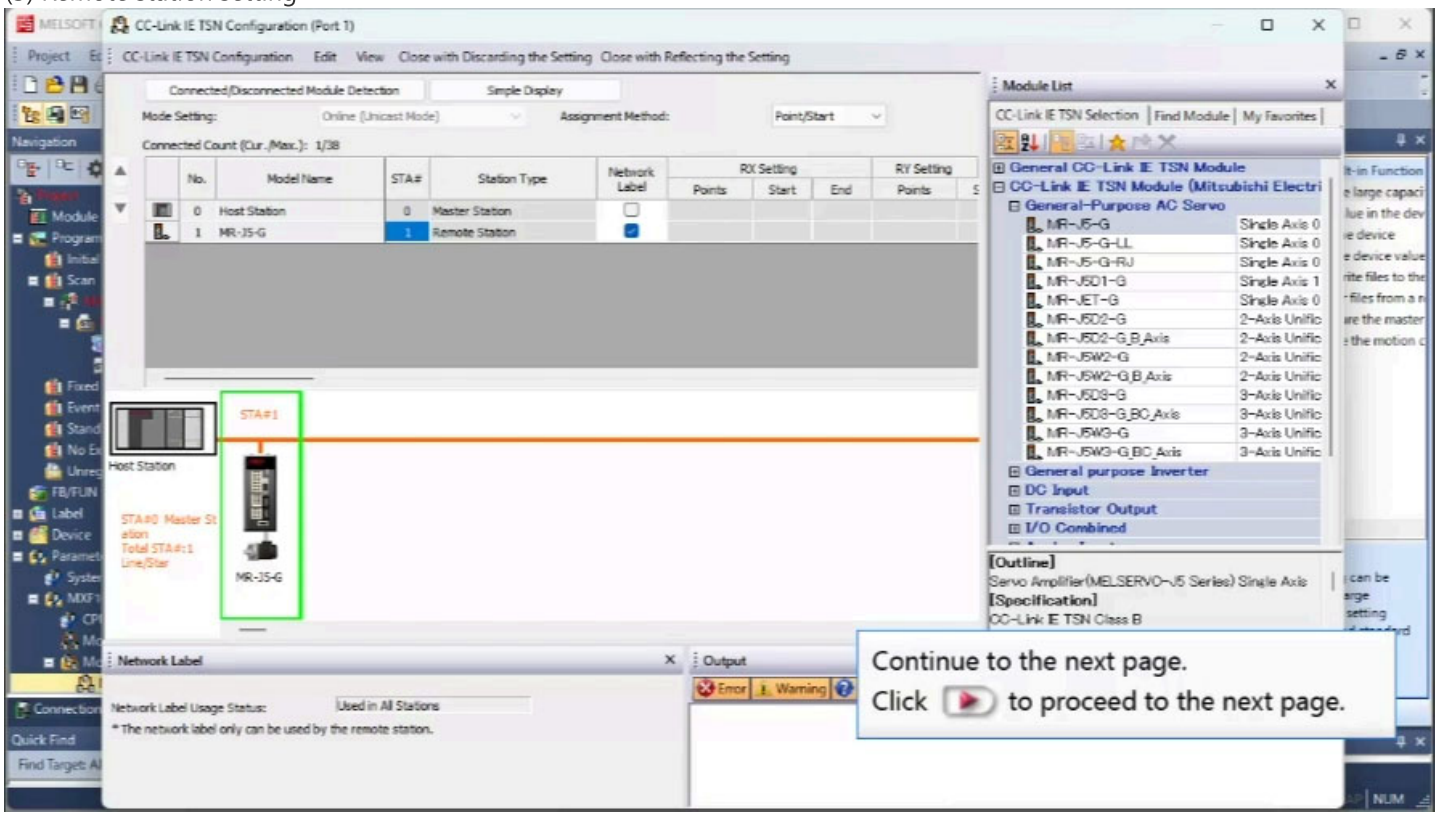
In the Navigation window, go to [Parameter] → [MXF100-8] → [Module Parameter (Port 1: CC-Link IE TSN)] → [Module Parameter (CC-Link IE TSN)] → [Basic Settings], and double-click in "Network Configuration Settings".



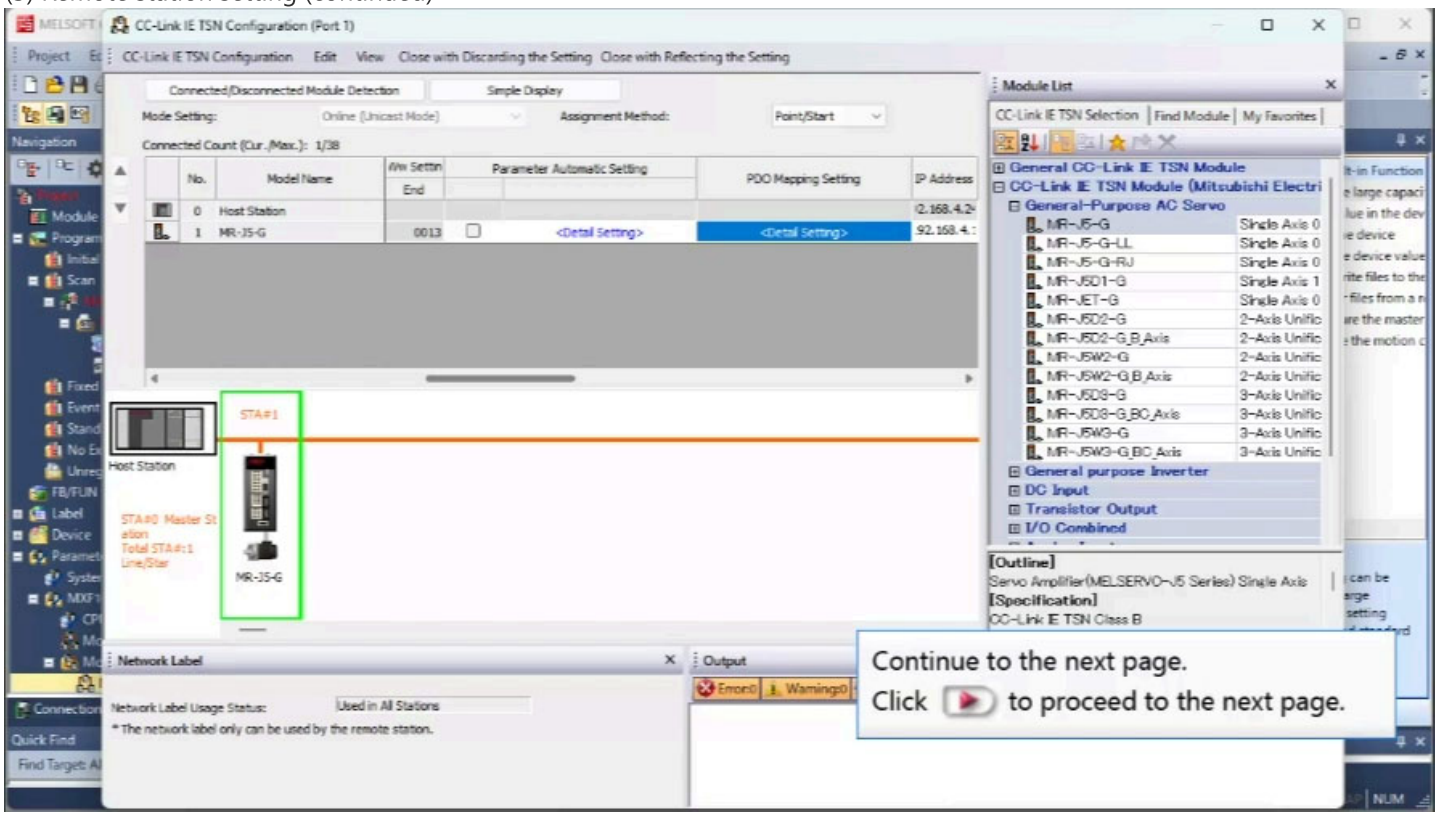
(2) Adding a module



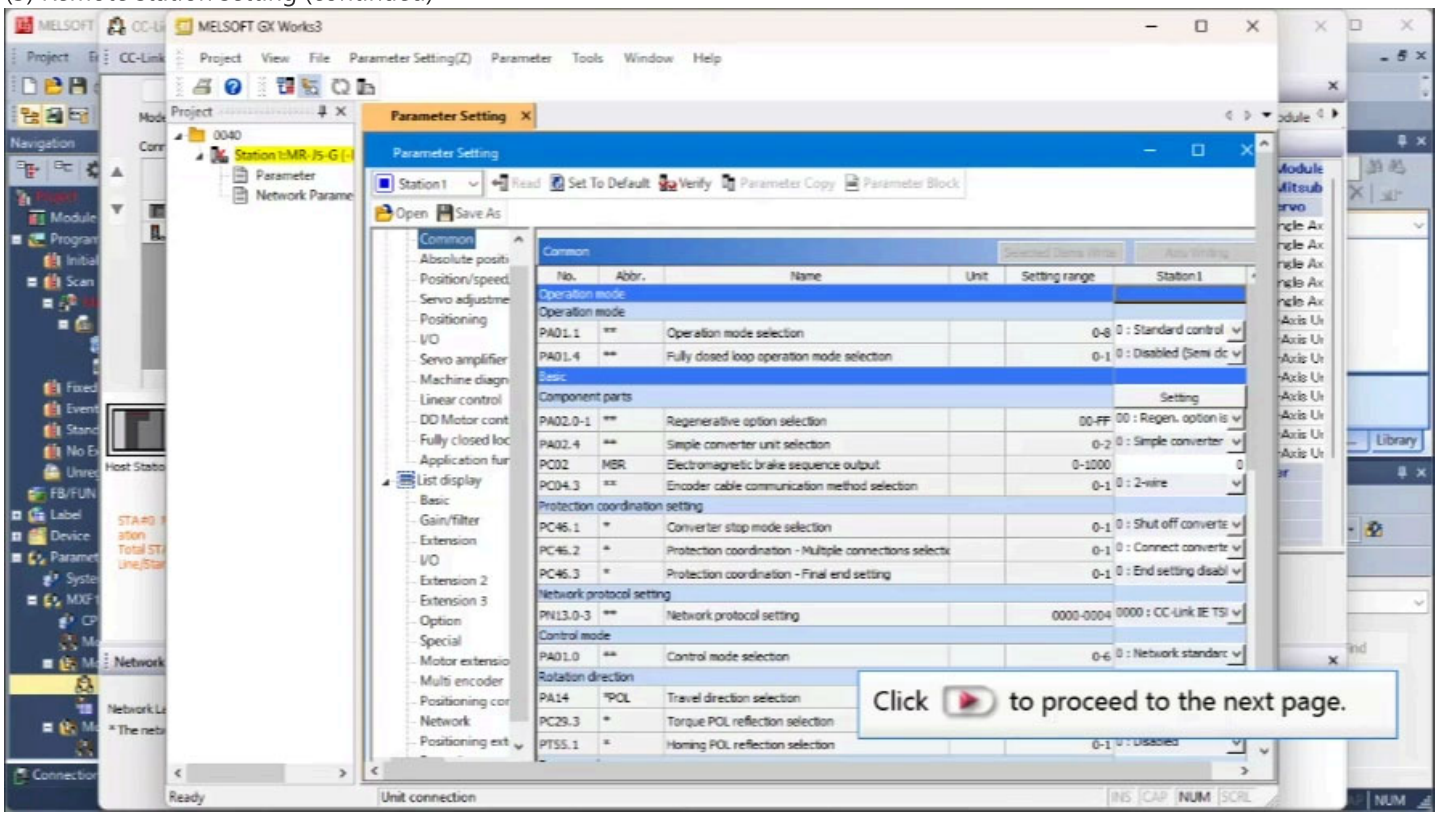
(3) Remote station setting



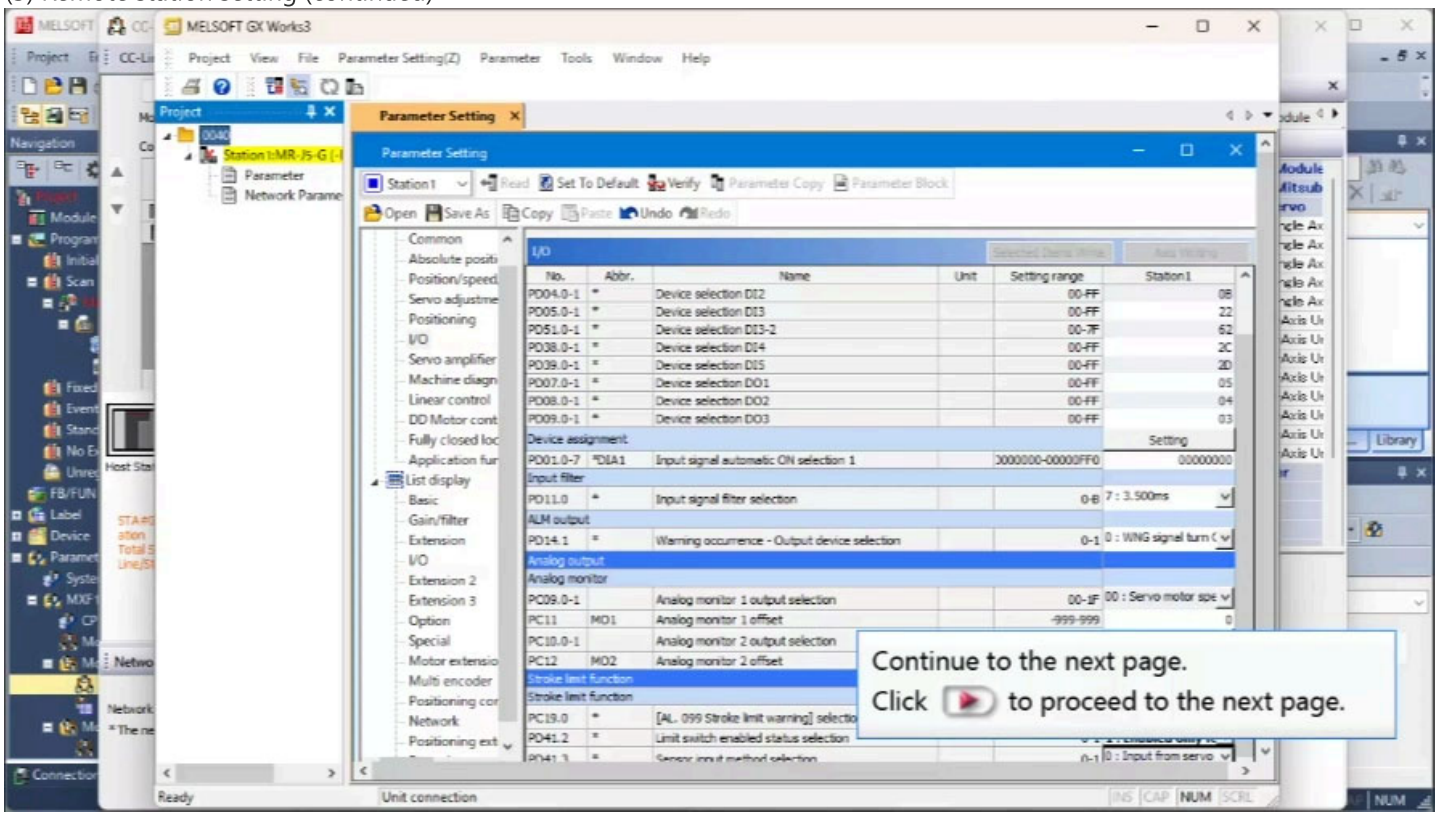
(3) Remote station setting (continued)



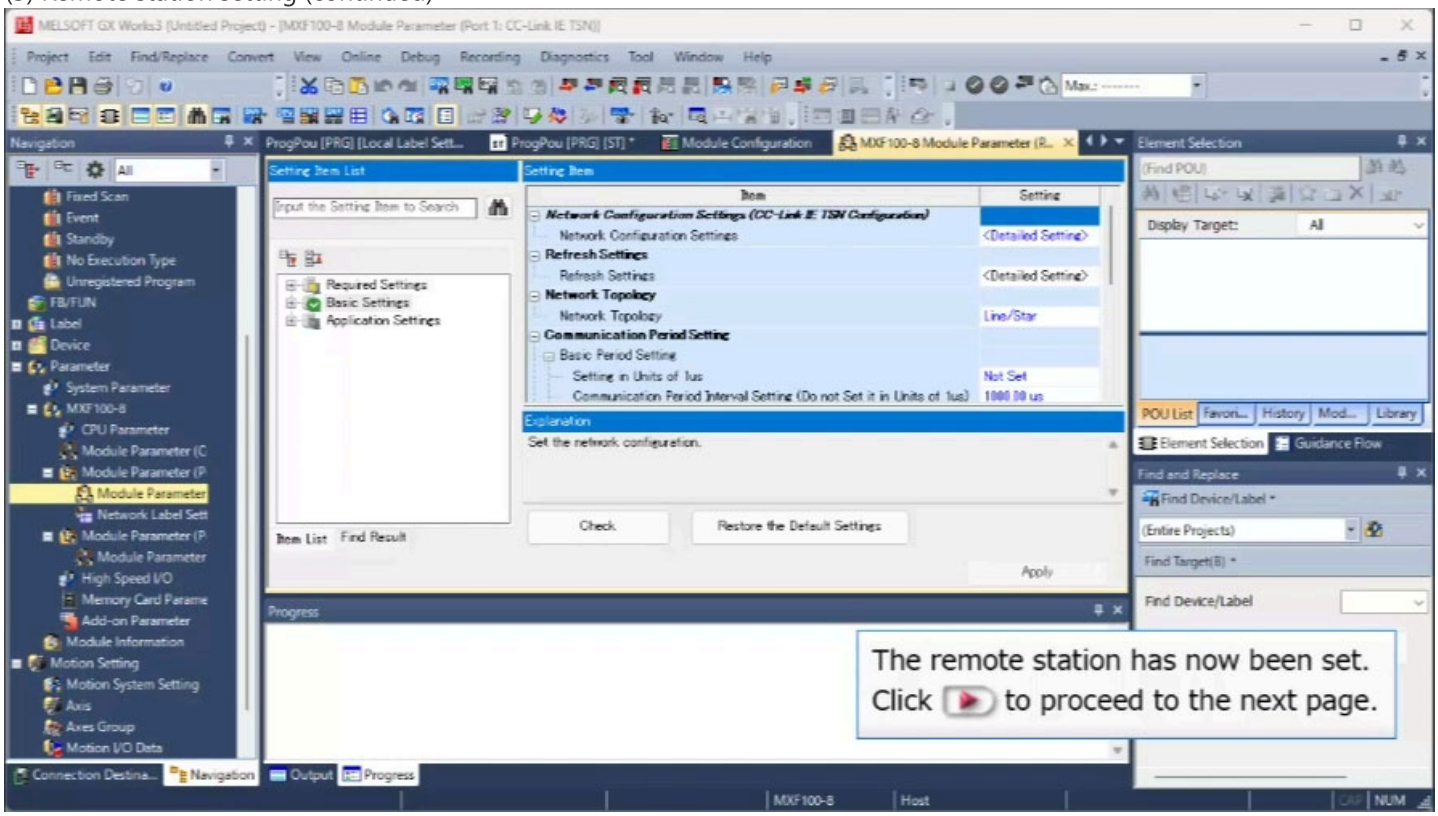
(3) Remote station setting (continued)



(3) Remote station setting (continued)



(3) Remote station setting (continued)



(4) PDO mapping

PDO is an abbreviation for Process Data Object, which is one of the communication profiles of CANopen objects.
PDO communication is equivalent to the conventional CC-Link cyclic transmission.
The object dictionary (OD) can be accessed directly.

PDO mapping links data (objects) to be sent and received between the controller and devices in cyclic transmission (PDO communication).

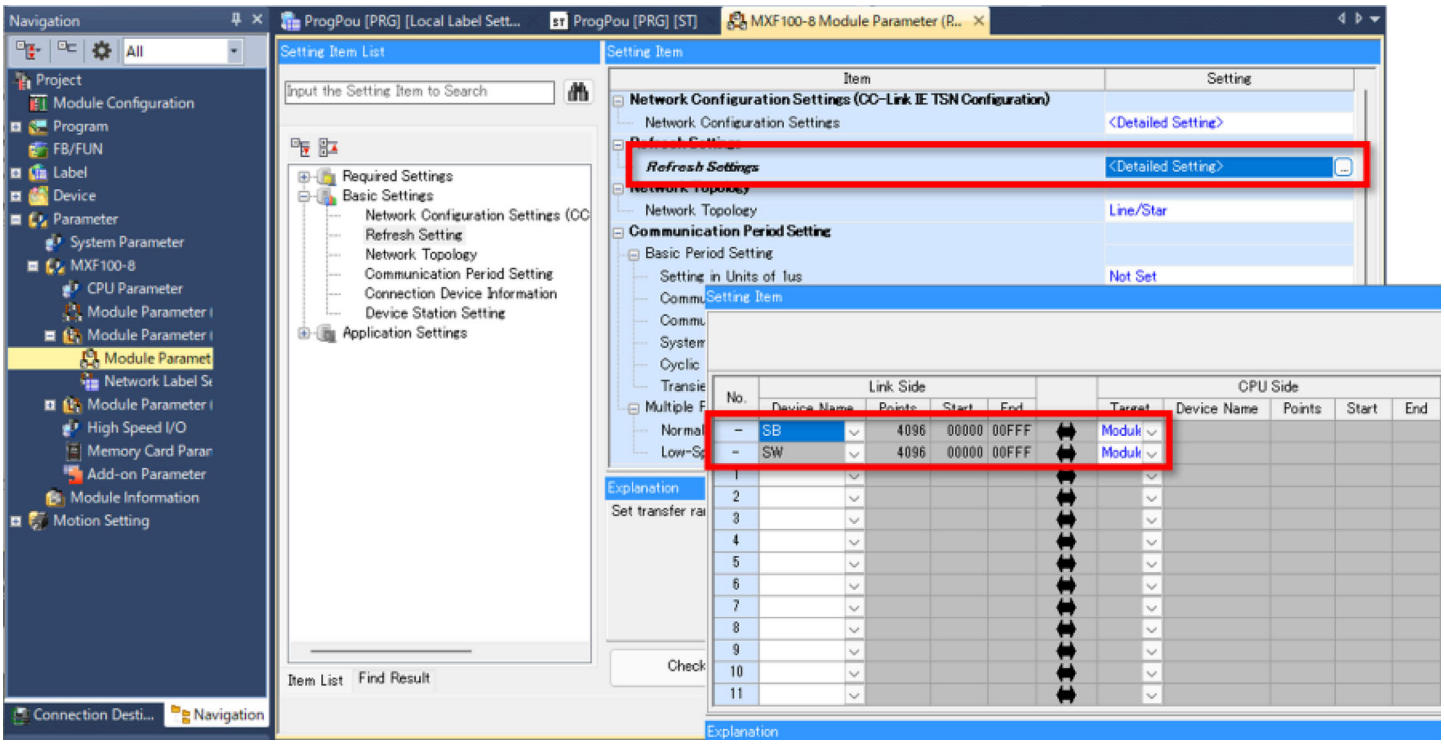
When adding remote stations or changing the IP address, perform the PDO mapping again.

Digital Inputs has been added to the PDO mapping so that the input signal status of the servo amplifier is sent to the MELSEC MX Controller in cyclic transmission.

(5) Refresh setting

Double-click <Detailed Setting> in "Refresh Settings".

Set the refresh target for SB and SW to "Module Label".

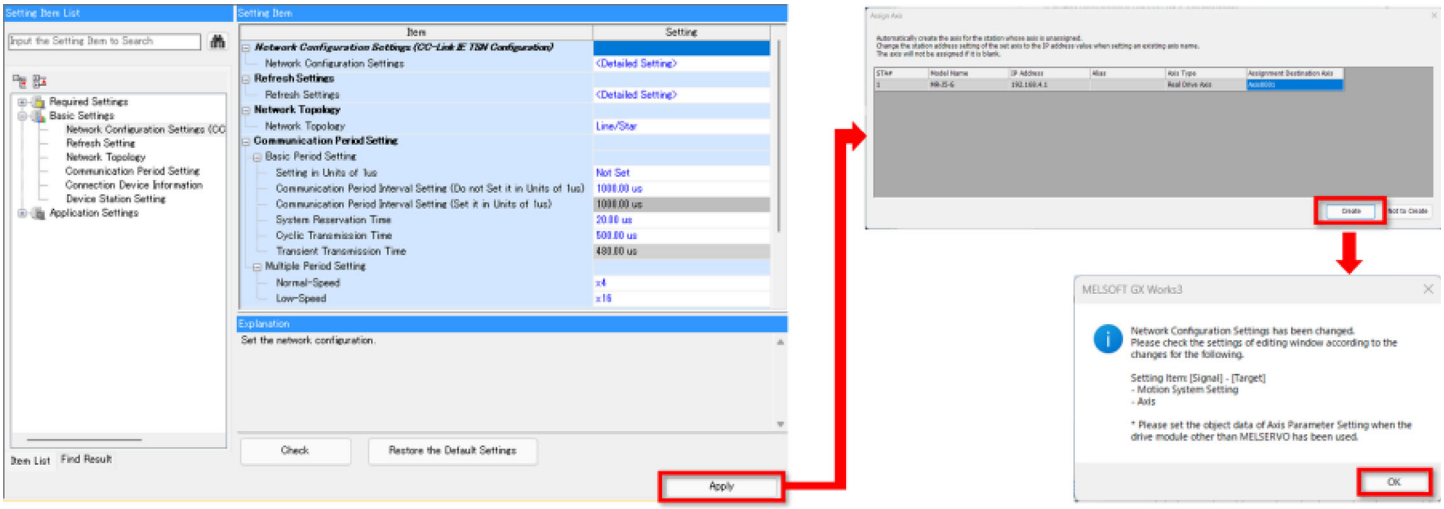


(6) Confirming module parameters

When the window returns to the main window of GX Works3, confirm the set parameters.

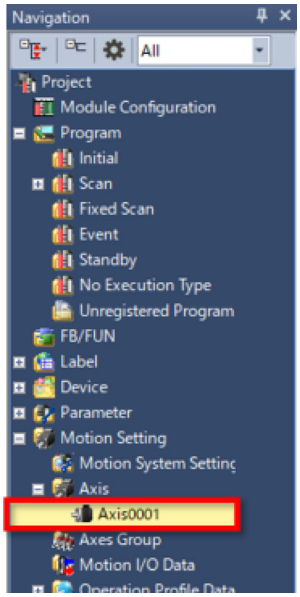
Be sure to click the [Apply] button at the bottom right of the window.

The "Assign Axis" window appears. Click the [Create] button.



3.4 Motion Setting

Configure the parameters for the axis created in the network configuration setting.
In the Navigation window, go to [Motion Setting] → [Axis], and double-click [Axis0001].



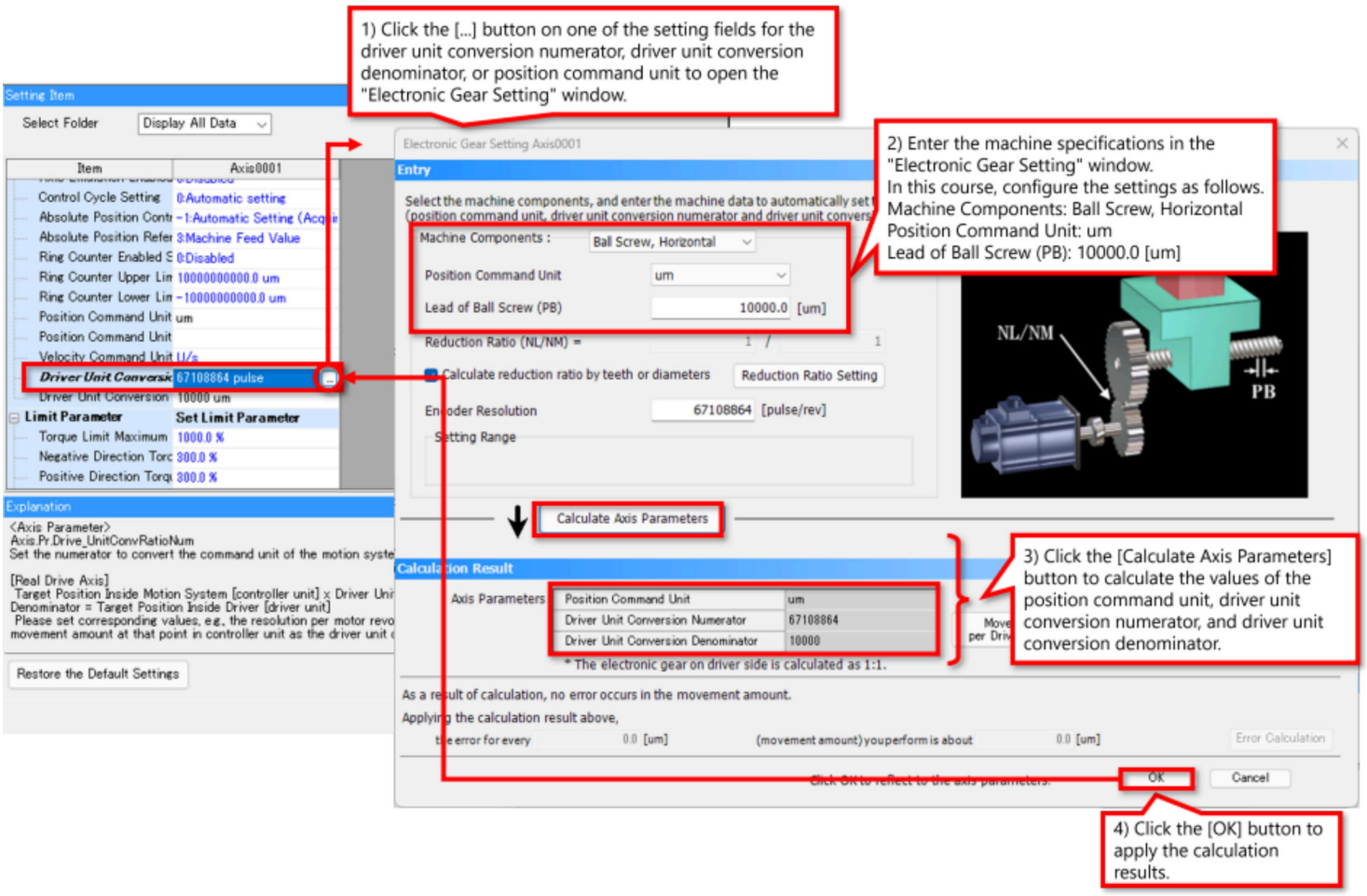
3.4.1 Axis registration

Setting the driver unit conversion

The "Axis Parameter Setting" window appears.

Set the command unit, electronic gear, and limit values.

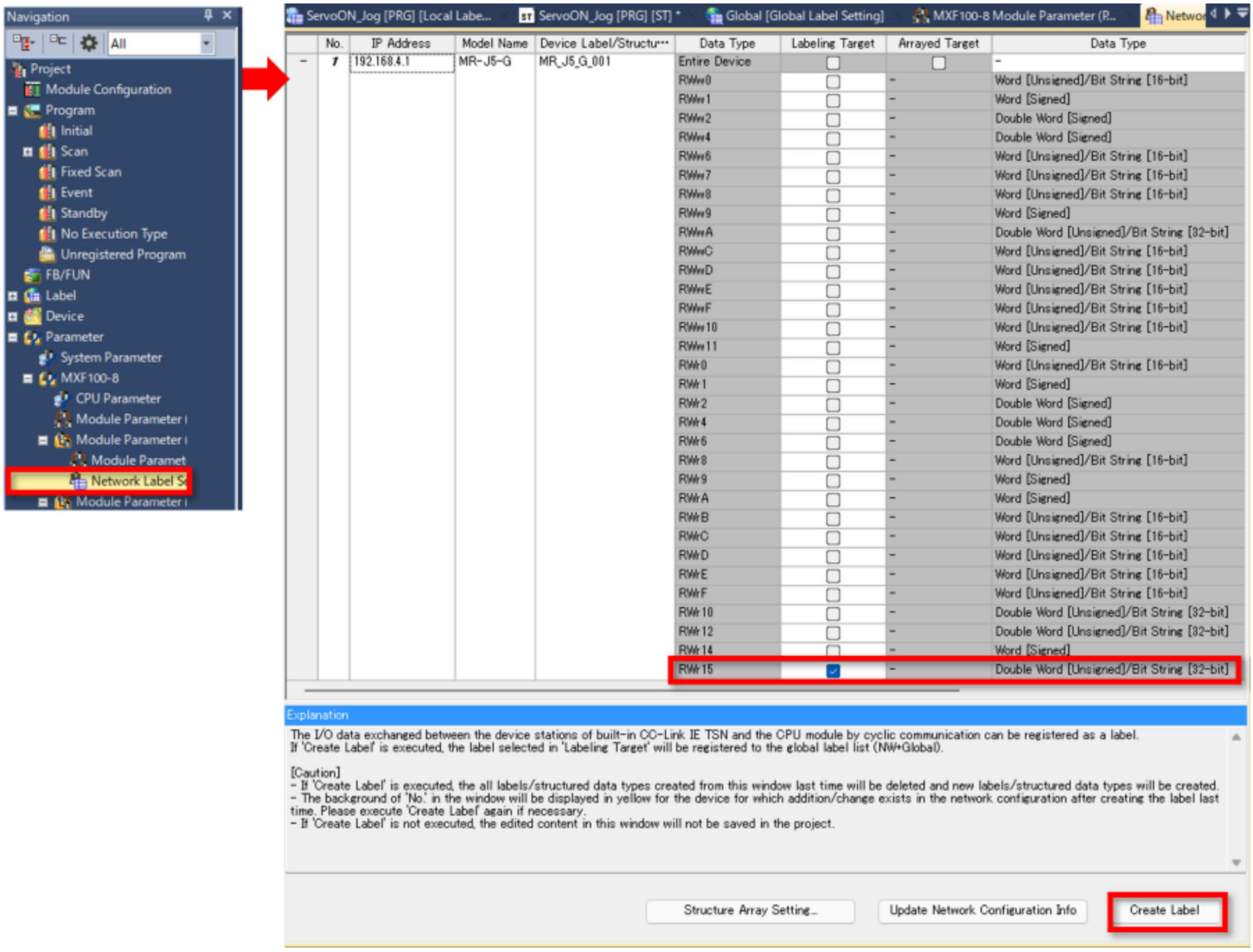
In this course, set the items in the red flames as shown below.



3.4.2 Network label registration

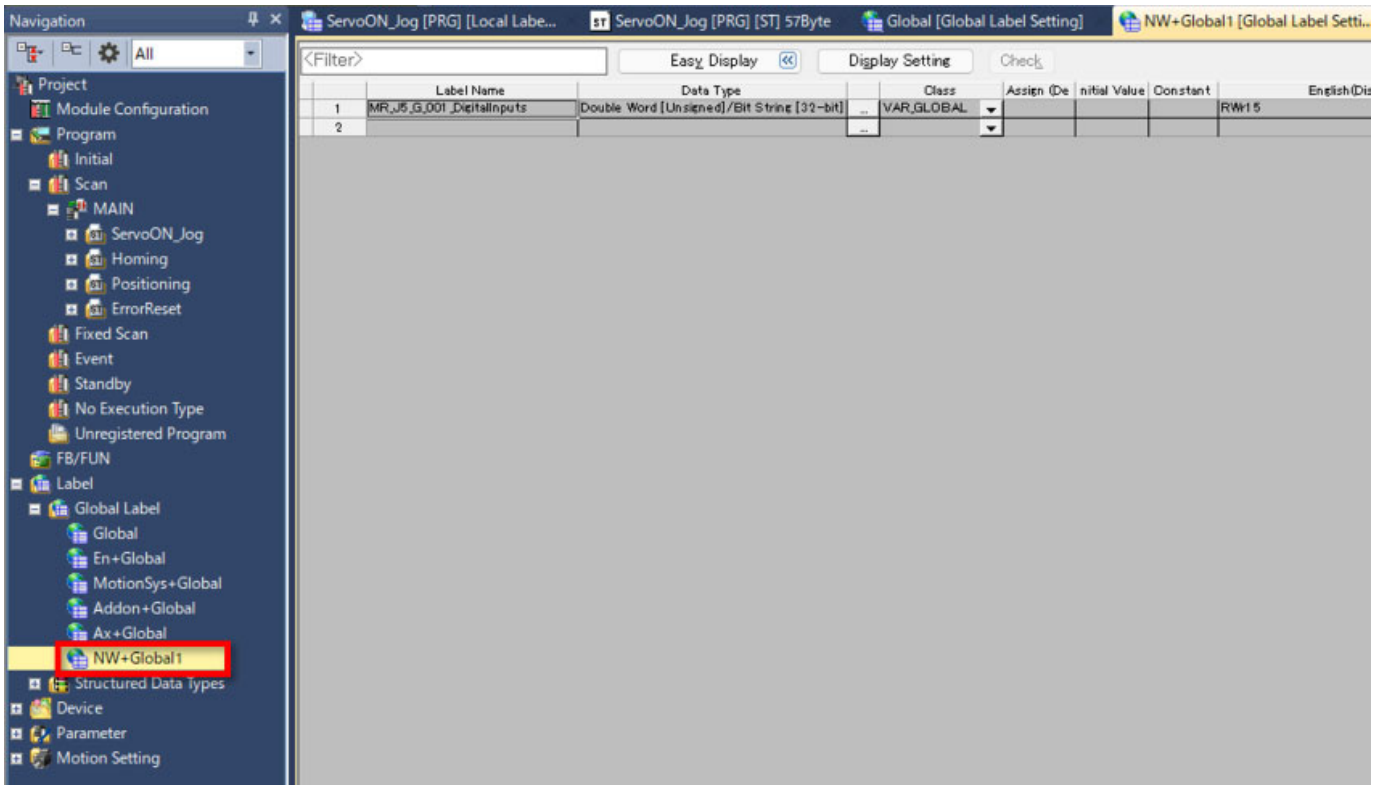
To use the MR-J5-G servo amplifier input signal in the program, register the network label.

1) In the Navigation window, go to [Parameter] → [MXF100-8] → [Module Parameter (Port 1: CC-Link IE TSN)] → [Network Label Setting].
2) In the "Network Label Setting" tab, select the data for labeling. Select [RWr15] of the MR-J5-G in this course.
3) Click the [Create Label] button to create the network label of the selected data.



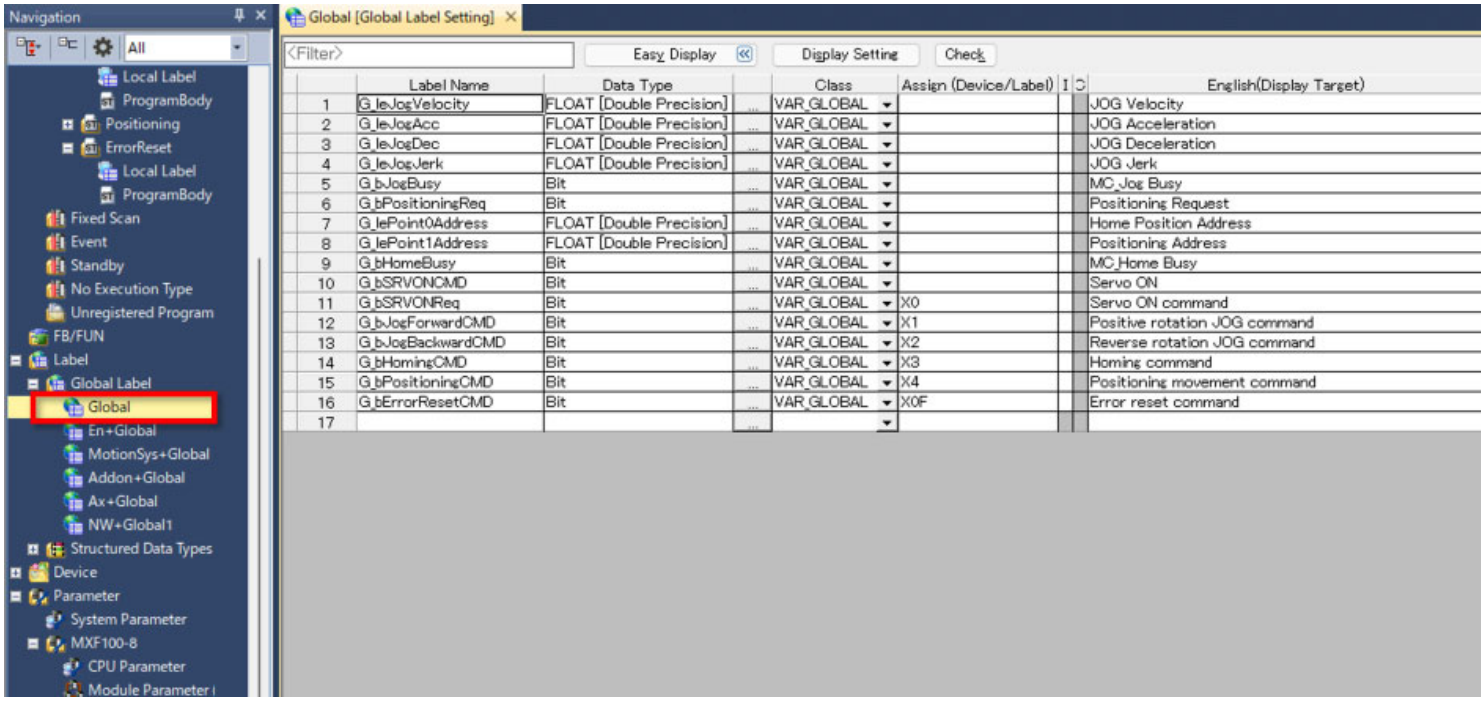
3.4.2 Network label registration

The created network label is registered in [Label] → [Global Label] → [NW+Global1] in the Navigation window.



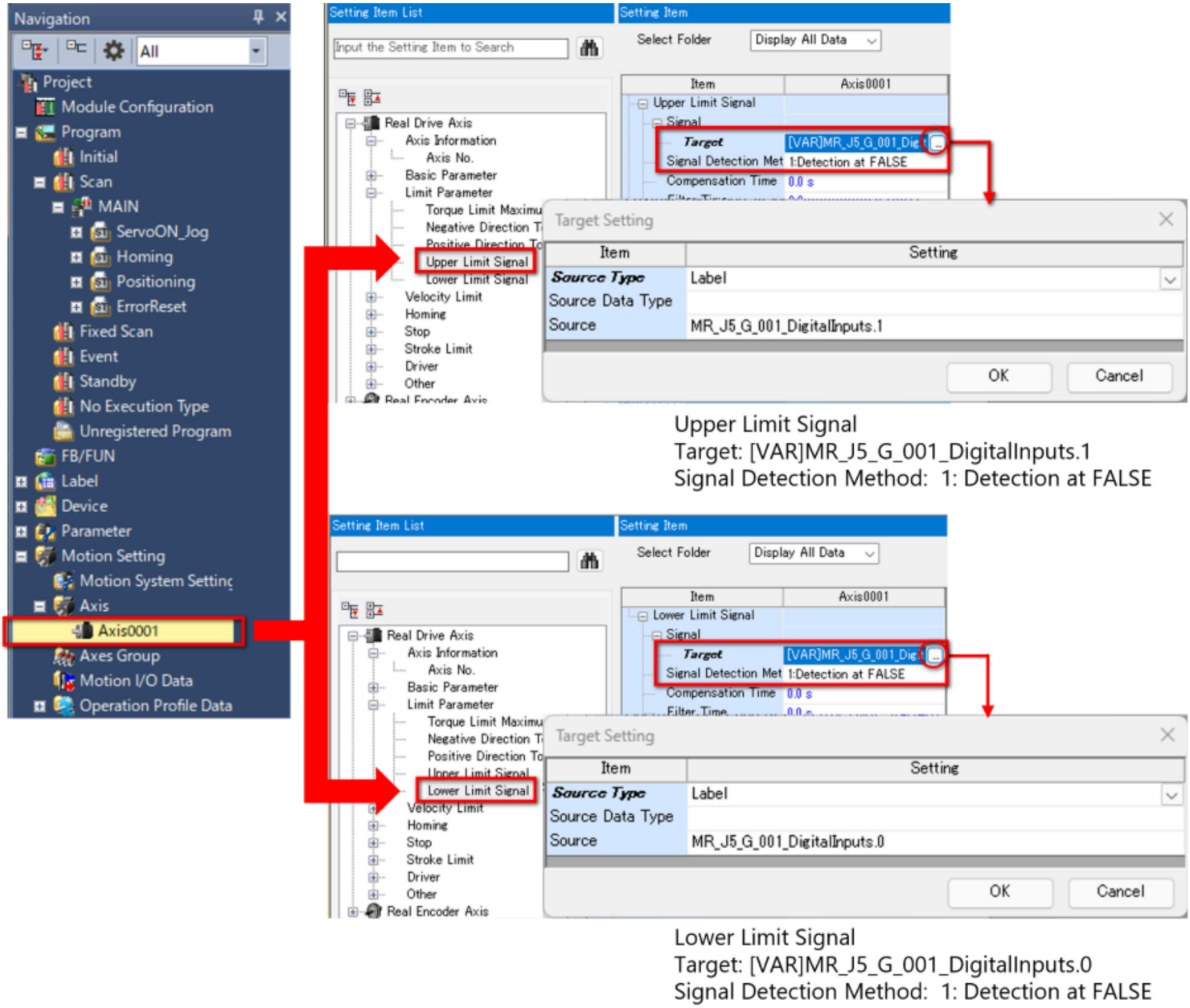
3.4.3 Global label registration

The following is a list of global labels used in this course.
Go to [Label] → [Global Label] → [Global] in the Navigation window and register them.



3.4.4 Limit signal setting

After the network label is created, double-click [Axis0001] in the Navigation window to display the "Axis Parameter Setting" window again.
Set [Upper Limit Signal] and [Lower Limit Signal] as shown below.
In the target field, the icon for displaying the input auxiliary window is displayed.



3.5 Summary

In this chapter, you have learned:

- Creating a New Project
- MELSEC MX Controller Settings
- Module Parameter Settings
- Motion Setting

Important points

Creating a New Project	• Create a new GX Works3 project and a module configuration diagram.
Module Parameter Settings	• In the module parameter (CC-Link IE TSN) window, set the network configuration and remote station parameters. • In the network configuration setting, add a remote station, set the IP address, and perform the PDO mapping. • In the link refresh setting, register a module label.
Motion Setting	• Register an axis. • Create a network label.

Chapter 4 Sample Program and Operation Check

This chapter describes the sample program and how to use Motion Control FB.
The relevant manuals and items are listed as follows.

Initial value setting and servo ON

- MELSEC MX Controller MX-F Model User's Manual
  Servo ON/OFF
- MELSEC MX Controller MX-F Model Programming Manual
  MC_Power (Operation Available)

JOG operation

- MELSEC MX Controller MX-F Model User's Manual
  Single axis manual control
- MELSEC MX Controller MX-F Model Programming Manual
  MCv_Jog (JOG)

Homing

- MELSEC MX Controller MX-F Model User's Manual
  Homing
- MELSEC MX Controller MX-F Model Programming Manual
  MC_Home (OPR)

Positioning control

- MELSEC MX Controller MX-F Model User's Manual
  Single axis positioning control, relative positioning control, and multiple start (buffer mode)
- MELSEC MX Controller MX-F Model Programming Manual
  MC_MoveRelative (Relative Value Positioning) and MC_MoveAbsolute (Absolute Value Positioning)

Error reset

- MELSEC MX Controller MX-F Model User's Manual
  Error and warning reset
- MELSEC MX Controller MX-F Model Programming Manual
  MC_Reset (Axis Error Reset)

Program conversion

- GX Works3 Operating Manual
  Converting Programs

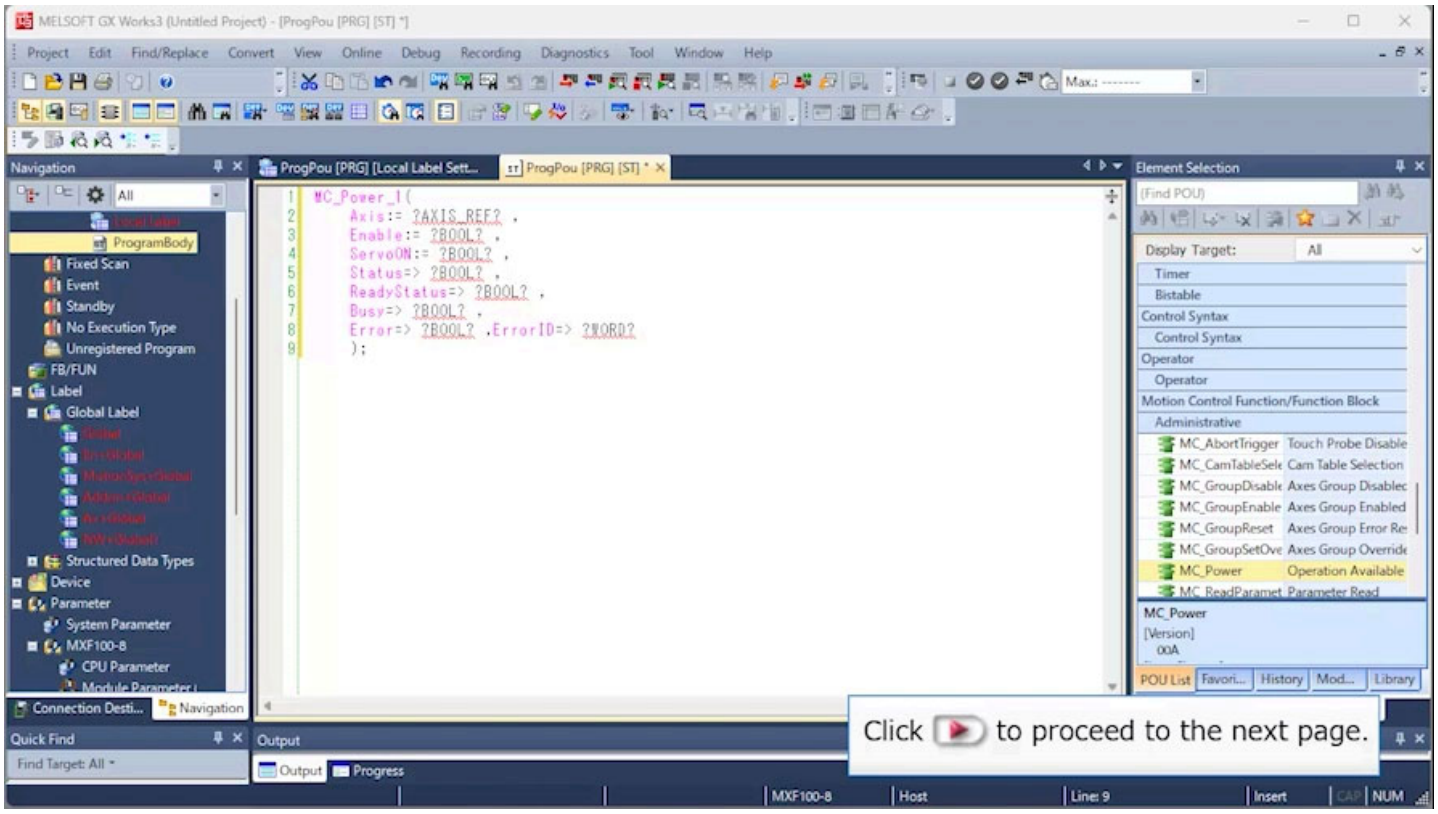
4.1 Sample Program

Use the sample program downloaded in Chapter 3.
Double-click [Program] → [Scan] → [MAIN] → [ServoON_JOG] → [Program Body] in the project tree to open the program.

```
//----PLC READY ON----
G_bSRVONCMD := (MotionSystem.Md.Ready AND MotionSystem.Cd.SequenceReady) AND G_bSRVONReq;

//-----Initial Value Setting-----------------------------
G_leJogVelocity     := 20000.0;//20000um/s  = 1200mm/min
G_leJogAcc          := 20000.0;//20000um/s2 = 1200mm/min/s
G_leJogDec          := 20000.0;//20000um/s2 = 1200mm/min/s
G_leJogJerk         := 25000.0;//25000um/s3

G_lePoint0Address   := 0.0;     //0.0mm
G_lePoint1Address   := 150000.0;//150.0000mm

//-----Axis0001 Servo ON----------------------------
MC_Power_1(
Axis        := Axis0001.AxisRef ,
Enable      := TRUE ,
ServoON     := G_bSRVONCMD ,//Servo ON PLC READY ON
Status      => bPowerStatus ,
ReadyStatus => bReadyStatus ,
Busy        => bPowerBusy ,
Error       => bPowerError ,
ErrorID     => uPowerErrorID
);
```

4.1.1 How to use Motion Control FB

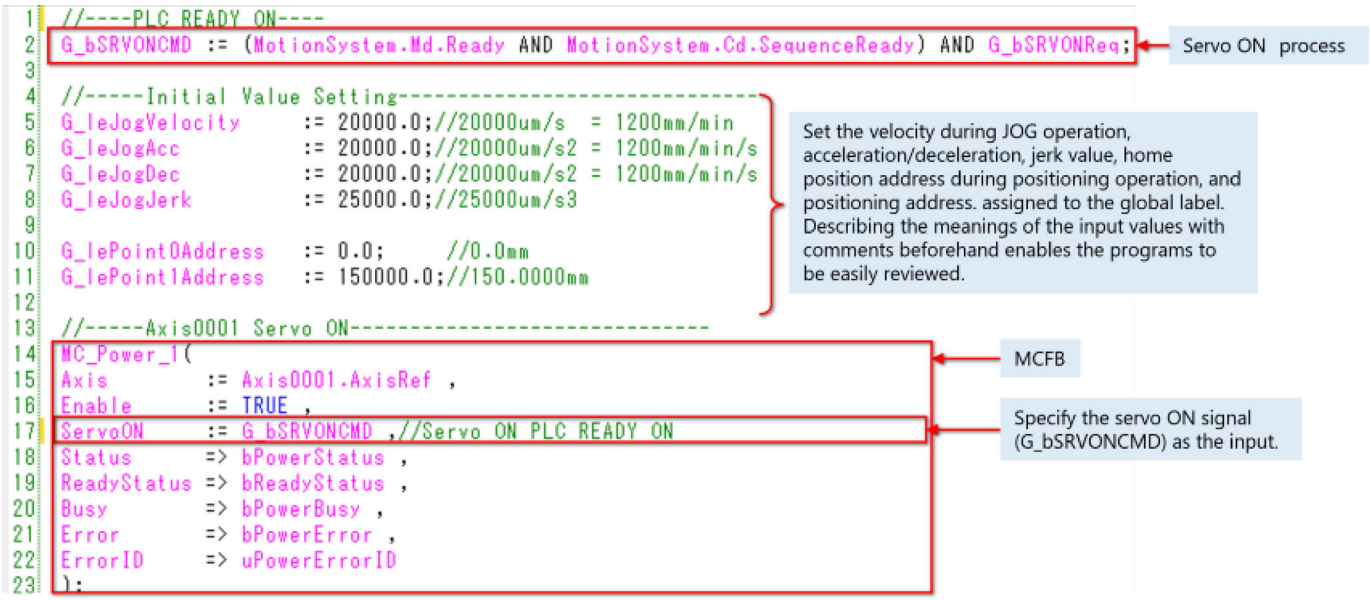
This section describes how to arrange Motion Control FB.



4.1.2 Initial value setting and servo ON

Program name in the sample program: ServoON_JOG

This program sets the initial values of global labels and executes servo ON. Write the servo ON process at the beginning.
When the servo ON request signal X0 (G_bSRVONReq) of the MELSEC MX Controller turns on while PLC READY (MotionSystem.Cd.SequenceReady) and Built-in Motion Ready (MotionSystem.Md.Ready) are on, the servo ON signal (G_bSRVONCMD) turns on.
Connect G_bSRVONCMD to Motion Control FB so that the program executes servo ON when X0 turns on.



<MC_Power specifications (excerpt)>

I/O variable name		Variable name	Data type	Description
Input	Enable	Enable	BOOL	While the Enable input is TRUE, the axis control is enabled.
	Servo ON request	ServoON	BOOL	Specifies the signal for the servo ON request.
Output	Operable	Status	BOOL	Indicates the operation ready status.
	Ready ON status	ReadyStatus	BOOL	Indicates the ready ON/OFF status.
	Executing	Busy	BOOL	Becomes TRUE while the FB is being executed.
	Error	Error	BOOL	Becomes TRUE if an error occurs in the FB.
	Error code	ErrorID	WORD (UINT)	Returns the error code generated in the FB.

Tips
If the MELSEC MX Controller manuals are installed in e-Manual Viewer, hover your cursor over a FB name in the engineering software and press F1 to open the FB specifications page.

4.1.3 JOG operation

Program name in the sample program: ServoON_JOG

Use Motion Control FB (MCv_JOG).
Implement the bJogEnable bit as an interlock to prevent MCv_Jog from running during homing or positioning.

<MCv_Jog specifications (excerpt)>

I/O variable name		Variable name	Data type	Description
Input	Positive rotation JOG command	JogForward	BOOL	Executes the positive rotation JOG when TRUE is set.
	Reverse rotation JOG command	JogBackWard	BOOL	Executes the reverse rotation JOG when TRUE is set.
	Target velocity	Velocity	LREAL	Sets the command velocity.
	Acceleration	Acceleration	LREAL	Sets the acceleration.
	Deceleration	Deceleration	LREAL	Sets the deceleration.
	Jerk	Jerk	LREAL	Sets the jerk.
	Options	Options	DWORD (HEX)(Note)	Sets the function option with bit specification. (→Refer to the next page.)
Output	Execution completion	Done	BOOL	Becomes TRUE only for one scan when a deceleration stop is completed by turning off the JOG command.
	Executing	Busy	BOOL	Becomes TRUE while the FB is being executed.
	Controlling	Active	BOOL	Becomes TRUE when the FB is controlling the axis.
	Abortion of execution	CommandAborted	BOOL	Becomes TRUE when the execution is aborted.
	Error	Error	BOOL	Becomes TRUE if an error occurs in the FB.
	Error code	ErrorID	WORD (UINT)	Returns the error code generated in the FB.

(Note) A hexadecimal value is written in the format of "H□" or "16#□".

4.1.3 JOG operation

The following describes the setting values and details of Options for MCv_JOG.

Setting value	Acceleration/deceleration method setting
0h	mcAccDec: Acceleration/deceleration specification method (Jerk acceleration/deceleration method)
1h	mcFixedTime: Acceleration/deceleration time-fixed method (Acceleration/deceleration time constant method)

When 0h: mcAccDec is specified, the acceleration/deceleration method is set to the jerk acceleration/deceleration method.
Set Acceleration and Deceleration in units of [U/s^2], and Jerk in units of [U/s^3]. (U: Axis command unit)
For details on the jerk acceleration/deceleration, refer to the next page.

When 1h: mcFixedTime is specified, the acceleration/deceleration method is set to the acceleration/deceleration time constant method.
Set Acceleration in units of [s].
Deceleration and Jerk are not used.

The following figures show the velocity waveforms during jerk acceleration/deceleration.
The sum of the time required to reach the target acceleration and the time from the target acceleration to the acceleration 0 at the end of acceleration is called the jerk applying time. The ratio of jerk applying time to acceleration (deceleration) time is called the jerk applying ratio.

The following figures show the velocity waveforms and acceleration waveforms at a time of acceleration when the command velocity and command acceleration are constant and the jerk is changed.
The larger the jerk value becomes, the smaller the jerk applying ratio becomes, and the velocity pattern changes to the trapezoidal acceleration/deceleration.
Additionally, the acceleration time and deceleration time will be shorter.

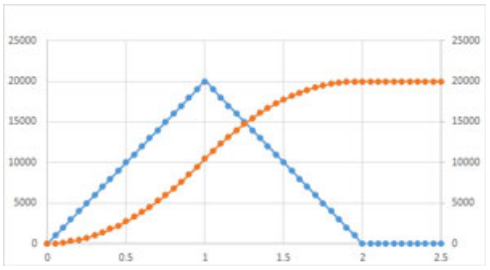
Command velocity: 20000 [μm/s] = 1200 [mm/min]

Command acceleration: 20000 [μm/s^2] = 1200 [mm/min/s]

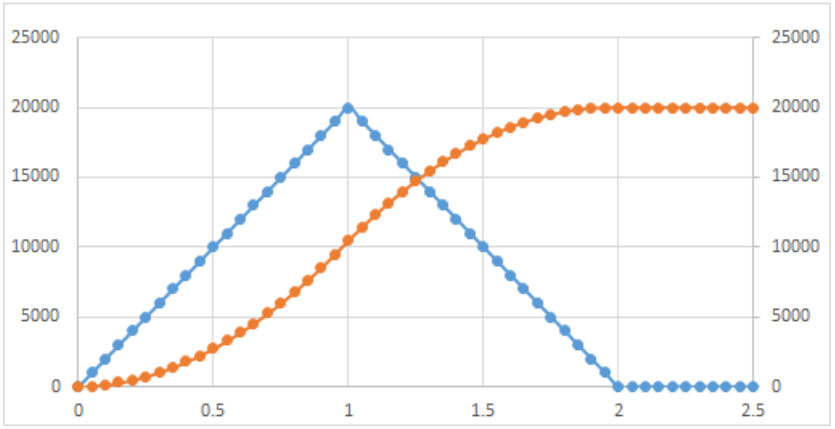
Blue line: Acceleration [μm/s^2] Vertical axis on the left

Orange line: Velocity [μm/s] Vertical axis on the right

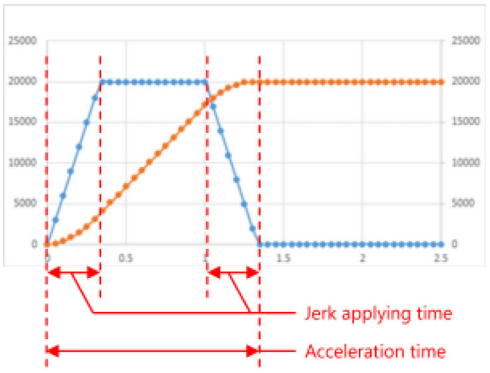
J = 20000 [μm/s^3]
Jerk applying ratio: 100%



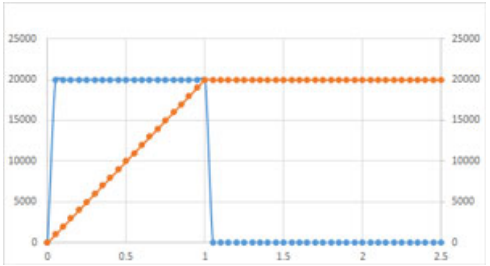
J= 20000[μm/s³] Jerk applying ratio: 100%



J = 60000 [μm/s^3]
Jerk applying ratio: 50%



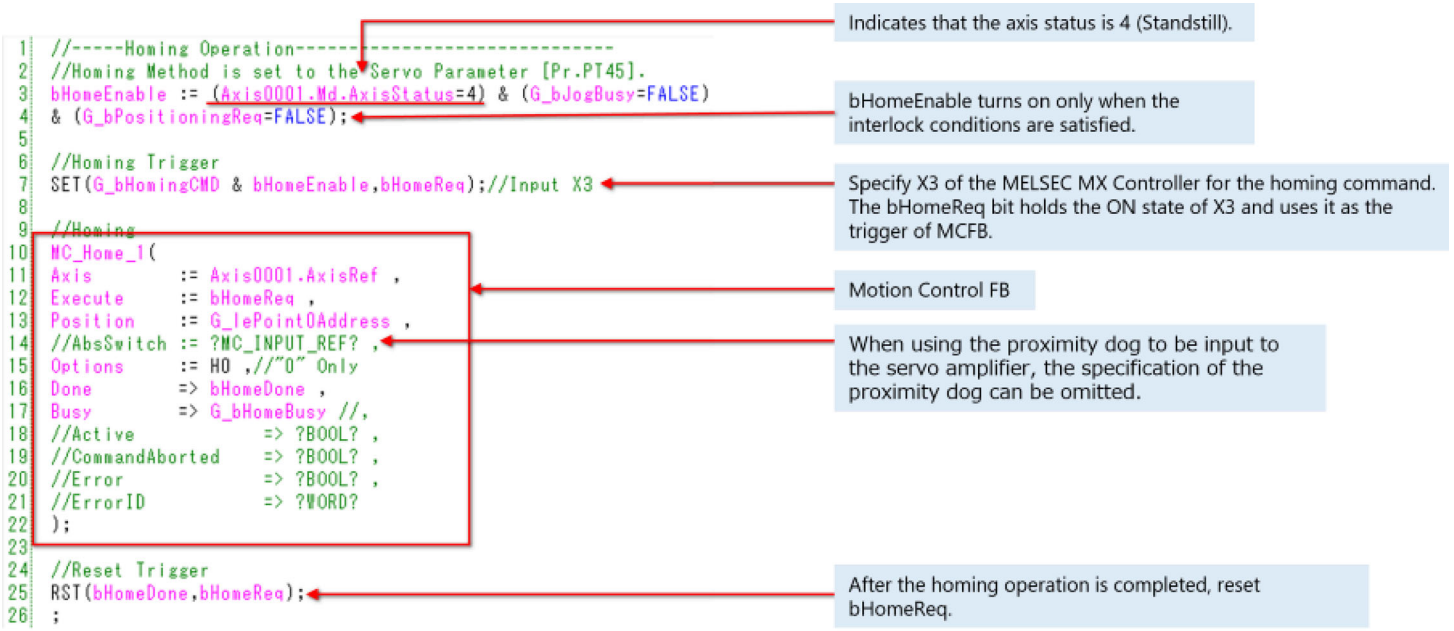
When J = 0 [μm/s^3] is set, the velocity pattern changes to the trapezoidal acceleration/deceleration.



4.1.4 Homing

Program name in the sample program: Homing

Use Motion Control FB (MC_Home).
Implement the bHomeEnable bit as an interlock to prevent MC_Home from running when the servo cannot be started, such as when the servo is off or when an error occurs, or during JOG operation or positioning.
Set the homing method using the parameter [Pr.PT45] of the MR-J5-G servo amplifier.




```
1  //-----Homing Operation-----------------------------
2  //Homing Method is set to the Servo Parameter [Pr.PT45].
3  bHomeEnable := (Axis0001.Md.AxisStatus=4) & (G_bJogBusy=FALSE)
4  & (G_bPositioningReq=FALSE);
5
6  //Homing Trigger
7  SET(G_bHomingCMD & bHomeEnable,bHomeReq);//Input X3
8
9  //Homing
10 MC_Home_1(
11 Axis        := Axis0001.AxisRef ,
12 Execute     := bHomeReq ,
13 Position    := G_lePointOAddress ,
14 //AbsSwitch := ?MC_INPUT_REF? ,
15 Options     := H0 ,//"0" Only
16 Done        => bHomeDone ,
17 Busy        => G_bHomeBusy //,
18 //Active            => ?BOOL? ,
19 //CommandAborted    => ?BOOL? ,
20 //Error             => ?BOOL? ,
21 //ErrorID           => ?WORD?
22 );
23
24 //Reset Trigger
25 RST(bHomeDone,bHomeReq);
26 ;
```

<MC_Home specifications (excerpt)>

I/O variable name		Variable name	Data type	Description
Input	Execution command	Execute	BOOL	Executes homing when TRUE is set.
	Target position	Position	LREAL	Specifies the home position address.
	Home position switch	AbsSwitch	MC_INPUT_REF	Specifies the proximity dog signal.
	Options	Options	DWORD (HEX)	Set "0".
Output	Execution completion	Done	BOOL	Becomes TRUE when homing is completed.
	Executing	Busy	BOOL	Becomes TRUE while the FB is being executed.
	Controlling	Active	BOOL	Becomes TRUE when the FB is controlling the axis.
	Abortion of execution	CommandAborted	BOOL	Becomes TRUE when the execution is aborted.
	Error	Error	BOOL	Becomes TRUE if an error occurs in the FB.
	Error code	ErrorID	WORD (UINT)	Returns the error code generated in the FB.

4.1.5 Positioning control

Program name in the sample program: Positioning

Use Motion Control FBs (MC_MoveRelative and MC_MoveAbsolute).
Implement the bMoveEnable bit as an interlock to prevent MC_MoveRelative and MC_MoveAbsolute from running when the servo cannot be started, such as when the servo is off or when an error occurs, when homing is not completed, or during JOG operation or homing.

```
//-----Initial Value Setting---------------------------------
lePosVelocity   := 20000.0 ;//20000um/s  = 1200mm/min
lePosAcc:= 20000.0 ;//20000um/s2 = 1200mm/min/s
lePosDec:= 20000.0 ;//20000um/s2 = 1200mm/min/s
lePosJerk:= 25000.0 ;//25000um/s3

//-----Positioning Operation---------------------------------
bMoveEnable := (Axis0001.Md.AxisStatus=4) & (Axis0001.Md.Homing_Request=FALSE)
& (G_bJogBusy=FALSE) & (G_bHomeBusy=FALSE);

//Start Trigger
SET(G_bPositioningCMD & bMoveEnable,G_bPositioningReq); //Input X4

//PTP1(Move Relative)
MC_MoveRelative_1(
Axis            := Axis0001.AxisRef ,
Execute         := G_bPositioningReq ,
ContinuousUpdate:= FALSE ,
Distance        := G_lePointlAddress ,
Velocity        := lePosVelocity ,
Acceleration    := lePosAcc ,
Deceleration    := lePosDec ,
Jerk            := lePosJerk ,
BufferMode      := 0 ,//0:mcAborting
Options         := H0 ,//0:mcAccDec
Done=> bMove1Done ,
Busy=> bMove1Busy //,
//Active            => ?BOOL? ,
//CommandAborted    => ?BOOL? ,
//Error             => ?BOOL? ,
//ErrorID           => ?WORD?
);

//Dwell
TON_1(
IN      := bMove1Done ,
PT      := T#500ms ,
Q       => bMove1Dwell //,
//ET    => ?TIME?
);

//PTP2(Move Absolute)
MC_MoveAbsolute_1(
Axis          := Axis0001.AxisRef ,
Execute       := bMove1Dwell ,
ContinuousUpdate:= FALSE ,
Position      := G_lePoint0Address ,
Velocity      := lePosVelocity ,
Acceleration:= lePosAcc ,
Deceleration:= lePosDec ,
Jerk          := lePosJerk ,
Direction     := 3 ,//3:mcShortestWay
BufferMode    := 0 ,//0:mcAborting
Options       := H0 ,//0:mcAccDec
Done          => bMove2Done ,
              => bMove2Busy //,
              tive            => ?BOOL? ,
//CommandAborted    => ?BOOL? ,
//Error             => ?BOOL? ,
//ErrorID           => ?WORD?
);

//Dwell
TON_2(
IN      := bMove2Done ,
PT      := T#500ms ,
Q       => bMove2Dwell //,
//ET        => ?TIME?
);

//Reset Trigger
RST(bMove2Dwell,G_bPositioningReq);
```

Going

Returning

Set the velocity, acceleration/deceleration, and jerk value during positioning operation.

bMoveEnable turns on only when the interlock conditions are satisfied.

Specify X4 of the MELSEC MX Controller for the positioning start command.
The G_bPositioningReq bit holds the ON state of X4 and uses it as the trigger of MCFB.

Motion Control FB

Standard FB for dwell (On-delay timer)

Motion Control FB

Standard FB for dwell (On-delay timer)

After the reciprocating operation is completed, reset G_bPositioningReq.

The following describes the I/O variables of MC_MoveRelative.

```
//PTP1(Move Relative)
MC_MoveRelative_1(
Axis             := Axis0001.AxisRef ,
Execute          := G_bPositioningReq ,
ContinuousUpdate:= FALSE ,
Distance         := G_lePoint1Address ,
Velocity         := lePosVelocity ,
Acceleration     := lePosAcc ,
Deceleration     := lePosDec ,
Jerk             := lePosJerk ,
BufferMode       := 0 ,//0:mcAborting
Options          := H0 ,//0:mcAccDec
Done=> bMove1Done ,
Busy=> bMove1Busy //,
//Active             => ?BOOL? ,
//CommandAborted     => ?BOOL? ,
//Error              => ?BOOL? ,
//ErrorID            => ?WORD?
);

```

<MC_MoveRelative specifications (excerpt)>

I/O variable name		Variable name	Data type	Description
Input	Execution command	Execute	BOOL	Executes positioning when TRUE is set.
	Continuous update	ContinuousUpdate	BOOL	The movement amount, velocity, acceleration, and deceleration can be continuously changed while TRUE is set.
	Movement amount	Distance	LREAL	Sets the relative position from the current position at start to the end point according to the axis unit.
	Velocity	Velocity	LREAL	Sets the velocity according to the axis unit.
	Acceleration	Acceleration	LREAL	Sets the acceleration according to the axis unit.
	Deceleration	Deceleration	LREAL	Sets the deceleration according to the axis unit.
	Jerk	Jerk	LREAL	Sets the jerk according to the axis unit.
	Buffer mode	BufferMode	MC_BUFFER_MODE	Selects the buffer mode. →Refer to Section 4.1.5-4.
	Options	Options	DWORD (HEX)	Selects the function option. →Refer to Section 4.1.5-6.
Output	Execution completion	Done	BOOL	Becomes TRUE when positioning is completed.
	Executing	Busy	BOOL	Becomes TRUE while the FB is being executed.
	Controlling	Active	BOOL	Becomes TRUE when the FB is controlling the axis.
	Abortion of execution	CommandAborted	BOOL	Becomes TRUE when the execution is aborted.
	Error	Error	BOOL	Becomes TRUE if an error occurs in the FB.
	Error code	ErrorID	WORD (UINT)	Returns the error code generated in the FB.

4.1.5 Positioning control

The following describes the I/O variables of MC_MoveRelative.

```
//PTP2(Move Absolute)
MC_MoveAbsolute_1(
Axis         := Axis0001.AxisRef ,
Execute      := bMove1Dwell ,
ContinuousUpdate:= FALSE ,
Position     := G_lePoint0Address ,
Velocity     := lePosVelocity ,
Acceleration:= lePosAcc ,
Deceleration:= lePosDec ,
Jerk         := lePosJerk ,
Direction    := 3 ,//3:mcShortestWay
BufferMode   := 0 ,//0:mcAborting
Options      := H0 ,//0:mcAccDec
Done         => bMove2Done ,
Busy         => bMove2Busy //,
//Active             => ?BOOL? ,
//CommandAborted     => ?BOOL? ,
//Error              => ?BOOL? ,
//ErrorID            => ?WORD?
);
```

<MC_MoveRelative specifications (excerpt)>

I/O variable name		Variable name	Data type	Description
Input	Execution command	Execute	BOOL	Executes positioning when TRUE is set.
	Continuous update	ContinuousUpdate	BOOL	The movement amount, velocity, acceleration, and deceleration can be continuously changed while TRUE is set.
	Target position	Position	LREAL	Sets the target position of the absolute position according to the axis unit.
	Velocity	Velocity	LREAL	Sets the velocity according to the axis unit.
	Acceleration	Acceleration	LREAL	Sets the acceleration according to the axis unit.
	Deceleration	Deceleration	LREAL	Sets the deceleration according to the axis unit.
	Jerk	Jerk	LREAL	Sets the jerk according to the axis unit.
	Direction selection	Direction	MC_DIRECTION	Selects the movement direction. →Refer to Section 4.1.5-5.
	Buffer mode	BufferMode	MC_BUFFER_MODE	Selects the buffer mode. →Refer to Section 4.1.5-4.
	Options	Options	DWORD (HEX)	Selects the function option. →Refer to Section 4.1.5-6.
Output	Execution completion	Done	BOOL	Becomes TRUE when positioning is completed.
	Executing	Busy	BOOL	Becomes TRUE while the FB is being executed.
	Controlling	Active	BOOL	Becomes TRUE when the FB is controlling the axis.
	Abortion of execution	CommandAborted	BOOL	Becomes TRUE when the execution is aborted.
	Error	Error	BOOL	Becomes TRUE if an error occurs in the FB.
	Error code	ErrorID	WORD (UINT)	Returns the error code generated in the FB.

4.1.5 Positioning control

The following describes the setting values and details of Buffer mode for MC_MoveAbsolute and MC_MoveRelative.

Setting value	Buffer mode type	Description
0: mcAborting	Aborting	Aborts (cancels) the FB being executed and executes the next FB immediately.
1: mcBuffered	Buffered	Buffers the next FB on the FB being executed. If the FB being executed has already been buffered, the next FB is buffered on the previous FB. (Up to 2.) When the FB being executed is completed, the buffering FB is executed sequentially.
2: mcBlendingLow	BlendingLow	The lower target velocity between the FB being executed and the buffering FB is used as the switching speed.
3: mcBlendingPrevious	BlendingPrevious	The target velocity of the FB being executed is used as the switching speed.
4: mcBlendingNext	BlendingNext	The target velocity of the buffering FB is used as the switching speed.
5: mcBlendingHigh	BlendingHigh	The higher target velocity between the FB being executed and the buffering FB is used as the switching speed.

The buffer mode is a function that starts multiple Motion Control FBs simultaneously and performs positioning continuously. For details, refer to the MELSEC iQ-R Series Motion Module Basics (RD78G(H)/Positioning Control), which is a course of an online training system, and relevant manuals.

4.1.5 Positioning control

The following describes the setting values and details of Direction selection for MC_MoveAbsolute.

Ignore this setting when the software stroke limit is valid. Perform positioning control in a direction in which the area outside the software stroke limit range is not crossed over. However, when both directions do not cross over the area outside the software stroke limit range, positioning control is performed in the direction closer to the target position (the one with the smaller absolute movement distance) based on the current position. If the distance is the same between the positive direction and the negative direction, the control is performed in the current direction.
When the software stroke limit is invalid, the movement direction from the current position to the target position can be selected from the positive direction, negative direction, and shortest path.

Setting value	Direction selection	Description
1: mcPositiveDirection	Positive direction	Positioning is performed in the positive direction (address increase) from the current position to the target position.
2: mcNegativeDirection	Negative direction	Positioning is performed in the negative direction (address decrease) from the current position to the target position.
3: mcShortestWay	Shortest path	Positioning control is performed in the direction closer to the target position (the one with the smaller absolute movement distance) based on the current position.

4.1.5 Positioning control

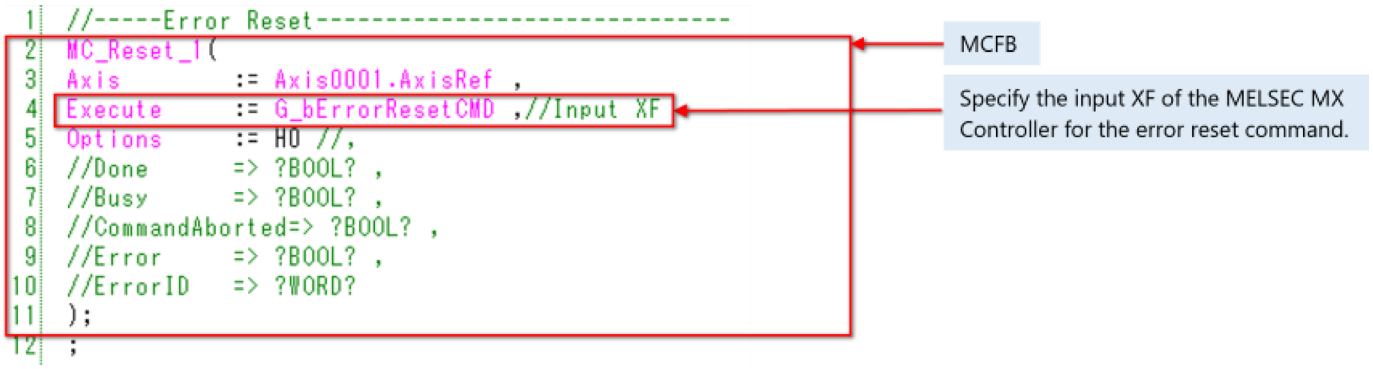
The following describes the setting values and details of Options for MC_MoveAbsolute and MC_MoveRelative.

Bit	Description
0 to 2	Acceleration/deceleration method specification (The contents are the same as MCv_Jog.) 0h: mcAccDec 1h: mcFixedTime
3	Only for MC_MoveRelative Position selection during buffer mode 0: Set position 1: Actual position For MC_MoveAbsolute, specify "0".
4	Reserved (Specify "0".)
5	Reverse rotation permission selection 0: Allow 1: Not allowed
6 to 15	Reserved (Specify "0".)
16	Only for MC_MoveAbsolute Target position specification exceeding the ring counter 0: Not allowed 1: Allow For MC_MoveRelative, specify "0".
17 to 31	Reserved (Specify "0".)

4.1.6 Error reset

Program name in the sample program: ErrorReset

Use Motion Control FB (MC_Reset).



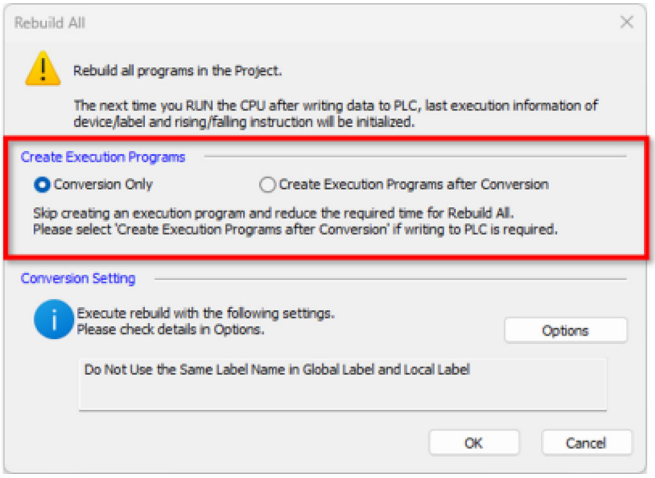
<MC_Reset specifications (excerpt)>

I/O variable name		Variable name	Data type	Description
Input	Execution command	Execute	BOOL	Executes the error reset when TRUE is set.
	Options	Options	DWORD (HEX)	Specify "0".
Output	Execution completion	Done	BOOL	Indicates that reset has been completed.
	Executing	Busy	BOOL	Becomes TRUE while the FB is being executed.
	Abortion of execution	CommandAborted	BOOL	Indicates that the command has been aborted due to timeout. Becomes FALSE when the Execution command (Execute) becomes FALSE.
	Error	Error	BOOL	Becomes TRUE if an error occurs in the FB.
	Error code	ErrorID	WORD (UINT)	Returns the error code generated in the FB.

4.1.7 Program conversion

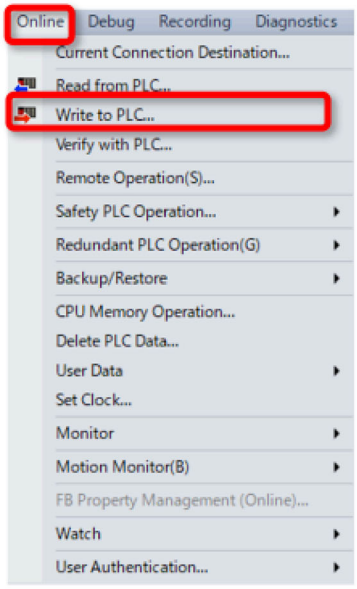
(1) This section describes how to convert the created program into executable code.
Data names displayed in red in the Navigation window have not yet been converted and must be converted.

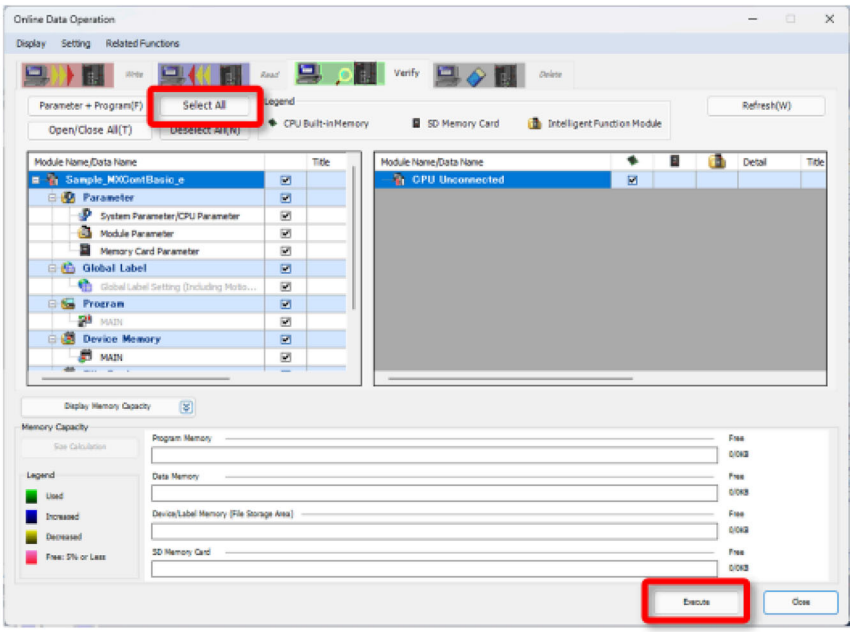
(2) Creating an execution program
When converting any or all program, whether to create an execution program can be selected.
To write data to the MELSEC MX Controller, select the checkbox of "Create Execution Programs after Conversion".
When selecting the checkbox of "Conversion Only", the required time for the conversion can be shortened.



(1) MELSEC MX Controller program

1) Convert all the MELSEC MX Controller programs.
2) Set the MELSEC MX Controller to the STOP state.
3) Select [Online] → [Write to PLC], and click [Select All] on the Write tab in the "Online Data Operation" window.
4) Click the [Execute] button to write data.





The operation check is complete.
Go to the next page

4.4 Summary

In this chapter, you have learned:

- MELSEC MX Controller Programs
- Writing Programs
- Operation Check

Important points

MELSEC MX Controller Programs	• When the servo ON request signal X0 (G_bSRVONReq) of the MELSEC MX Controller turns on while PLC READY (MotionSystem.Cd.SequenceReady) and Built-in Motion Ready (MotionSystem.Md.Ready) are on, the servo ON signal (G_bSRVONCMD) turns on. • Drag and drop Motion Control FB from the Element Selection window. • Use MC_Power for servo ON, MCv_Jog for JOG operation, MC_Home for homing, MC_MoveRelative for relative value positioning, MC_MoveAbsolute for absolute value positioning, and MC_Reset for error reset. • Set the homing method using the parameters of the servo amplifier.
Writing Programs	• Write the programs to the MELSEC MX Controller.
Operation Check	• The sample system operation is checked in the video.

Test Final Test

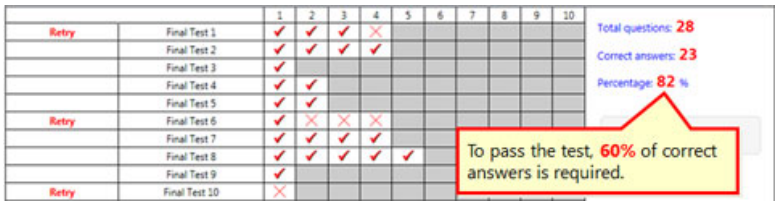
Now that you have completed all of the lessons of the **MELSEC MX Controller Basics (Motion Control Startup)** Course, you are ready to take the final test. If you are unclear on any of the topics covered, please take this opportunity to review those topics.

There are a total of 5 questions (7 items) in this Final Test.

You can take the final test as many times as you like.

Score results

The number of correct answers, the number of questions, the percentage of correct answers, and the pass/fail result will appear on the score page.



Test **Final Test 1**

Select the correct statement(s) below. (Multiple choices)

Q1

- [x] **Velocity waveforms during jerk acceleration/deceleration change smoothly.**
- [ ] **If the jerk value increases, the acceleration/deceleration time becomes longer.**
- [ ] **In the MELSEC MX Controller, programs are created with the FBs created by Mitsubishi Electric.**
- [ ] **Statements must end with a colon ":" in the ST language.**
- [x] **Local labels can be used only in the declared POU.**

Test **Final Test 2**

Select the correct terms for ( ) in the following sentences.

- To perform the test operation, change the (Q1) of the servo amplifier before power-on.
- Check the servo motor rotation direction and machine operation using the test operation function of (Q2).
- Set the (Q3) with the rotary switches of the servo amplifier.

Q1 1: DIP switch

Q2 2: MR Configurator2

Q3 1: IP address

Test **Final Test 3**

Select the correct statement(s) below. (Multiple choices)

Q1

- [ ] **Once the PDO mapping is performed, there is no problem even if the network configuration is changed.**
- [x] **Parameters of the servo amplifier can be transferred from the controller during the initial communication or can be written to each axis using MR Configuretor2.**
- [x] **In the MELSEC MX Controller, motion settings are configured using GX Works3.**

Test **Final Test 4**

Select the correct statement(s) about programs when using the MELSEC MX Controller. (Multiple choices)

Q1

- [x] **Check that Built-in Motion Ready [MotionSystem.Md.Ready] is on.**
- [ ] **The servo ON signal turns on when PLC READY and Built-in Motion Ready are on.**
- [x] **Motion Control FB can be written to the program editor by drag-and-drop action.**
- [ ] **All the I/O signals of Motion Control FB must be set.**

Test **Final Test 5**

Select the correct statement about the homing method setting.

Q1

- Use the input variable "Options" in the MC_Home function block.
- Use axis parameters in the motion setting window.
- Use parameters of the MR-J5-G servo amplifier.

Test **Test Score**

You have completed the Final Test. You results area as follows.
To end the Final Test, proceed to the next page.

		1	2	3	4	5	6	7	8	9	10
	Final Test 1	✓									
	Final Test 2	✓	✓	✓							
	Final Test 3	✓									
	Final Test 4	✓									
	Final Test 5	✓									

Total questions: 7

Correct answers: 7

Percentage: 100 %

Clear

You have completed the "MELSEC MX Controller Basics (Motion Control Startup)" Course.

Thank you for taking this course.

We hope you enjoyed the lessons and the information you acquired in this course is useful for configuring systems in the future.

You can review the course as many times as you want.

Review

Close