OMRON

Sysmac Library

User's Manual for SLMP Communications Library

SYSMAC-XR017

sysmac
always in control

W597-E1-02

NOTE

(1) All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, mechanical, electronic, photocopying, recording, or otherwise, without the prior written permission of OMRON.

(2) No patent liability is assumed with respect to the use of the information contained herein. Moreover, because OMRON is constantly striving to improve its high-quality products, the information contained in this manual is subject to change without notice.

(3) Every precaution has been taken in the preparation of this manual. Nevertheless, OMRON assumes no responsibility for errors or omissions.
Neither is any liability assumed for damages resulting from the use of the information contained in this publication.

Trademarks

- Sysmac and SYSMAC are trademarks or registered trademarks of OMRON Corporation in Japan and other countries for OMRON factory automation products.
- Microsoft, Windows, Windows Vista, Excel, and Visual Basic are either registered trademarks or trademarks of Microsoft Corporation in the United States and other countries.
- EtherCAT® is a patented technology and registered trademark, licensed by Beckhoff Automation GmbH, Germany.
- ODVA, CIP, CompoNet, DeviceNet, and EtherNet/IP are trademarks of ODVA.
- The SD and SDHC logos are trademarks of SD-3C, LLC.

Other company names and product names in this document are the trademarks or registered trademarks of their respective companies.

Copyrights

- Microsoft product screen shots reprinted with permission from Microsoft Corporation.

Introduction

Thank you for purchasing an NJ/NX-series CPU Unit, PC for NY-series production.
This manual contains information that is necessary to use Function block for SLMP Communications Library (hereafter, sometimes abbreviated to FB). Please read this manual and make sure you understand the functionality and performance of the product before you attempt to use it in a control system.
This manual provides function block specifications. It does not describe application restrictions or combination restrictions for Controllers, Units, and components.
Make sure to read the user's manual for each product before use.
Keep this manual in a safe place where it will be available for reference during operation.

Features of the Library

The SLMP Communications Library is a collection of functional objects that uses the SLMP communications protocol for the Sequencer made by Mitsubishi Electric to carry out communications control.
Various FB types are provided for easy incorporation of the OMRON Machine Automation Controller in networks configured by the Mitsubishi Electric Sequencer.
This enables the use of a Machine Automation Controller to monitor systems configured by the Mitsubishi Electric Sequencer and perform controls.

Intended Audience

This manual is intended for the following personnel,
who must also have knowledge of electrical systems (an electrical engineer or the equivalent).

- Personnel in charge of introducing FA systems.
- Personnel in charge of designing FA systems.
- Personnel in charge of installing and maintaining FA systems.
- Personnel in charge of managing FA systems and facilities.

For programming, this manual is intended for personnel who understand the programming language specifications in international standard IEC 61131-3 or Japanese standard JIS B 3503.

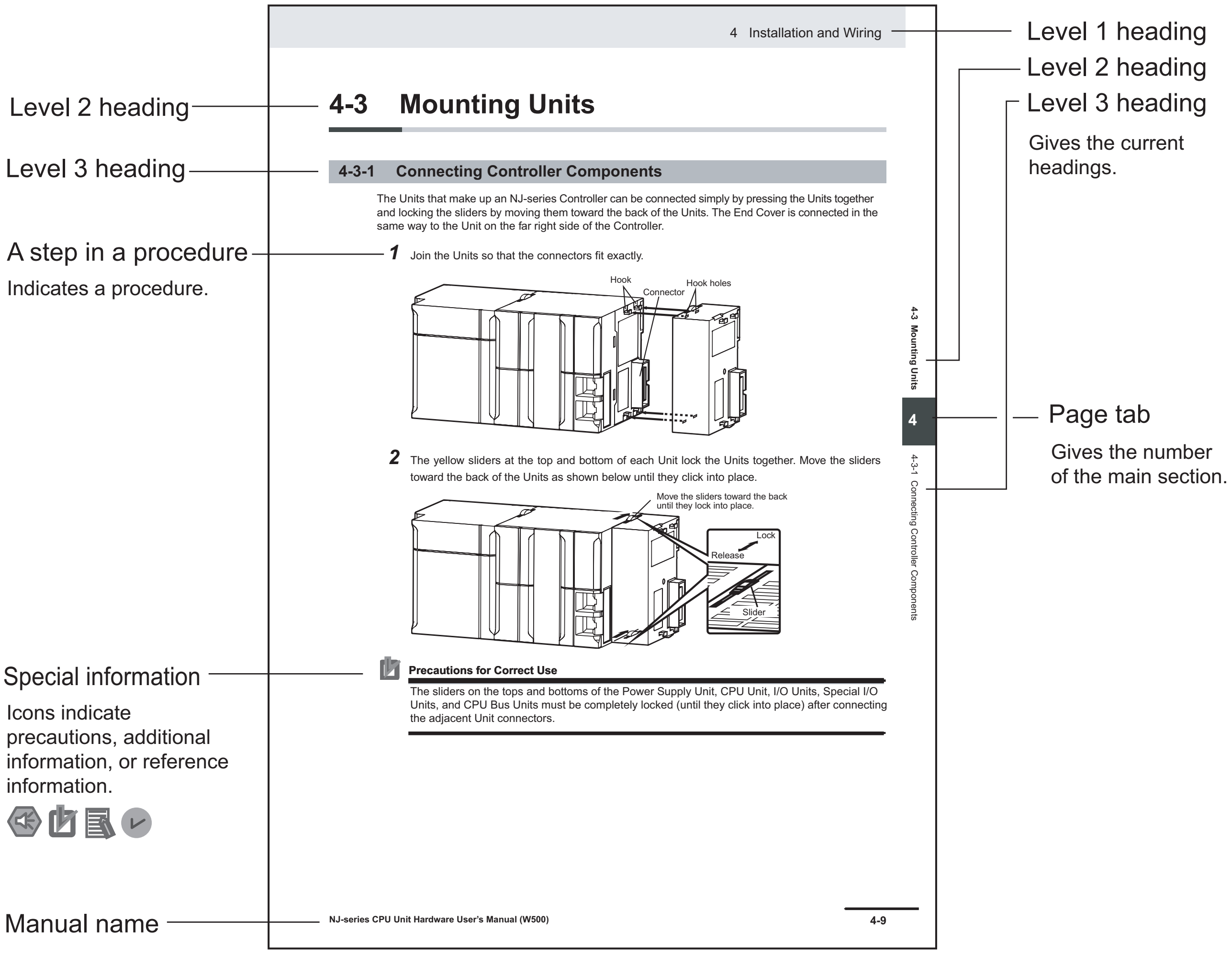
Applicable Products

For the model numbers and versions of an NJ/NX-series CPU Unit, NY-series Industrial PC, and the Sysmac Studio that this library supports, refer to *Sysmac Library Version Information* in the *SYSMAC-XR□□□ Sysmac Library Catalog (Cat. No. P102)*. This catalog can be downloaded from the OMRON website (http://www.ia.omron.com/products/family/3459/download/catalog.html).

Manual Structure

Page Structure

The following page structure is used in this manual.



Level 1 heading
Level 2 heading
Level 3 heading
Gives the current headings.
Level 2 heading
Level 3 heading
A step in a procedure
Indicates a procedure.
Page tab
Gives the number of the main section.
Special information
Icons indicate precautions, additional information, or reference information.
Manual name
Hook
Connector
Hook holes
Move the sliders toward the back until they lock into place.
Lock
Release
Slider
4-3 Mounting Units
4
4-3-1 Connecting Controller Components

4 Installation and Wiring

4-3 Mounting Units

4-3-1 Connecting Controller Components

The Units that make up an NJ-series Controller can be connected simply by pressing the Units together and locking the sliders by moving them toward the back of the Units. The End Cover is connected in the same way to the Unit on the far right side of the Controller.

1 Join the Units so that the connectors fit exactly.

2 The yellow sliders at the top and bottom of each Unit lock the Units together. Move the sliders toward the back of the Units as shown below until they click into place.

Precautions for Correct Use

The sliders on the tops and bottoms of the Power Supply Unit, CPU Unit, I/O Units, Special I/O Units, and CPU Bus Units must be completely locked (until they click into place) after connecting the adjacent Unit connectors.

NJ-series CPU Unit Hardware User's Manual (W500)

4-9

Note This illustration is provided only as a sample. It may not literally appear in this manual.

Special Information

Special information in this manual is classified as follows:

Precautions for Safe Use

Precautions on what to do and what not to do to ensure safe usage of the product.

Precautions for Correct Use

Precautions on what to do and what not to do to ensure proper operation and performance.

Additional Information

Additional information to read as required.
This information is provided to increase understanding and make operation easier.

Version Information

Information on differences in specifications and functionality for CPU Units with different unit versions and for different versions of the industrial-use PC, Sysmac Studio are given.

CONTENTS

Section 3 Common Specifications of Function Blocks

Section 4 Individual Specifications of Function Blocks

Appendix

Index

Terms and Conditions Agreement

Warranty, Limitations of Liability

Warranties

- **Exclusive Warranty**

Omron's exclusive warranty is that the Products will be free from defects in materials and workmanship for a period of twelve months from the date of sale by Omron (or such other period expressed in writing by Omron). Omron disclaims all other warranties, express or implied.

- **Limitations**

OMRON MAKES NO WARRANTY OR REPRESENTATION, EXPRESS OR IMPLIED, ABOUT NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE OF THE PRODUCTS. BUYER ACKNOWLEDGES THAT IT ALONE HAS DETERMINED THAT THE PRODUCTS WILL SUITABLY MEET THE REQUIREMENTS OF THEIR INTENDED USE.

Omron further disclaims all warranties and responsibility of any type for claims or expenses based on infringement by the Products or otherwise of any intellectual property right.

- **Buyer Remedy**

Omron's sole obligation hereunder shall be, at Omron's election, to (i) replace (in the form originally shipped with Buyer responsible for labor charges for removal or replacement thereof) the non-complying Product, (ii) repair the non-complying Product, or (iii) repay or credit Buyer an amount equal to the purchase price of the non-complying Product; provided that in no event shall Omron be responsible for warranty, repair, indemnity or any other claims or expenses regarding the Products unless Omron's analysis confirms that the Products were properly handled, stored, installed and maintained and not subject to contamination, abuse, misuse or inappropriate modification. Return of any Products by Buyer must be approved in writing by Omron before shipment. Omron Companies shall not be liable for the suitability or unsuitability or the results from the use of Products in combination with any electrical or electronic components, circuits, system assemblies or any other materials or substances or environments. Any advice, recommendations or information given orally or in writing, are not to be construed as an amendment or addition to the above warranty.

See http://www.omron.com/global/ or contact your Omron representative for published information.

Limitation on Liability; Etc

OMRON COMPANIES SHALL NOT BE LIABLE FOR SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, LOSS OF PROFITS OR PRODUCTION OR COMMERCIAL LOSS IN ANY

WAY CONNECTED WITH THE PRODUCTS, WHETHER SUCH CLAIM IS BASED IN CONTRACT, WARRANTY, NEGLIGENCE OR STRICT LIABILITY.

Further, in no event shall liability of Omron Companies exceed the individual price of the Product on which liability is asserted.

Application Considerations

Suitability of Use

Omron Companies shall not be responsible for conformity with any standards, codes or regulations which apply to the combination of the Product in the Buyer's application or use of the Product. At Buyer's request, Omron will provide applicable third party certification documents identifying ratings and limitations of use which apply to the Product. This information by itself is not sufficient for a complete determination of the suitability of the Product in combination with the end product, machine, system, or other application or use. Buyer shall be solely responsible for determining appropriateness of the particular Product with respect to Buyer's application, product or system. Buyer shall take application responsibility in all cases.

NEVER USE THE PRODUCT FOR AN APPLICATION INVOLVING SERIOUS RISK TO LIFE OR PROPERTY OR IN LARGE QUANTITIES WITHOUT ENSURING THAT THE SYSTEM AS A WHOLE HAS BEEN DESIGNED TO ADDRESS THE RISKS, AND THAT THE OMRON PRODUCT(S) IS PROPERLY RATED AND INSTALLED FOR THE INTENDED USE WITHIN THE OVERALL EQUIPMENT OR SYSTEM.

Programmable Products

Omron Companies shall not be responsible for the user's programming of a programmable Product, or any consequence thereof.

Disclaimers

Performance Data

Data presented in Omron Company websites, catalogs and other materials is provided as a guide for the user in determining suitability and does not constitute a warranty. It may represent the result of Omron's test conditions, and the user must correlate it to actual application requirements. Actual performance is subject to the Omron's Warranty and Limitations of Liability.

Change in Specifications

Product specifications and accessories may be changed at any time based on improvements and other reasons. It is our practice to change part numbers when published ratings or features are changed, or when significant construction changes are made. However, some specifications of the Product may be changed without any notice. When in doubt, special part numbers may be assigned to fix or establish key specifications for your application. Please consult with your Omron's representative at any time to confirm actual specifications of purchased Product.

Errors and Omissions

Information presented by Omron Companies has been checked and is believed to be accurate; however, no responsibility is assumed for clerical, typographical or proofreading errors or omissions.

Safety Precautions

Definition of Precautionary Information

The following notation is used in this user's manual to provide precautions required to ensure safe usage of this library on the NJ/NX-series CPU Unit, PC for NY-series production.

The safety precautions that are provided are extremely important for safety. Always read and heed the information provided in all safety precautions.

The following notation is used.

⚠ **WARNING**	Indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury. Additionally, there may be severe property damage.
⚠ **Caution**	Indicates a potentially hazardous situation which, if not avoided, may result in minor or moderate injury, or property damage.

Symbols

🛇	The circle and slash symbol indicates operations that you must not do. The specific operation is shown in the circle and explained in text. This example indicates that disassembly is prohibited.
⚡	The triangle symbol indicates precautions (including warnings). The specific operation is shown in the triangle and explained in text. This example indicates a precaution for electric shock.
⚠	The triangle symbol indicates precautions (including warnings). The specific operation is shown in the triangle and explained in text. This example indicates a general precaution.
❗	The filled circle symbol indicates operations that you must do. The specific operation is shown in the circle and explained in text. This example shows a general precaution for something that you must do.

WARNING

⚠ Caution

Read all related manuals carefully before you use this library.	⚠
Emergency stop circuits, interlock circuits, limit circuits, and similar safety measures must be provided in external control circuits.	⚠
Check the user program, data, and parameter settings for proper execution before you use them for actual operation.	⚠
The Sysmac Library and manuals are assumed to be used by personnel that is given in Intended Audience in this manual. Otherwise, do not use them.	⚠
Perform the test run by holding an emergency stop switch in hand or otherwise prepare for rapid motor operation in an application to control the motor. Also perform the test run by using parameters for which the motor does not rapidly accelerate or decelerate before you gradually adjust the parameters.	⚠
In heating or cooling applications, perform the test run by using parameters for which rapid temperature changes will not occur before you gradually adjust the parameters.	⚠
You must confirm that the user program and parameter values are appropriate to the specifications and operation methods of the devices.	⚠
The sample programming shows only the portion of a program that uses the function or function block from the library.	⚠
When using actual devices, also program safety circuits, device interlocks, I/O with other devices, and other control procedures.	⚠
Understand the contents of sample programming before you use the sample programming and create the user program.	⚠

Precautions for Correct Use

Using the Library

- When you use the library, functions or function blocks that are not described in the library manual may be displayed on the Sysmac Studio. Do not use functions or function blocks that are not described in the manual.
- You cannot change the source code of the functions or function blocks that are provided in the Sysmac Library.
- The multi-execution (buffer mode) cannot be performed in the Sysmac Library.

Using Sample Programming

- Create a user program that will produce the intended device operation.
- Check the user program for proper execution before you use it for actual operation.

Operation

- Specify the input parameter values within the valid range.
- In a function or function block with an Enabled output variable, if the value of Enabled is FALSE, do not use the processing result of the function or function block as a command value to the control target.
- In the function block with Execute, do not perform re-execution by the same instance. The output value of the function block will return to the default value.

Related Manuals

The following are the manuals related to this manual. Use these manuals for reference.

Manual name	Man. No.	Model	Application	Description
NX-series CPU Unit Hardware User's Manual	W535	NX701-□□□□	Learning the basic specifications of the NX701 CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NX701 system is provided along with the following information on the CPU Unit. • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection
NX-series NX102 CPU Unit Hardware User's Manual	W593	NX102-□□□□	Learning the basic specifications of the NX102 CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NX102 system is provided along with the following information on the CPU Unit. • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection
NX-series NX1P2 CPU Unit Hardware User's Manual	W578	NX1P2-□□□□	Learning the basic specifications of the NX1P2 CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NX1P2 system is provided along with the following information on the CPU Unit. • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection

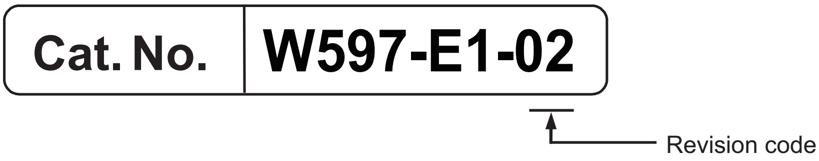
Manual name	Man. No.	Model	Application	Description
NJ-series CPU Unit Hardware User's Manual	W500	NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning the basic specifications of the NJ-series CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NJ-series system is provided along with the following information on the CPU Unit. • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection
NY-series IPC Machine Controller Industrial Panel PC Hardware User's Manual	W557	NY532-□□□□	Learning the basic specifications of the NY-series Industrial Panel PCs, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NY-series system is provided along with the following information on the Industrial Panel PC. • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection
NY-series IPC Machine Controller Industrial Box PC Hardware User's Manual	W556	NY512-□□□□	Learning the basic specifications of the NY-series Industrial Box PCs, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NY-series system is provided along with the following information on the Industrial Box PC. • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection
NJ/NX-series CPU Unit Software User's Manual	W501	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning how to program and set up an NJ/NX-series CPU Unit. Mainly software information is provided.	The following information is provided on a Controller built with an NJ/NX-series CPU Unit. • CPU Unit operation • CPU Unit features • Initial settings • Programming based on IEC 61131-3 language specifications

Manual name	Man. No.	Model	Application	Description
NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Software User's Manual	W558	NY532-□□□□ NY512-□□□□	Learning how to program and set up the Controller functions of an NY-series Industrial PC.	The following information is provided on the NY-series Controller functions. • Controller operation • Controller features • Controller settings • Programming based on IEC 61131-3 language specifications
NJ/NX-series Instructions Reference Manual	W502	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning detailed specifications on the basic instructions of an NJ/NX-series CPU Unit.	The instructions in the instruction set (IEC 61131-3 specifications) are described.
NY-series Instructions Reference Manual	W560	NY532-□□□□ NY512-□□□□	Learning detailed specifications on the basic instructions of an NY-series Industrial PC.	The instructions in the instruction set (IEC 61131-3 specifications) are described.
NJ/NX-series CPU Unit Motion Control User's Manual	W507	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning about motion control settings and programming concepts.	The settings and operation of the CPU Unit and programming concepts for motion control are described.
NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Motion Control User's Manual	W559	NY532-□□□□ NY512-□□□□	Learning about motion control settings and programming concepts of an NY-series Industrial PC.	The settings and operation of the Controller and programming concepts for motion control are described.
NJ/NX-series Motion Control Instructions Reference Manual	W508	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning about the specifications of the motion control instructions.	The motion control instructions are described.
NY-series Motion Control Instructions Reference Manual	W561	NY532-□□□□ NY512-□□□□	Learning about the specifications of the motion control instructions of an NY-series Industrial PC.	The motion control instructions are described.
NJ/NX-series CPU Unit Built-in EtherNet/IP™ Port User's Manual	W506	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Using the built-in EtherNet/IP port on an NJ/NX-series CPU Unit.	Information on the built-in EtherNet/IP port is provided. Information is provided on the basic setup, tag data links, and other features.

Manual name	Man. No.	Model	Application	Description
NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Built-in EtherNet/IP™ Port User's Manual	W563	NY532-□□□□ NY512-□□□□	Using the built-in EtherNet/IP port in an NY-series Industrial PC.	Information on the built-in EtherNet/IP port is provided. Information is provided on the basic setup, tag data links, and other features.
NJ/NY-series NC Integrated Controller User's Manual	O030	NJ501-5300 NY532-5400	Performing numerical control with NJ/NY-series Controllers.	Describes the functionality to perform the numerical control.
NJ/NY-series G code Instructions Reference Manual	O031	NJ501-5300 NY532-5400	Learning about the specifications of the G code/M code instructions.	The G code/M code instructions are described.
Sysmac Studio Version 1 Operation Manual	W504	SYSMAC -SE2□□□	Learning about the operating procedures and functions of the Sysmac Studio.	Describes the operating procedures of the Sysmac Studio.
CNC Operator Operation Manual	O032	SYSMAC-RTNC0□□□D	Learning an introduction of the CNC Operator and how to use it.	An introduction of the CNC Operator, installation procedures, basic operations, connection operations, and operating procedures for main functions are described.

Revision History

A manual revision code appears as a suffix to the catalog number on the front and back covers of the manual.



Revision code	Date	Revised content
01	April 2018	Original production
02	January 2019	Added the target model number.

Sections in this Manual

1

Sysmac Library Usage Procedure

The section describes the procedure to use Sysmac Library installed using the installer, and Sysmac Library in the CPU unit or Industrial PC.

1-1 Procedure to Use Sysmac Library Installed Using the Installer

This section describes the procedure to use Sysmac Library installed using the installer.

There are two ways to use libraries.

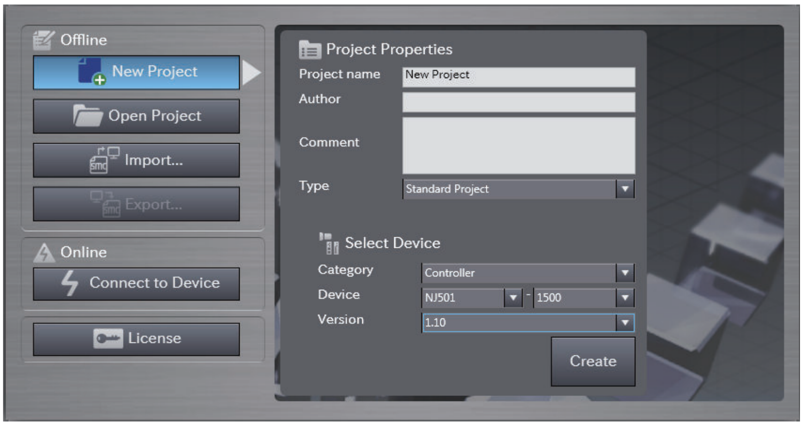
- Using a newly installed Sysmac Library
- Using an upgraded Sysmac Library

Version Information

To use Sysmac Library, you need Sysmac Studio Ver.1.14 or higher.

1-1-1 Using a Newly Installed Sysmac Library

1 Start the Sysmac Studio and open a project using Sysmac Library, or create a new one.



Project Properties

Project name: New Project

Author:

Comment:

Type: Standard Project

Select Device

Category: Controller

Device: NJ501 - 1500

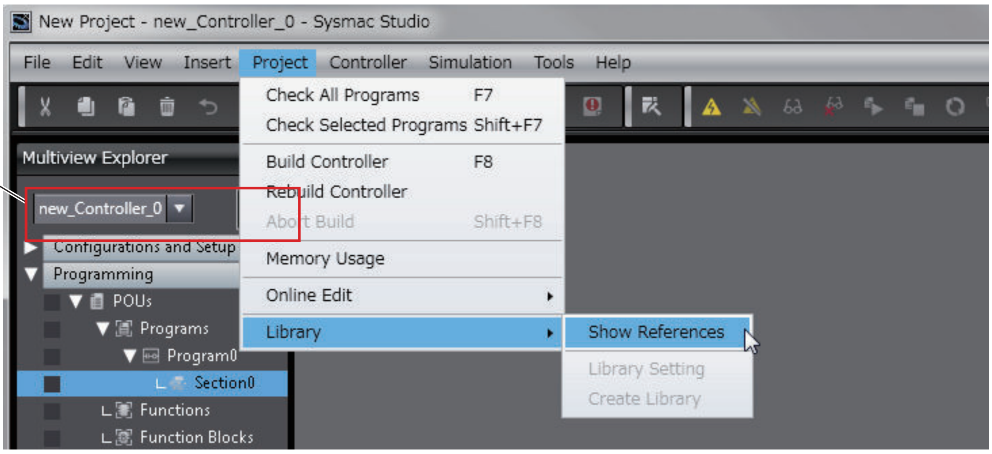
Version: 1.10

Precautions for Correct Use

If you create a new project, be sure to configure the settings as follows to enable use of the Sysmac Library. Without the settings below, you cannot proceed to Step 2 and later steps.

- Set the project type to Standard Project or Library Project.
- Set the device category to Controller.
- For the setting of Controller and Version in the Select Device section, refer to .

2 Select **Project** - **Library** - **Show References**.

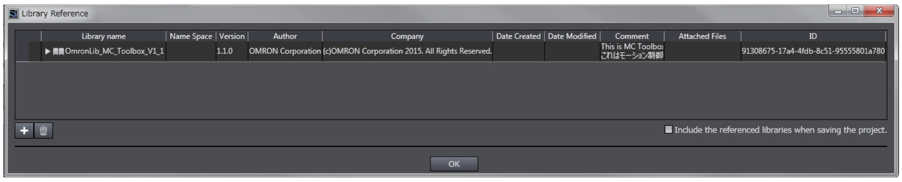


Device

Precautions for Correct Use

If you have multiple devices registered in the project, make sure that the currently selected device is the NJ/NX-series CPU Unit or NY-series Industrial PC. If the NJ/NX-series CPU Unit or NY-series Industrial PC is not selected, the menu for browsing the library will not appear. When the selected device is the NJ/NX-series CPU Unit or NY-series Industrial PC, the device icon displayed in Multiview Explorer changes to ▮▮.

3 Add Sysmac Library to the list and click **OK**.



Library Reference

Library name	Name Space	Version	Author	Company	Date Created	Date Modified	Comment	Attached Files	ID
▶ OmronLib_MC_Toolbox_V1_1		1.1.0	OMRON Corporation	(c)OMRON Corporation 2015. All Rights Reserved.			This is MC Toolbo これはモーション制御		91308675-17a4-4fdb-8c51-95555801a780

☐ Include the referenced libraries when saving the project.

OK

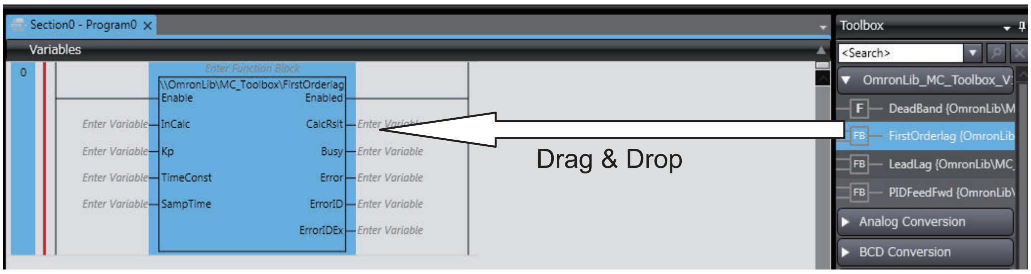
Sysmac Library is read into the project.

Now, when you select the Ladder Editor or ST Editor, the function blocks and functions included in the Sysmac Library appear in the Toolbox.

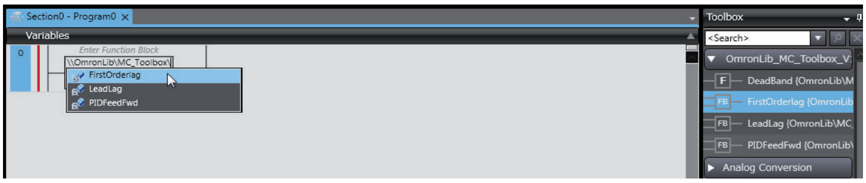
For the procedure for adding and setting libraries in the above screen, refer to *Sysmac Studio Version 1 Operation Manual (W504)*.

4 Insert the Sysmac Library's function blocks and functions into the circuit using one of the following two methods.

- Select the desired function block or function in the Toolbox and drag and drop it onto the Ladder Editor.



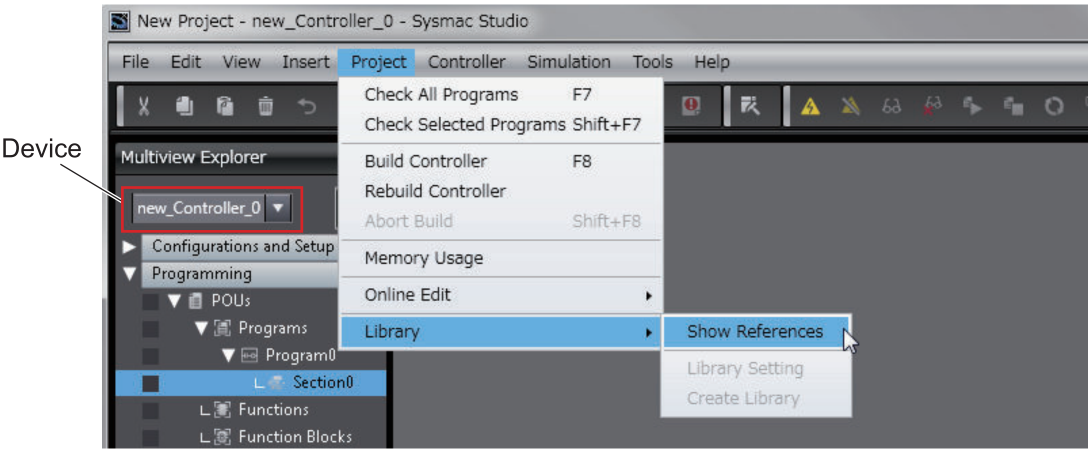
- Right-click the Ladder Editor, select **Insert Function Block** in the menu, and enter the fully qualified name (¥¥namespacename¥FBname).



1-1-2 Using an Upgraded Sysmac Library

1 Start Sysmac Studio and open a project in which any old-version Sysmac Library is included.

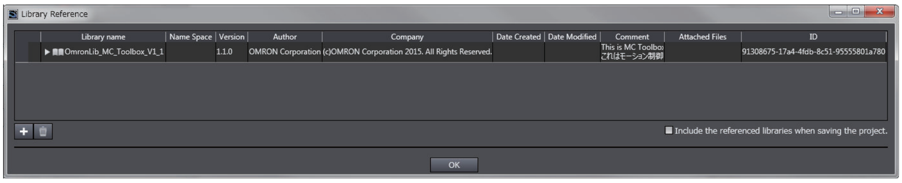
2 Select **Project** - **Library** - **Show References**.



Precautions for Correct Use

If you have multiple devices registered in the project, make sure that the currently selected device is the NJ/NX-series CPU Unit or NY-series Industrial PC. If the NJ/NX-series CPU Unit or NY-series Industrial PC is not selected, the menu for browsing the library will not appear. When the selected device is the NJ/NX-series CPU Unit or NY-series Industrial PC, the device icon displayed in Multiview Explorer changes to .

3 Select an old-version Sysmac Library and click the **Delete Reference** Button.

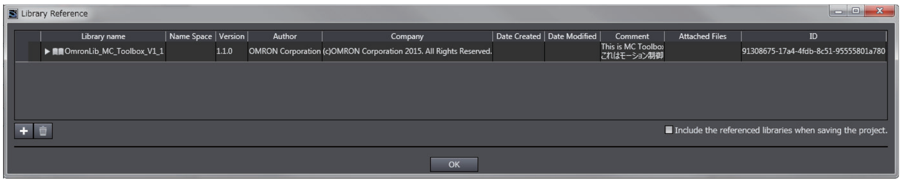

Library Reference

Library name	Name Space	Version	Author	Company	Date Created	Date Modified	Comment	Attached Files	ID
▶ OmronLib_MC_Toolbox_V1_1		1.1.0	OMRON Corporation	(c)OMRON Corporation 2015. All Rights Reserved.			This is MC Toolbox これはモーション制御		91308675-17a4-4fdb-8c51-95555801a780

□ Include the referenced libraries when saving the project.

OK

4 Add Sysmac Library to the list and click **OK**.


Library Reference

Library name	Name Space	Version	Author	Company	Date Created	Date Modified	Comment	Attached Files	ID
▶ OmronLib_MC_Toolbox_V1_1		1.1.0	OMRON Corporation	(c)OMRON Corporation 2015. All Rights Reserved.			This is MC Toolbox これはモーション制御		91308675-17a4-4fdb-8c51-95555801a780

□ Include the referenced libraries when saving the project.

OK

Precautions for Correct Use

Upgrade the Sysmac Library version, and then execute All Program Check, and confirm that there are no errors in the Build Window Program Check results.
From the Main Menu, select **Project** - **All Program Check**.

1-2 How to use Sysmac Library in the CPU Unit or Industrial PC

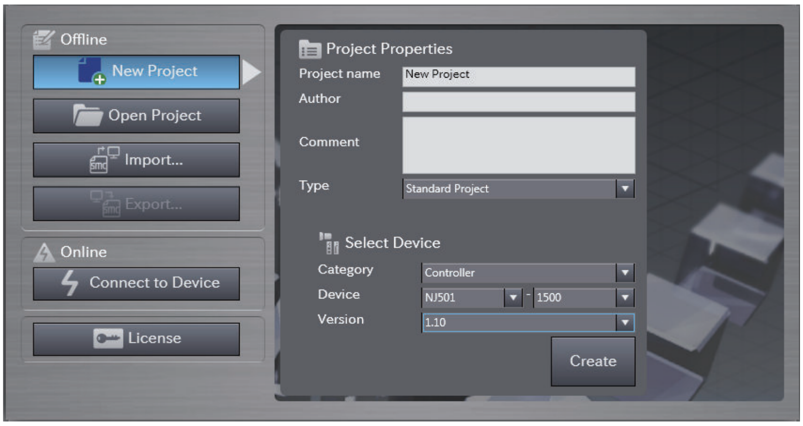
Even when Sysmac Library is not installed on your computer, you can use Sysmac Library by uploading it from the CPU Unit or Industrial PC to your computer.

The procedure to use Sysmac Library in the CPU Unit or Industrial PC is as follows.

Version Information

To use Sysmac Library, you need Sysmac Studio Ver.1.14 or higher.

1 Start the Sysmac Studio and create a new project in which you want to use Sysmac Library.



Project name: New Project

Type: Standard Project

Category: Controller

Device: NJ501 - 1500

Version: 1.10

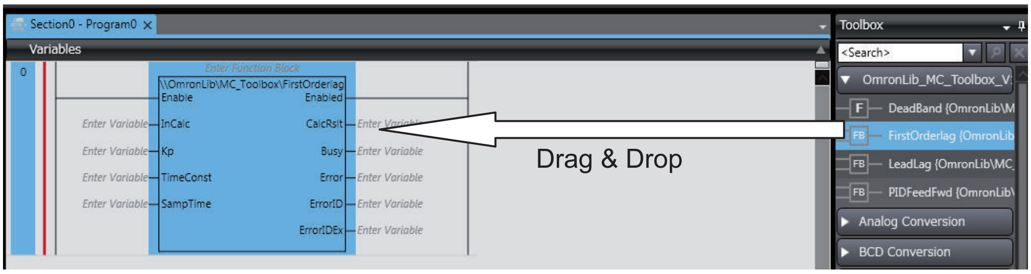
2 Connect online to the CPU Unit or Industrial PC.

3 Upload the POUs in which Sysmac Library is used.

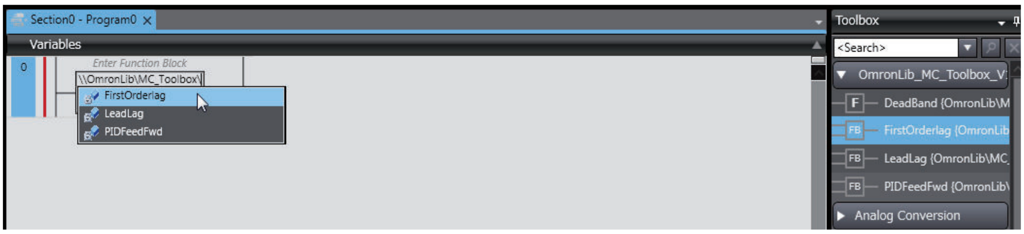
Now, when you select the Ladder Editor or ST Editor, the function blocks and functions included in the Sysmac Library used in the uploaded POUs appear in the Toolbox.

4 Insert the Sysmac Library's function blocks and functions into the circuit using one of the following two methods.

- Select the desired function block or function in the Toolbox and drag and drop it onto the Ladder Editor.



- Right-click the Ladder Editor, select **Insert Function Block** in the menu, and enter the fully qualified name (¥¥namespacename¥FBname).



Precautions for Correct Use

- The Sysmac Studio installs Sysmac Library library files to the specified folder on the computer if they are not present. However, the Sysmac Studio does not install libraries to the specified folder on the computer if they are present.
  The specified folder here means the folder in which library files are installed by the installer.
- Note that uploading Sysmac Library from a CPU Unit or Industrial PC does not install the manual and help files for Sysmac Library, unlike installation using the installer. Please install the manual and help files using the installer if you need them.

2

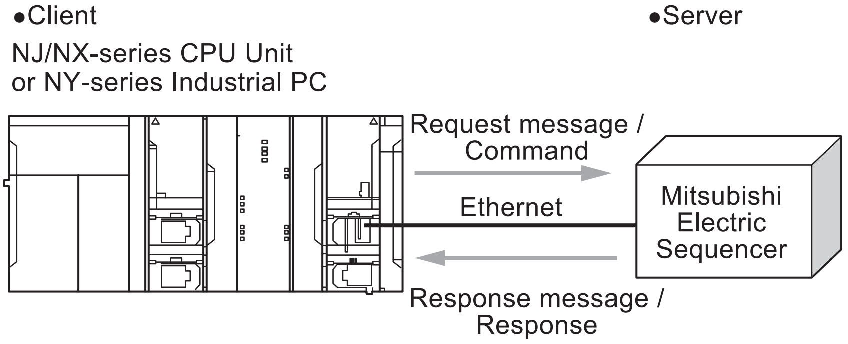
SLMP Communications Library

This section describes the shared specifications of each FB in the SLMP Communications Library.

2-1 Overview

The SLMP Communications Library is a collection of functional objects that uses the SLMP communications protocol for the Sequencer made by Mitsubishi Electric to carry out communications control.
Various FB types (Device Read, Device Write) are provided for easy incorporation of the OMRON Machine Automation Controller in networks configured by the Mitsubishi Electric Sequencer.
This enables the use of a Machine Automation Controller to monitor systems configured by the Mitsubishi Electric Sequencer and perform controls.
Of the four communications models in SLMP, this library supports the Client/Server (single request) type.

- System configuration



The specifications of the 3 FBs provided by this library are described below.

2-1-1 SLMP_DeviceRead (Device Read)

You can use this function to read values for the specified device.
The read position, size, and type (word, bit) can be specified.

2-1-2 SLMP_DeviceWriteWord, SLMP_DeviceWriteBool (Device Write)

You can use this function to write values for the specified device.
The write position and size can be specified.
Use the write type to select the FB used.
Word: SLMP_DeviceWriteWord
Bit: SLMP_DeviceWriteBool

2-1-3 Support for MC Protocol

Since SLMP has the same message format as the MC protocol's QnA interchangeable 3E frame and 4E frame, this library can also be used for Mitsubishi Electric Sequencers supporting the MC protocol.

2-2 Usage Method

This section describes the usage method common to the FB provided in this library.

2-2-1 Preliminary Preparations for SLMP Communications

When performing SLMP communications, you must perform the setting to enable SLMP communications on the Mitsubishi Electric Sequencer side.

2-2-2 Connection Establishment/Disconnection

When you use this library, processes (connection establishment, clearing the reception buffer) for performing Mitsubishi Electric Sequencer and Ethernet communications, and post-processing (connection disconnection) functions are required.
Perform communications preparations in the user program, and execute FB in conditions where communications with the Mitsubishi Electric Sequencer are enabled.
After completion of FB execution, perform the post-processing at a suitable timing.
Refer to the EtherNet/IP communications instructions below in the *NJ/NX-series Instructions Reference Manual (W508)* or *NY-series Instructions Reference Manual (W560)* for details on function blocks for connection control.

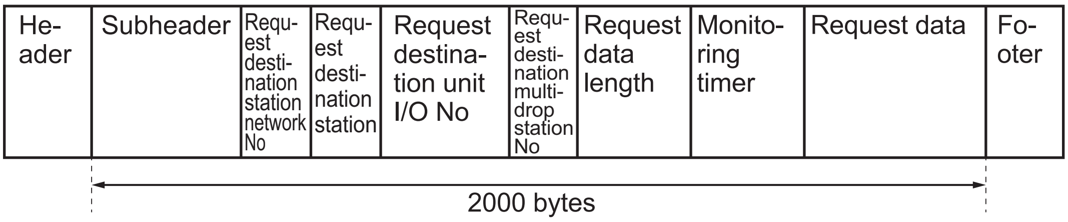
Process	Instructions
Establish connection	SktTCPConnect
Clear the reception buffer	SktClearBuf
Shut off connection	SktClose

2-2-3 Precautions for FB Execution

Use a handle with which connections are established to execute the FBs provided in this library.
To send multiple requests with one handle, send the next request after the previous request has been successfully sent.

2-2-4 Maximum Size of Message

The maximum size of message that can be sent in this library is 2000 bytes.


Header
Subheader
Request destination station network No
Request destination station
Request destination unit I/O No
Request destination multi-drop station No
Request data length
Monitoring timer
Request data
Footer
2000 bytes

2-3 Set Values

2-3-1 Device Code (Device Access for Sequencer CPU Made by Mitsubishi Electric)

The devices that support SLMP_DeviceRead, SLMP_DeviceWriteWord, and SLMP_DeviceWriteBool in this library are shown below.
Use the device code listed in ( ) to use 0001/0000 in the sub-command of the command message.

Support	DeviceCode	Device		Type	SLMP Device Code ASCII	SLMP Device Code Binary	Device No. range
N	0	Special relay (SM)		Bit	SM** (SM)	0x0091 (0x91)	Decimal
N	1	Special register (SD)		Word	SD** (SD)	0x00A9 (0xA9)	Decimal
Y	2	Input (X)		Bit	X*** (X*)	0x009C (0x9C)	Hexadecimal
Y	3	Output (Y)		Bit	Y*** (Y*)	0x009D (0x9D)	Hexadecimal
Y	4	Internal relay (M)		Bit	M*** (M*)	0x0090 (0x90)	Decimal
Y	5	Latch relay (L)		Bit	L*** (L*)	0x0092 (0x92)	Decimal
N	6	Annunciator (F)		Bit	F*** (F*)	0x0093 (0x93)	Decimal
N	7	Edge relay (V)		Bit	V*** (V*)	0x0094 (0x94)	Decimal
Y	8	Link relay (B)		Bit	B*** (B*)	0x00A0 (0xA0)	Hexadecimal
Y	9	Data register (D)		Word	D*** (D*)	0x00A8 (0xA8)	Decimal
Y	10	Link register (W)		Word	W*** (W*)	0x00B4 (0xB4)	Hexadecimal
N	11	Timer (T)	Contact (TS)	Bit	TS** (TS)	0x00C1 (0xC1)	Decimal
N	12		Coil (TC)	Bit	TC** (TC)	0x00C0 (0xC0)	Decimal
N	13		Current value (TN)	Word	TN** (TN)	0x00C2 (0xC2)	Decimal
N	14	Long timer (LT)	Contact (LTS)	Bit	LTS* (—)	0x0051 (—)	Decimal
N	15		Coil (LTC)	Bit	LTC* (—)	0x0050 (—)	Decimal
N	16		Current value (LTN)	Double word	LTN* (—)	0x0052 (—)	Decimal

Support	DeviceCode	Device		Type	SLMP Device Code ASCII	SLMP Device Code Binary	Device No. range
N	17	Retentive timer (ST)	Contact (STS)	Bit	STS* (SS)	0x00C7 (0xC7)	Decimal
N	18		Coil (STC)	Bit	STC* (SC)	0x00C6 (0xC6)	Decimal
N	19		Current value (STN)	Word	STN* (SN)	0x00C8 (0xC8)	Decimal
N	20	Long retentive timer (LST)	Contact (LSTS)	Bit	LSTS (—)	0x0059 (—)	Decimal
N	21		Coil (LSTC)	Bit	LSTC (—)	0x0058 (—)	Decimal
N	22		Current value (LSTN)	Double word	LSTN (—)	0x005A (—)	Decimal
N	23	Counter (C)	Contact (CS)	Bit	CS** (CS)	0x00C4 (0xC4)	Decimal
N	24		Coil (CC)	Bit	CC** (CC)	0x00C3 (0xC3)	Decimal
N	25		Current value (CN)	Word	CN** (CN)	0x00C5 (0xC5)	Decimal
N	26	Long counter (LC)	Contact (LCS)	Bit	LCS* (—)	0x0055 (—)	Decimal
N	27		Coil (LCC)	Bit	LCC* (—)	0x0054 (—)	Decimal
N	28		Current value (LCN)	Double word	LCN* (—)	0x0056 (—)	Decimal
N	29	Link special relay (SB)		Bit	SB** (SB)	0x00A1 (0xA1)	Hexadecimal
N	30	Link special register (SW)		Word	SW** (SW)	0x00B5 (0xB5)	Hexadecimal
N	31	Direct access input (DX)		Bit	DX** (DX)	0x00A2 (0xA2)	Hexadecimal
N	32	Direct access output (DY)		Bit	DY** (DY)	0x00A3 (0xA3)	Hexadecimal
N	33	Index register (Z)		Word	Z*** (Z*)	0x00CC (0xCC)	Decimal
N	34	Long index register (LZ)		Double word	LZ** (—)	0x0062 (—)	Decimal
N	35	File register (R)		Word	R*** (R*)	0x00AF (0xAF)	Decimal
Y	36	File register (ZR)		Word	ZR** (ZR)	0x00B0 (0xB0)	Hexadecimal

Support	DeviceCode	Device	Type	SLMP Device Code ASCII	SLMP Device Code Binary	Device No. range
Y	37	Extended data register (D)	Word	— (D*)	— (0xA8)	Decimal
Y	38	Extended link register (W)	Word	— (W*)	— (0xB4)	Hexadecimal
N	39	Refresh data register (RD)	Word	RD** (—)	0x002C (—)	Decimal
N	40	Link direct device				
N	41	Module access device				
N	42	CPU buffer memory access device				

The table below shows whether read/write is enabled or not for a device type.

FB	Bit device	Word device
SLMP_DeviceRead (word access)	Enabled	Enabled
SLMP_DeviceRead (bit access)	Enabled	Disabled [*1]
SLMP_DeviceWriteWord	Enabled	Enabled
SLMP_DeviceWriteBool	Enabled	Disabled [*1]

[*1] If you execute this combination, the FB will produce an error (illegal device code).

3

Common Specifications of Function Blocks

This section describes the shared specifications of each FB in the Sysmac Library.

3-1 Common Variables

This section describes the specifications of variables (EN, Execute, Enable, Abort, ENO, Done, CalcRslt, Enabled, Busy, CommandAborted, Error, ErrorID, and ErrorIDEx) that are used for more than one function or function block. The specifications are described separately for functions, for execute-type function blocks, and for enable-type function blocks.

3-1-1 Definition of Input Variables and Output Variables

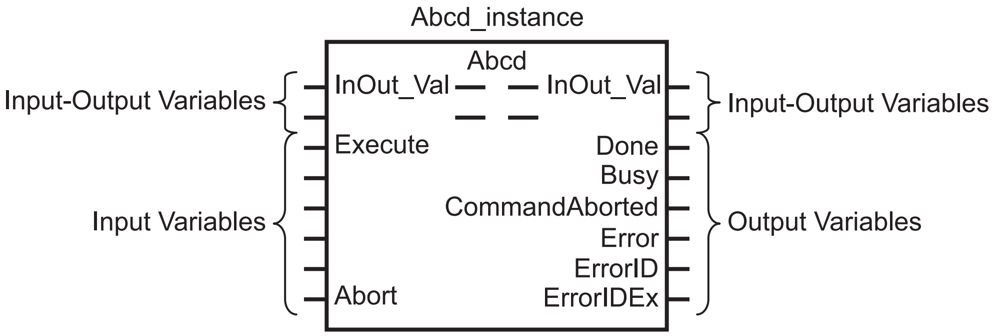
Common input variables and output variables used in functions and function blocks are as follows.

Variable	I/O	Data type	Function/function block type to use			Meaning	Definition
			Function block		Function		
			Execute-type	Enable-type			
EN	Input	BOOL			OK	Execute	The processing is executed while the variable is TRUE.
Execute		BOOL	OK			Execute	The processing is executed when the variable changes to TRUE.
Enable		BOOL		OK		Run	The processing is executed while the variable is TRUE.
Abort		BOOL	OK			Abort	The processing is aborted. You can select the aborting method.
ENO	Output	BOOL			OK	Done	The variable changes to TRUE when the processing ends normally. It is FALSE when the processing ends in an error, the processing is in progress, or the execution condition is not met.
Done		BOOL	OK			Done	The variable changes to TRUE when the processing ends normally. It is FALSE when the processing ends in an error, the processing is in progress, or the execution condition is not met.
Busy		BOOL	OK	OK		Executing	The variable is TRUE when the processing is in progress. Turns to FALSE while the process is not being executed.
CalcRslt		LREAL		OK		Calculation Result	The calculation result is output.

Variable	I/O	Data type	Function/function block type to use			Meaning	Definition
			Function block		Function		
			Execute-type	Enable-type			
Enabled		BOOL		OK		Enabled	The variable is TRUE when the output is enabled. It is used to calculate the control amount for motion control, temperature control, etc.
CommandAborted		BOOL	OK			CommandAborted	The variable changes to TRUE when the processing is aborted. It changes to FALSE when the processing is executed the next time again.
Error		BOOL	OK	OK		Error	This variable is TRUE while there is an error. It is FALSE when the processing ends normally, the processing is in progress, or the execution condition is not met.
ErrorID		WORD	OK	OK		Error Code	An error code is output.
ErrorIDEx		DWORD	OK	OK		Expansion Error Code	An expansion error code is output.

3-1-2 Execute-type Function Blocks

- Processing starts when Execute changes to TRUE.
- When Execute changes to TRUE, Busy also changes to TRUE. When processing is completed normally, Busy changes to FALSE and Done changes to TRUE.
- When continuously executing function blocks of the same instance, change the next Execute to TRUE for at least one task period after Done changes to FALSE in the previous execution.
- If the function block has a CommandAborted (Instruction Aborted) output variable and processing is aborted, CommandAborted changes to TRUE and Busy changes to FALSE.
- If an error occurs in the function block, Error changes to TRUE and Busy changes to FALSE.
- For function blocks that output the result of calculations for motion control and temperature control, you can use the BOOL input variable Abort to abort the FB process. When Abort changes to TRUE, CommandAborted changes to TRUE and the execution of the function block is aborted.

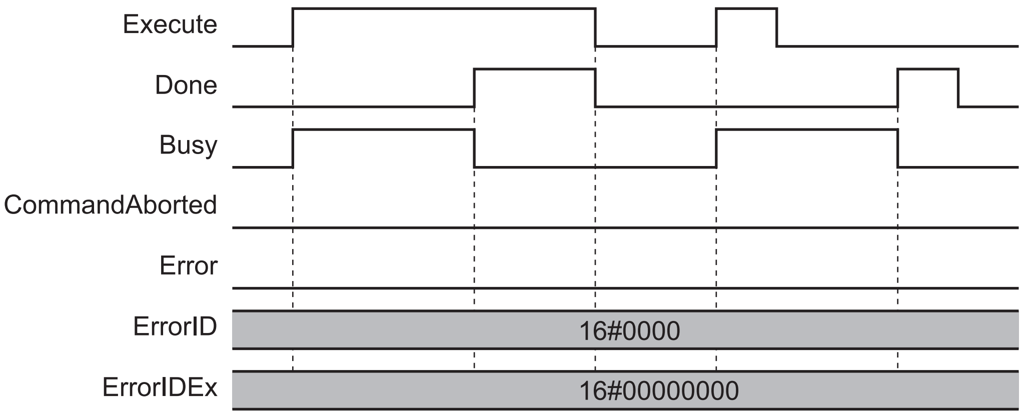


- If Execute is TRUE and Done, CommandAborted, or Error changes to TRUE, Done, CommandAborted, or Error changes to FALSE when Execute is changed to FALSE.
- If Execute is FALSE and Done, CommandAborted, or Error changes to TRUE, Done, CommandAborted, or Error changes to TRUE for only one task period.
- If an error occurs in the function block, the relevant error code and expansion error code are set in ErrorID (Error Code) and ErrorIDEx (Expansion Error Code). The error codes are retained even after Error changes to FALSE, but ErrorID is set to 16#0000 and ErrorIDEx is set to 16#0000 0000 when Execute changes to TRUE.

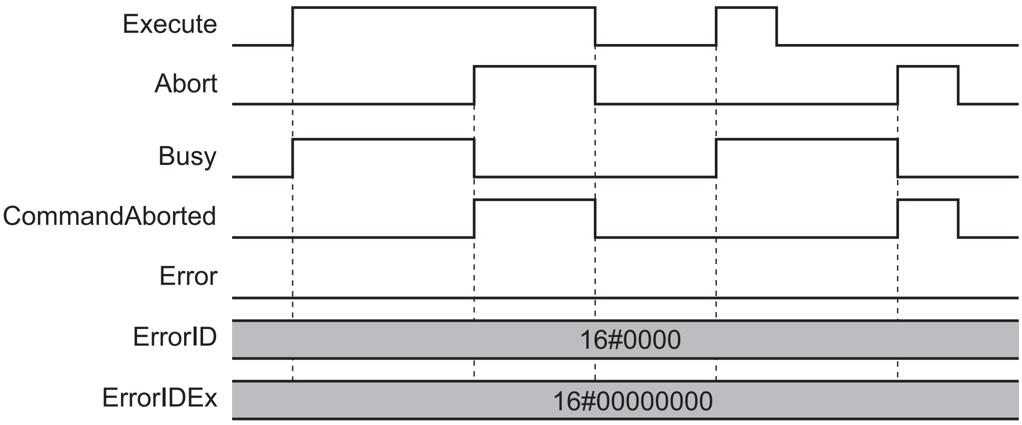
Timing Chart

This section provides timing charts for a normal end, canceled execution, aborted execution, and errors.

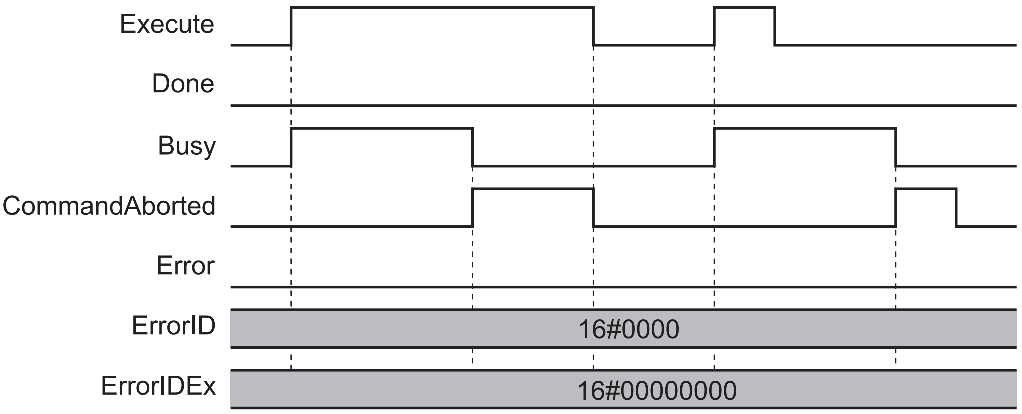
● Normal End



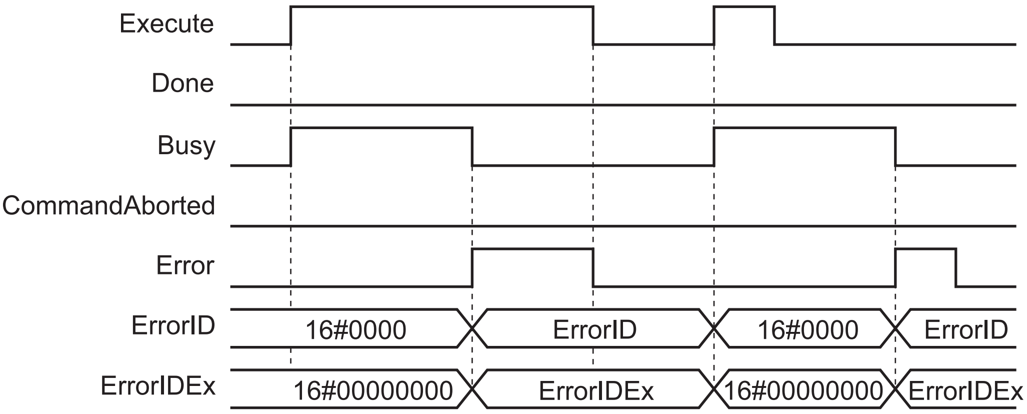
● Canceled Execution



● Aborted Execution

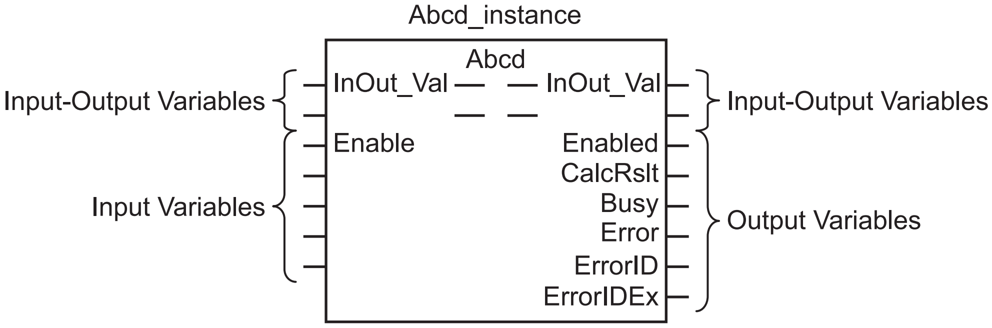


● Errors



3-1-3 Enable-type Function Blocks

- Processing is executed while Enable is TRUE.
- When Enable changes to TRUE, Busy also changes to TRUE. Enabled is TRUE during calculation of the output value.
- If an error occurs in the function block, Error changes to TRUE and Busy and Enabled change to FALSE. When Enable changes to FALSE, Enabled, Busy, and Error change to FALSE.



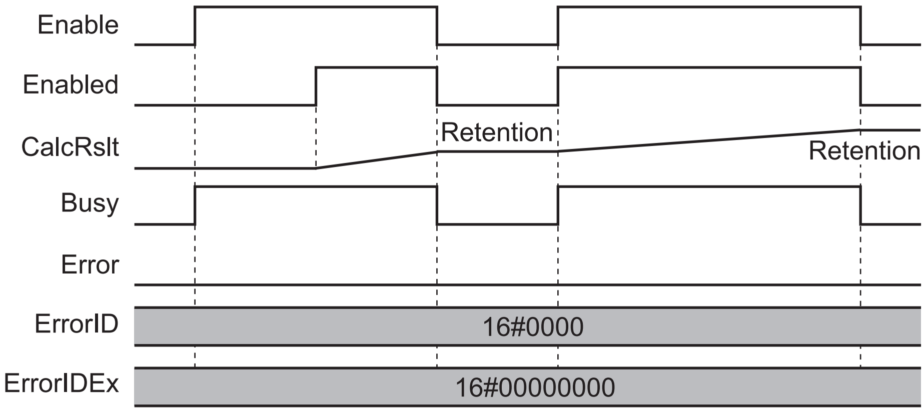
- If an error occurs in the function block, the relevant error code and expansion error code are set in ErrorID (Error Code) and ErrorIDEx (Expansion Error Code). The error codes are retained even after Error changes to FALSE, but ErrorID is set to 16#0000 and ErrorIDEx is set to 16#0000 0000 when Execute changes to TRUE.

- For function blocks that calculate the control amount for motion control, temperature control, etc., Enabled is FALSE when the value of CalcRslt (Calculation Result) is incorrect. In such a case, do not use CalcRslt. In addition, after the function block ends normally or after an error occurs, the value of CalcRslt is retained until Enable changes to TRUE. The control amount will be calculated based on the retained CalcRslt value, if it is the same instance of the function block that changed Enable to TRUE. If it is a different instance of the function block, the control amount will be calculated based on the initial value.

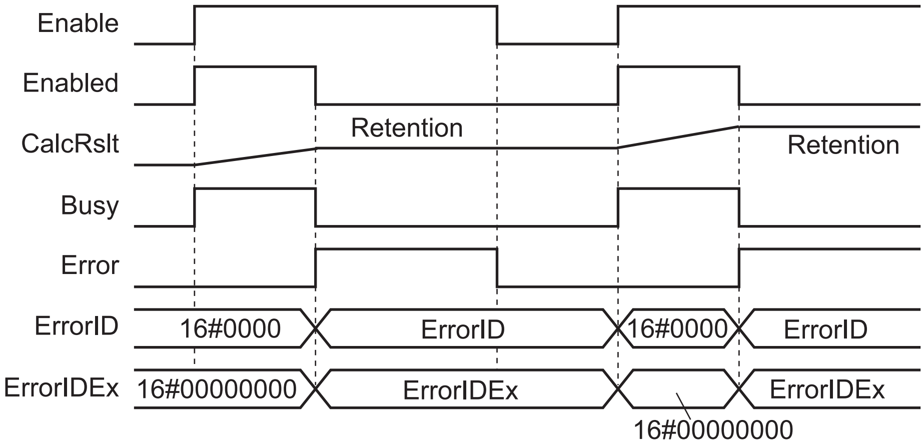
Timing Charts

This section provides timing charts for a normal end and errors.

● Normal End



● Errors



3-2 Precautions

This section provides precautions for the use of this function block.

3-2-1 Nesting

You can nest calls to this function block for up to four levels.
Refer to *NJ/NX-series CPU Unit Software User's Manual (W501)* or *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Software User's Manual (W558)* for details on the nesting function block.

3-2-2 Instruction Options

You cannot use the upward differentiation option for this function block.

3-2-3 Re-execution of Function Blocks

Execute-type function blocks cannot be re-executed by the same instance.
If you do so, the output value will be the initial value.
Refer to *NJ/NX-series CPU Unit Motion Control User's Manual (W507)* or *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Motion Control User's Manual (W559)* for details on re-execution.

4

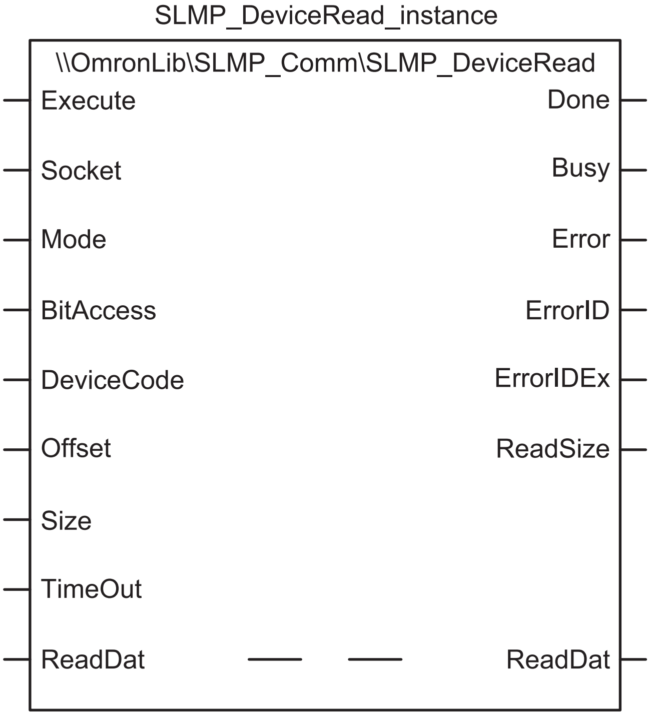
Individual Specifications of Function Blocks

This section describes the individual specifications of each FB in the SLMP Communications Library.

SLMP_DeviceRead

Use SLMP communications to read the value from the sequencer device made by Mitsubishi Electric.

Function block name	Name	FB/ FUN	Graphic expression	ST expression

SLMP_DeviceRead	Internal memory batch read	FB	 SLMP_DeviceRead_instance
			\\OmronLib\SLMP_Comm\SLMP_DeviceRead
			Execute
			Socket
			Mode
			BitAccess
			DeviceCode
			Offset
			Size
			TimeOut
			ReadDat
			Done
			Busy
			Error
			ErrorID
			ErrorIDEx
			ReadSize
			ReadDat
				SLMP_DeviceRead_instance(
				Execute:=,
				Socket:=,
				Mode:=,
				BitAccess:=,
				DeviceCode:=,
				Offset:=,
				Size:=,
				TimeOut:=,
				Done=>,
				Busy=>,
				Error=>,
				ErrorID=>,
				ErrorIDEx=>,
				ReadSize=>,
				ReadDat:=);

Function Block and Function Information

Item	Description
Library file name	OmronLib_SLMP_Comm_Vx_x.slr (x shows the unit version.)
Namespace	OmronLib\SLMP_Comm
Function block and function number	00169
Source code published/not published	Not Published

Input Variables

Variable	Meaning	Data type	Description	Valid range	Unit	Default
Execute	Execute	BOOL	TRUE: Execute FALSE: Do not execute	TRUE, FALSE	—	FALSE
Socket	Socket	_sSOCKET*1	Socket	—	—	—

Variable	Meaning	Data type	Description	Valid range	Unit	Default
Mode	Mode	BOOL	SLMP frame communicating code TRUE: ASCII mode FALSE: BINARY mode	TRUE, FALSE	—	FALSE
BitAccess	Bit access specification	BOOL	SLMP frame communicating code TRUE: Bit unit FALSE: Word unit	TRUE, FALSE	—	FALSE
Device-Code	Device code	UINT	Specifies the device code for the read target. Details are listed in *2-3-1 Device Code (Device Access for Sequencer CPU Made by Mitsubishi Electric)* on page 2 - 4.	Depends on data type.	—	UINT#0
Offset	First device No.	UDINT	Specifies the read device first No.	Depends on data type.	—	UDINT#0
Size	Number of device points	UINT	Specifies the read device number.	Depends on data type.	—	UINT#0
TimeOut	Timeout time	UINT	The timeout time for FB For "0", 2.0s	Depends on data type.	0.1s	UINT#0

[1] Refer to the SktTCPConnect instructions in the *NJ/NX-series Instructions Reference Manual (W502)* or *NY-series Instructions Reference Manual (W560)* for details on the Data Type function block.

Output Variables

Variable	Meaning	Data type	Description	Valid range	Unit	Default
Done	Done	BOOL	TRUE: Normal end FALSE: Error end, execution in progress, or execution condition not met	TRUE, FALSE	—	—
Busy	Executing	BOOL	TRUE: Executing. FALSE: Not executing.	TRUE, FALSE	—	—
Error	Error	BOOL	TRUE: Monitoring in progress. FALSE: Normal end, executing, or execution condition not met	TRUE, FALSE	—	—
ErrorID	Error Code	WORD	This is the error ID for an error end. The value is 16#0 for a normal end.	[1]	—	—
ErrorIDEx	Expansion Error Code	DWORD	This is the error ID for an Expansion Unit Hardware Error. The value is 16#0 for a normal end.	[1]	—	—

Variable	Meaning	Data type	Description	Valid range	Unit	Default
ReadSize	Read data size	UINT	Stores the actual read-out data size.	Depends on data type.	Bytes	—

*1. Refer to *Troubleshooting* on page 4 - 8 for details.

Input-Output Variables

Variable	Meaning	Data type	Description	Valid range	Unit	Default
ReadDat	Read data	ARRAY[*] OF BYTE *1	Stores the read-out data. *2*3	Depends on data type.	—	—

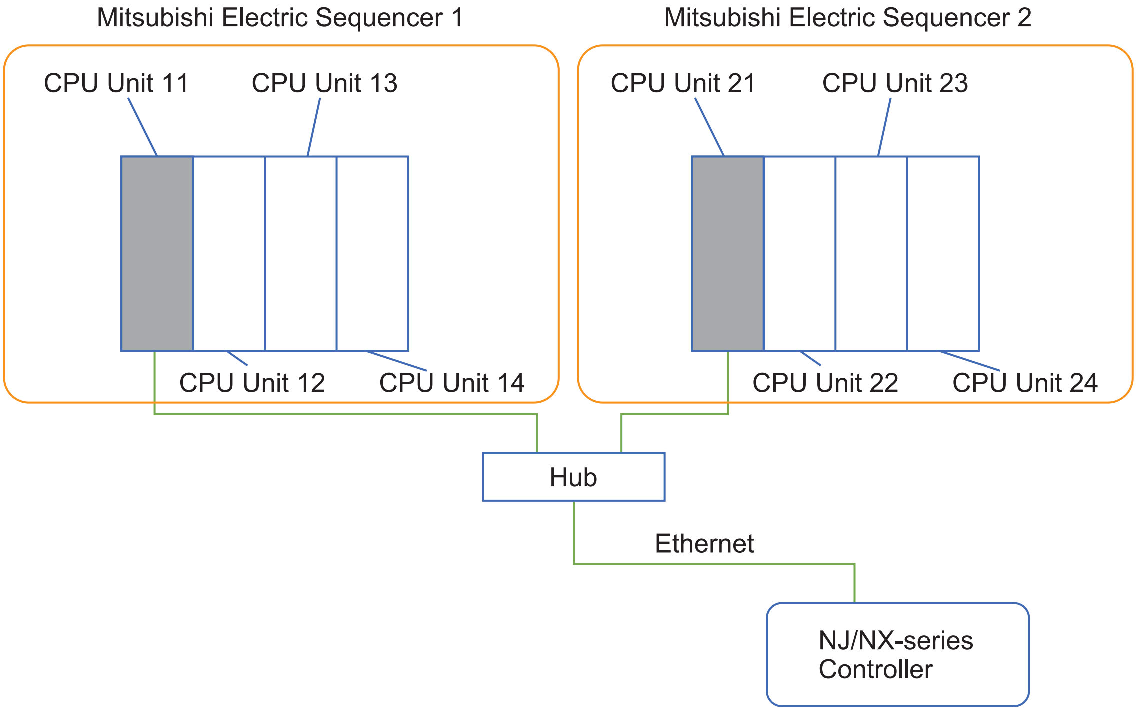
*1. You can input any desired Byte array length. Array length is in the range from 1 to 1920. Basic type Byte cannot be input. Byte arrays can start with either element 0 or with element n.

*2. It is stored in the communicating code specified in Mode. Conversion from BINARY to ASCII and conversion from ASCII to BINARY cannot be performed.

*3. If the received read data exceeds the memory size, the excess data is discarded.

Function

- Use SLMP communications to read the sequencer device made by Mitsubishi Electric on the connected station.
- In the image below, the Mitsubishi Electric Sequencers that can read-out are CPU Unit 11 and CPU Unit 21. CPU Units 12 to 14 and CPU Units 22 to 24 do not support read-out.



- Connect the device specified by the input variable *DeviceCode* from the input variable *Offset*, and read the input variable *Size* amount.
- The SLMP request message is based on the input variables *Mode* and *BitAccess*, and created using Read (Command: 0401, Sub-command: 0001/0000).
- The read result is output to the input/output variable *ReadDat* with the SLMP response data array as is. The output size is output in byte units to the output variable *ReadSize*.

- The Offset range is as follows.

Device No. range	Mode	
	FALSE BINARY mode 3 bytes	TRUE ASCII mode 6 bytes
Hexadecimal	'0' - '16777215 (0xFFFFFF)'	'0' - 'FFFFFF'
Decimal	Same as above (For Decimal → Hexadecimal conversion)	'0' - '999999'

- The possible input range for the input variable *Size*, based on the Input variable *Mode* and input variable *BitAccess*, and the output variable *ReadSize* output range are as shown below.
  When Size is set to 0, the command is completed (Done=TRUE) without a command being sent.
  In this case, ReadSize=0 and ReadDat show no changes.

BitAccess	Input variables	Mode	
		FALSE BINARY mode	TRUE ASCII mode
TRUE When read in bit units	Size	1 - 3840 points	1 - 1920 points
	ReadSize	1 - 1920 bytes	1 - 1920 bytes
FALSE When read in word units	Size	1 - 960 points	1 - 480 points
	ReadSize	2 - 1920 bytes	4 - 1920 bytes

Example of 8 Bit Read-Out from M100 to M107 in Bit Units

- BINARY mode:

Size 8

ReadDat

[0]	[1]	[2]	[3]
0x00	0x01	0x00	0x11
M100/M101	M102/M103	M104/M105	M106/M107

ReadSize 4

- ASCII mode:

Size 8

ReadDat

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
0	0	0	1	0	0	1	1
0x30	0x30	0x30	0x31	0x30	0x30	0x31	0x31

M100 ... M107

ReadSize 8

Example of 2 Word Read-Out from M100 to M131 (Bit Device) in Word Units

M100-M131: 0010/1100/0100/1000 0100/0000/0000/0000

- BINARY mode:

Size 2

ReadDat

[0]	[1]	[2]	[3]
0x34	0x12	0x02	0x00
M107...M100	M115...M108	M123...M116	M131...M124

ReadSize 4

- ASCII mode:

Size 2

ReadDat

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
1	2	3	4	0	0	0	2
0x31	0x32	0x33	0x34	0x30	0x30	0x30	0x32
M115	....		M100	M131	....		M116

ReadSize 8

Example of 3 Word Read-Out from D100 to D102 (Word Device) in Word Units

D100: 0x1234, D101: 0x0002, D102: 0x1DEF

- BINARY mode:

Size 3

ReadDat

[0]	[1]	[2]	[3]	[4]	[5]
0x34	0x12	0x02	0x00	0xEF	0x1D
D100		D101		D102	

ReadSize 6

- ASCII mode:

Size 3

ReadDat

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]
1	2	3	4	0	0	0	2	1	D	E	F
0x31	0x32	0x33	0x34	0x30	0x30	0x30	0x32	0x31	0x44	0x45	0x46
D100				D101				D102			

ReadSize 12

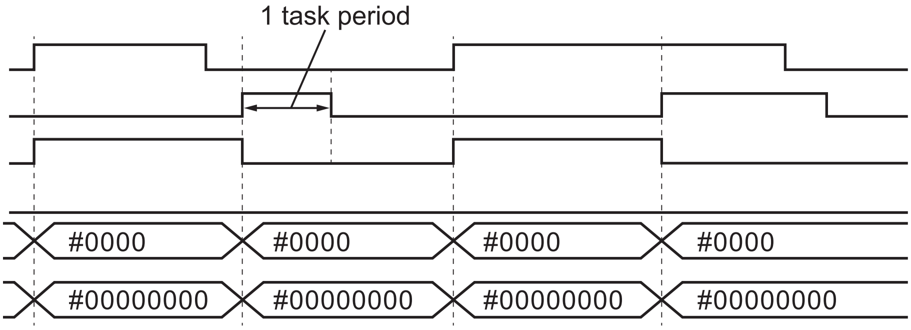
Timing Chart

The timing charts are shown below.

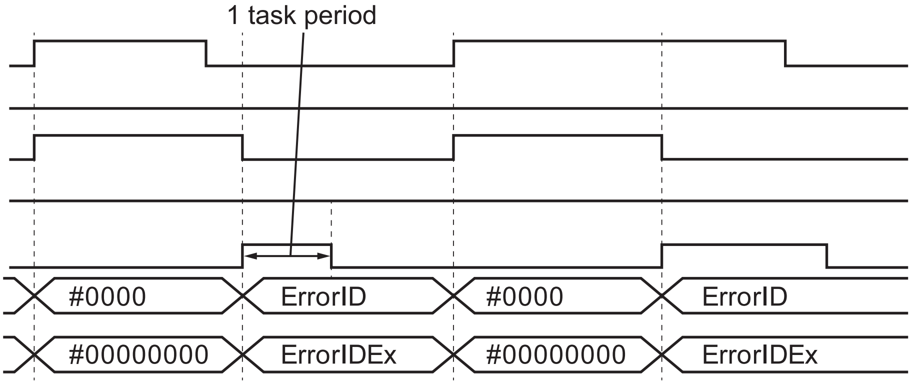
- *Busy* (Executing) changes to TRUE when *Execute* (Execute) changes to TRUE.
- When a response message from the Mitsubishi Electric Sequencer is received successfully, and EndCode=0, *Done* changes to TRUE.

- If an error occurs when execution of the function block is in progress, *Error* changes to TRUE and *Busy* (Executing) changes to FALSE.
  You can find out the cause of the error by accessing the values output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code).
- If *Execute* (Execute) changes to FALSE before execution of the FB is ended, *Done* and *Error* are TRUE only for one task period.
- If *Execute* (Execute) remains TRUE even after execution of the function block is ended, the output values of *Done* and *Error* are retained.

- Timing Chart for Normal End



- Timing Chart for Error End



Additional Information

- For this FB, use the socket service function. Refer to *NJ/NX-series CPU Unit Built-in EtherNet/IP™ Port User's Manual (W506)* or *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Built-in EtherNet/IP™ Port User's Manual (W563)* for details of the socket service function.

Precautions for Correct Use

- Execution of this function block will be continued until processing is ended even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is ended. Use this to confirm normal ending of processing.
- The FB performs one execution for the SktTCPRcv command and the SktTCPSend command, respectively. For the number of simultaneous executions, refer to the *NJ/NX-series Instructions Reference Manual (W502)* or *NY-series Instructions Reference Manual (W560)* SktTCPRcv command and SktTCPSend command.

Precautions for Correct Use

Refer to *2-2 Usage Method* on page 2 - 3.

Troubleshooting

Error code	Expansion error code	Status	Description	Corrective action
16#0000	16#00000000	Normal end	—	—
16#2003	16#00000000	Socket status error	Status at time of socket service instruction execution is not suitable.	Refer to the *NJ/NX-series Instructions Reference Manual (W502)* or *NY-series Instructions Reference Manual (W560)*, and the *NJ/NX-series CPU Unit Built-in EtherNet/IP™ Port User's Manual (W506)* or *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Built-in EtherNet/IP™ Port User's Manual (W563)* and perform troubleshooting.
16#2006	16#00000000	Socket timeout	In socket service instruction, a timeout occurred.	
16#2007	16#00000000	Illegal socket handle	The handle specified in socket service instruction is illegal.	
16#2008	16#00000000	Socket communications resource overflow	Instructions were executed in excess of the socket service instruction resources that can be simultaneously executed.	
16#3CEA	16#00000001	Illegal device code	This function block specified an unsupported device code.	Check the FB input variable *DeviceCode*.
16#3CEA	16#00000002	Illegal first device No.	This FB specified an unsupported first device No.	Check the FB input variable *Offset*.
16#3CEA	16#00000003	Illegal number of device points	Using a combination of the input variables *BitAccess* and *Size*, the read data size exceeded the controller specification upper limit value.	Check the FB input variables *BitAccess* and *Size*.
16#3CEA	16#00000004	ReadDat upper limit exceeded	The array size allocated to *ReadDat* exceeds the upper limit.	Check that the array size allocated to *ReadDat* is 1 - 1920.
16#3CEA	16#00000010	Response reception timeout	A response message could not be received within the time specified in *TimeOut*.	Check the network status and communications target Mitsubishi Electric Sequencer status. Or, increase the timeout value.
16#3CEA	16#FFFF0001 - 16#FFFFFFFF	SLMP end code	An error response message was received.	Reference the SLMP regulations, and perform troubleshooting.

Sample Programming

Precautions for Correct Use

- The sample programming shows only the portion of a program that uses the function or function block from the library.
- When programming actual applications, also program safety circuits, device interlocks, I/O with other devices, and other control procedures.
- Create a user program that will produce the intended device operation.
- Check the user program for proper execution before you use it for actual operation.

Example of program reading 8 bits from M100 to M107 in bit units. It can be used with the MELSEC iQ-R Series, iQ-F Series, Q Series, and L Series.

Conditions

- The timeout time is 2 s (default).

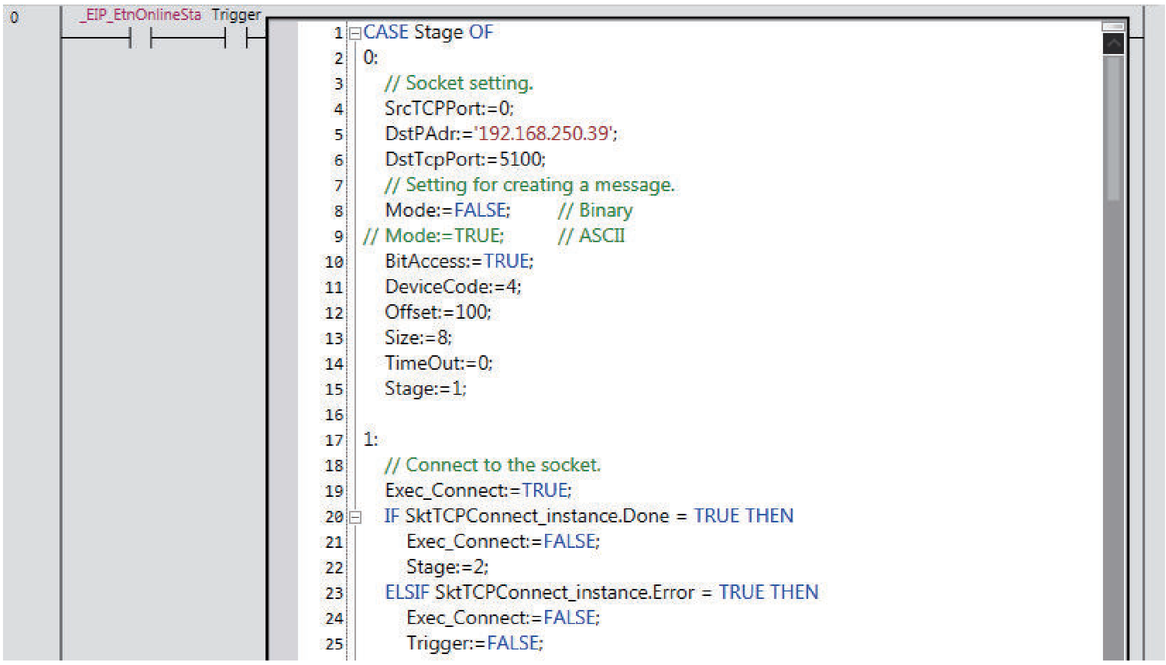
1. Execute the SktTCPConnect command, and connect to the partner TCP port.
2. Execute the SktClearBuf command, and clear the TCP socket receiving buffer.
3. Confirm that no other communications FB execution is in progress.
4. Start up the SLMP_DeviceRead_instance.
5. After completing FB execution, check the read data.
6. Execute the SktClose command, and close the TCP socket.

Main Variables

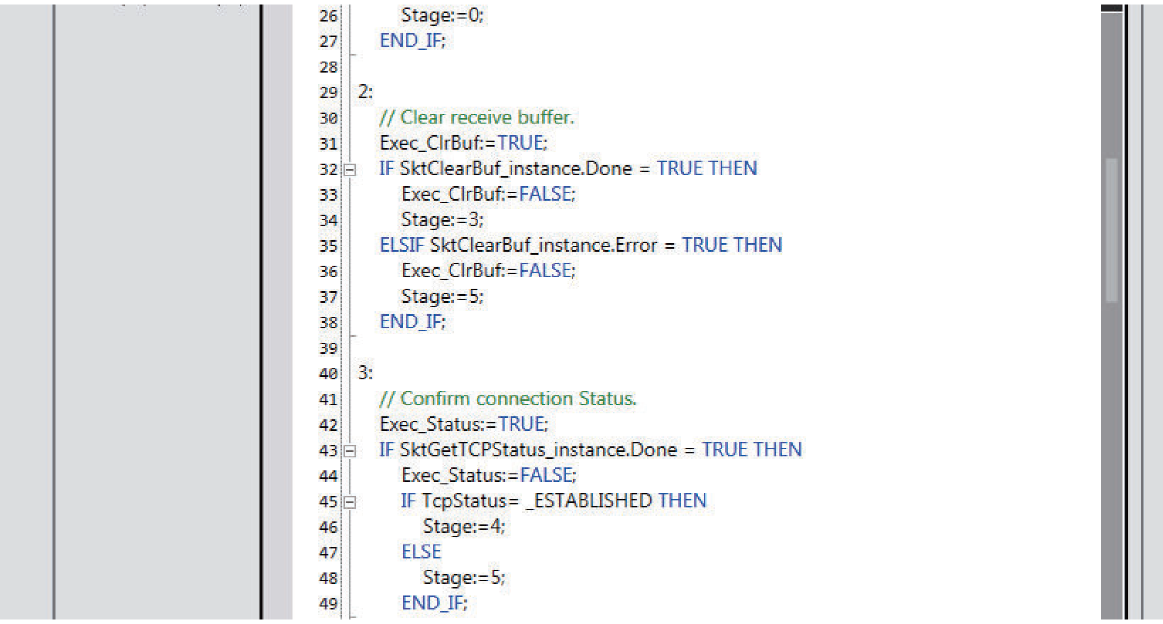
Variable	Data type	Default
Trigger	BOOL	FALSE
Stage	USINT	0
SktTCPConnect_instance	SktTCPConnect	
SktClearBuf_instance	SktClearBuf	
SktGetTCPStatus_instance	SktGetTCPStatus	
SLMP_DeviceRead_instance	OmronLib\SLMP_Comm\SLMP_DeviceRead	
SktClose_instance	SktClose	
Exec_Connect	BOOL	FALSE
Exec_ClrBuf	BOOL	FALSE
Exec_Status	BOOL	FALSE
Exec_Read	BOOL	FALSE
Exec_Close	BOOL	FALSE
ReadDat	ARRAY[1..8] OF BYTE	
TcpStatus	_eCONNECTION_STATE	
Mode	BOOL	
BitAccess	BOOL	
DeviceCode	UINT	

Variable	Data type	Default
Offset	UDINT	
Size	UINT	
TimeOut	UINT	
SrcTCPPort	UINT	
DstPAdr	STRING[64]	
DstTcpPort	UINT	
Socket	_sSOCKET	
ReadSize	UINT	

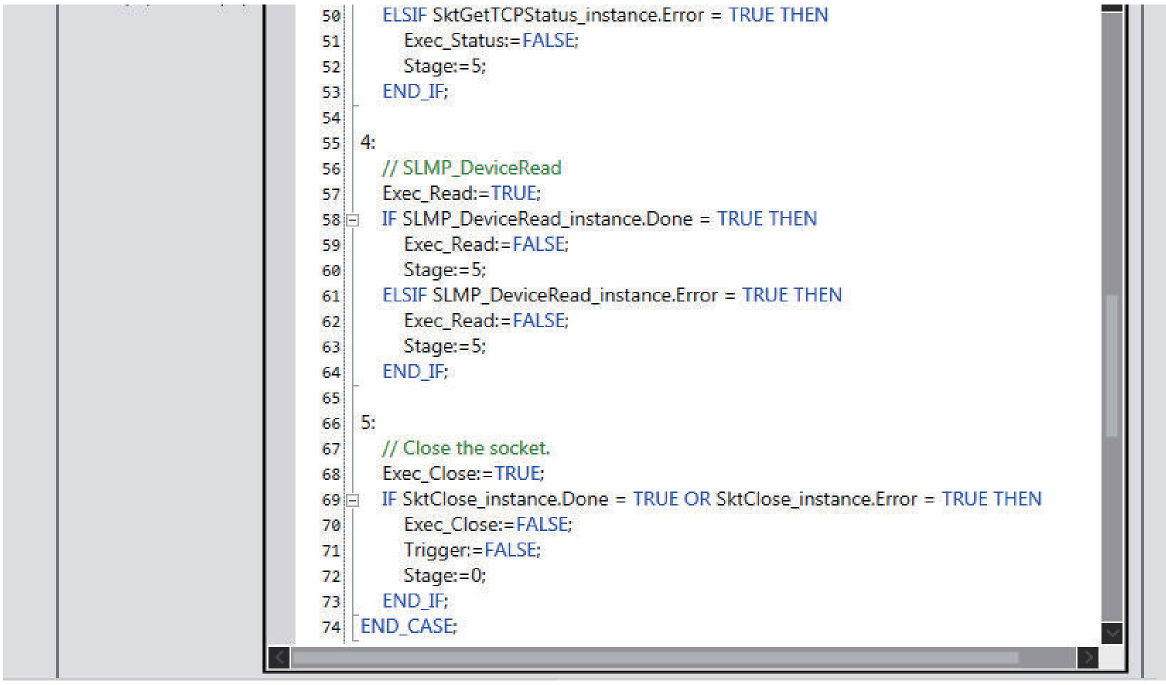
Ladder Diagram



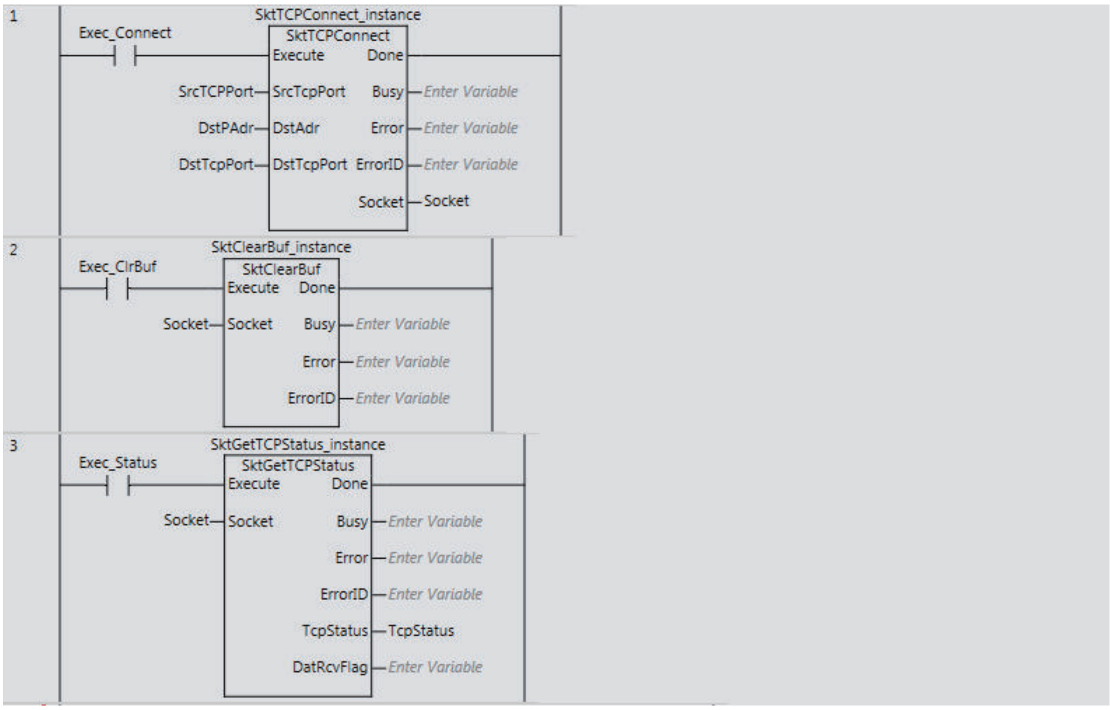
```
CASE Stage OF
0:
    // Socket setting.
    SrcTCPPort:=0;
    DstPAdr:='192.168.250.39';
    DstTcpPort:=5100;
    // Setting for creating a message.
    Mode:=FALSE;        // Binary
// Mode:=TRUE;          // ASCII
    BitAccess:=TRUE;
    DeviceCode:=4;
    Offset:=100;
    Size:=8;
    TimeOut:=0;
    Stage:=1;

1:
    // Connect to the socket.
    Exec_Connect:=TRUE;
    IF SktTCPConnect_instance.Done = TRUE THEN
        Exec_Connect:=FALSE;
        Stage:=2;
    ELSIF SktTCPConnect_instance.Error = TRUE THEN
        Exec_Connect:=FALSE;
        Trigger:=FALSE;
```

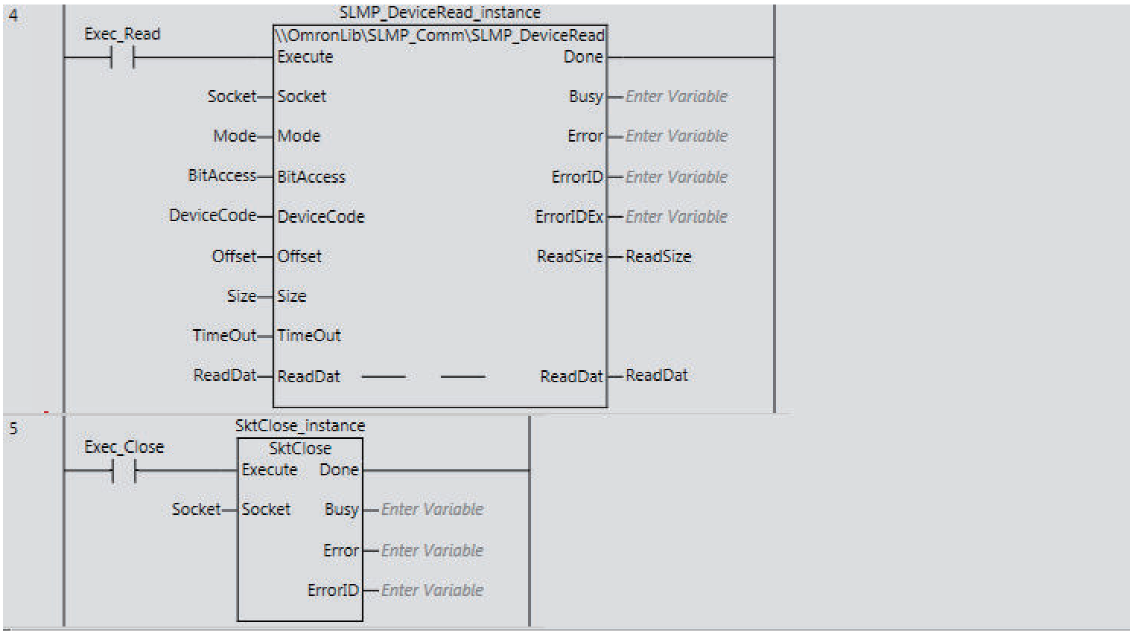


```
        Stage:=0;
    END_IF;

2:
    // Clear receive buffer.
    Exec_ClrBuf:=TRUE;
    IF SktClearBuf_instance.Done = TRUE THEN
        Exec_ClrBuf:=FALSE;
        Stage:=3;
    ELSIF SktClearBuf_instance.Error = TRUE THEN
        Exec_ClrBuf:=FALSE;
        Stage:=5;
    END_IF;

3:
    // Confirm connection Status.
    Exec_Status:=TRUE;
    IF SktGetTCPStatus_instance.Done = TRUE THEN
        Exec_Status:=FALSE;
        IF TcpStatus= _ESTABLISHED THEN
            Stage:=4;
        ELSE
            Stage:=5;
        END_IF;
```



```
  ELSIF SktGetTCPStatus_instance.Error = TRUE THEN
    Exec_Status:=FALSE;
    Stage:=5;
  END_IF;

4:
  // SLMP_DeviceRead
  Exec_Read:=TRUE;
  IF SLMP_DeviceRead_instance.Done = TRUE THEN
    Exec_Read:=FALSE;
    Stage:=5;
  ELSIF SLMP_DeviceRead_instance.Error = TRUE THEN
    Exec_Read:=FALSE;
    Stage:=5;
  END_IF;

5:
  // Close the socket.
  Exec_Close:=TRUE;
  IF SktClose_instance.Done = TRUE OR SktClose_instance.Error = TRUE THEN
    Exec_Close:=FALSE;
    Trigger:=FALSE;
    Stage:=0;
  END_IF;
END_CASE;
```





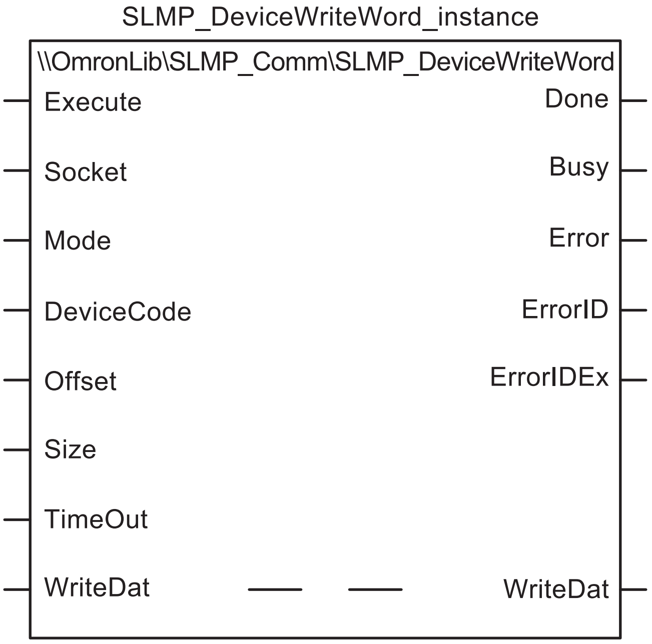
ST

```
IF (Trigger = TRUE) AND (_EIP_EtnOnlineSta=TRUE) THEN
   CASE Stage OF
   0:
      // Socket setting.
      SrcTCPPort:=0;
      DstPAdr:='192.168.250.39';
      DstTcpPort:=5100;
      // Setting for creating a message.
      Mode:=FALSE;         // Binary
//    Mode:=TRUE;          // ASCII
      BitAccess:=TRUE;
      DeviceCode:=4;
      Offset:=100;
      Size:=8;
      TimeOut:=0;
      Stage:=1;

   1:
      // Connect to the socket.
      Exec_Connect:=TRUE;
      IF SktTCPConnect_instance.Done = TRUE THEN
         Exec_Connect:=FALSE;
         Stage:=2;
      ELSIF SktTCPConnect_instance.Error = TRUE THEN
         Exec_Connect:=FALSE;
         Trigger:=FALSE;
         Stage:=0;
   END_IF;

2:
   // Clear receive buffer.
   Exec_ClrBuf:=TRUE;
   IF SktClearBuf_instance.Done = TRUE THEN
      Exec_ClrBuf:=FALSE;
      Stage:=3;
   ELSIF SktClearBuf_instance.Error = TRUE THEN
      Exec_ClrBuf:=FALSE;
      Stage:=5;
   END_IF;

3:
   // Confirm connection Status.
   Exec_Status:=TRUE;
   IF SktGetTCPStatus_instance.Done = TRUE THEN
      Exec_Status:=FALSE;
      IF TcpStatus = _ESTABLISHED THEN
         Stage:=4;
      ELSE
         Stage:=5;
      END_IF;
   ELSIF SktGetTCPStatus_instance.Error = TRUE THEN
      Exec_Status:=FALSE;
      Stage:=5;
   END_IF;

4:
   // SLMP_DeviceRead
   Exec_Read:=TRUE;
   IF SLMP_DeviceRead_instance.Done = TRUE THEN
      Exec_Read:=FALSE;
      Stage:=5;
   ELSIF SLMP_DeviceRead_instance.Error = TRUE THEN
      Exec_Read:=FALSE;
      Stage:=5;
   END_IF;

5:
   // Close the socket.
   Exec_Close:=TRUE;
   IF SktClose_instance.Done = TRUE OR SktClose_instance.Error = TRUE THEN
      Exec_Close:=FALSE;
      Trigger:=FALSE;
      Stage:=0;
   END_IF;
   END_CASE;
ELSE
   Exec_Connect:=FALSE;
   Exec_ClrBuf:=FALSE;
   Exec_Status:=FALSE;
   Exec_Read:=FALSE;
   Exec_Close:=FALSE;
END_IF;

// FB execution.
SktTCPConnect_instance(
   Execute:=Exec_Connect,
   SrcTcpPort:=SrcTCPPort,
   DstAdr:=DstPAdr,
   DstTcpPort:=DstTcpPort,
   Socket=>Socket
   );
SktClearBuf_instance(
   Execute:=Exec_ClrBuf,
   Socket:=Socket
   );
SktGetTCPStatus_instance(
   Execute:=Exec_Status,
   Socket:=Socket,
   TcpStatus=>TcpStatus
   );
SLMP_DeviceRead_instance(
   Execute:=Exec_Read,
   Socket:=Socket,
   Mode:=Mode,
   BitAccess:=BitAccess,
   DeviceCode:=DeviceCode,
   Offset:=Offset,
   Size:=Size,
   TimeOut:=TimeOut,
   ReadSize=>ReadSize,
   ReadDat:=ReadDat
   );
SktClose_instance(
   Execute:=Exec_Close,
   Socket:=Socket
   );
```

SLMP_DeviceWriteWord

Use SLMP communications to write in word units to the sequencer device made by Mitsubishi Electric.

Function block name	Name	FB/ FUN	Graphic expression	ST expression

SLMP_DeviceWriteWord	Internal memory batch write (Word)	FB	 SLMP_DeviceWriteWord_instance
			\\OmronLib\SLMP_Comm\SLMP_DeviceWriteWord
			Execute
			Done
			Socket
			Busy
			Mode
			Error
			DeviceCode
			ErrorID
			Offset
			ErrorIDEx
			Size
			TimeOut
			WriteDat
			WriteDat
				SLMP_DeviceWriteWord_instance(
				Execute:=,
				Socket:=,
				Mode:=,
				DeviceCode:=,
				Offset:=,
				Size:=,
				TimeOut:=,
				Done=>,
				Busy=>,
				Error=>,
				ErrorID=>,
				ErrorIDEx=>,
				WriteDat:=);

Function Block and Function Information

Item	Description
Library file name	OmronLib_SLMP_Comm_Vx_x.slr (x shows the unit version.)
Namespace	OmronLib\SLMP_Comm
Function block and function number	00170
Source code published/not published	Not Published

Input Variables

Variable	Meaning	Data type	Description	Valid range	Unit	Default
Execute	Execute	BOOL	TRUE: Execute FALSE: Do not execute	TRUE, FALSE	—	FALSE
Socket	Socket	_sSOCKET[*1]	Socket	—	—	—
Mode	Mode	BOOL	SLMP frame communicating code TRUE: ASCII mode FALSE: BINARY mode	TRUE, FALSE	—	FALSE

Variable	Meaning	Data type	Description	Valid range	Unit	Default
Device-Code	Device code	UINT	Specifies the device code for the write target. Details are listed in *2-3-1 Device Code (Device Access for Sequencer CPU Made by Mitsubishi Electric)* on page 2 - 4.	Depends on data type.	—	UINT#0
Offset	First device No.	UDINT	Specifies the write device first No.	Depends on data type.	—	UDINT#0
Size	Number of device points	UINT	Specifies the write device number.	Depends on data type.	—	UINT#0
TimeOut	Timeout time	UINT	The timeout time for FB For "0", 2.0s	Depends on data type.	0.1s	UINT#0

*1. Refer to the SktTCPConnect instructions in the *NJ/NX-series Instructions Reference Manual (W502)* or *NY-series Instructions Reference Manual (W560)* for details on the Data Type function block.

Output Variables

Variable	Meaning	Data type	Description	Valid range	Unit	Default
Done	Done	BOOL	TRUE: Normal end FALSE: Error end, execution in progress, or execution condition not met	TRUE, FALSE	—	—
Busy	Executing	BOOL	TRUE: Executing. FALSE: Not executing.	TRUE, FALSE	—	—
Error	Error	BOOL	TRUE: Error end. FALSE: Normal end, executing, or execution condition not met	TRUE, FALSE	—	—
ErrorID	Error Code	WORD	This is the error ID for an error end. The value is 16#0 for a normal end.	*1	—	—
ErrorIDEx	Expansion Error Code	DWORD	This is the error ID for an Expansion Unit Hardware Error. The value is 16#0 for a normal end.	*1	—	—

*1. Refer to *Troubleshooting* on page 4 - 21 for details.

Input-Output Variables

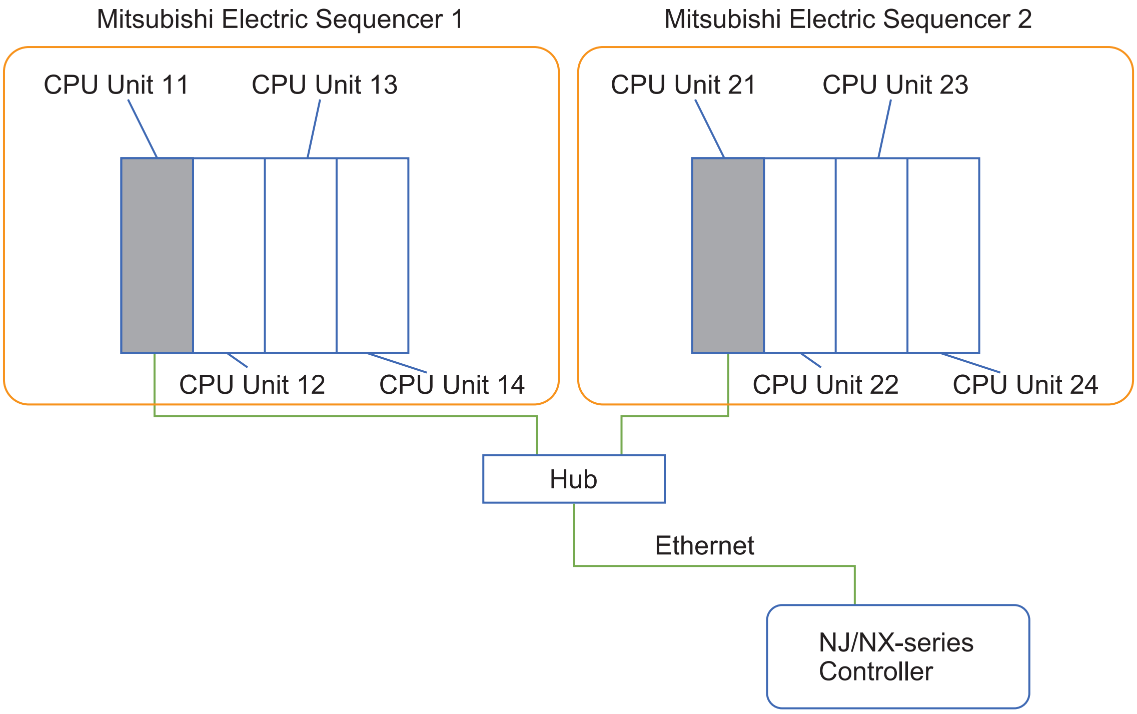
Variable	Meaning	Data type	Description	Valid range	Unit	Default
WriteDat	Write data	ARRAY[*] OF WORD[*1]	Stores the write data. [*2]	Depends on data type.	—	—

[*1.] You can input any desired Word array length. Array length is in the range of 1 - 960 (when in BINARY mode), 1 - 480 (when in ASCII mode). Basic type Word cannot be input. Word arrays can start with either element 0 or with element n.

[*2.] It must be stored in the communicating code specified in Mode. Conversion from BINARY to ASCII and conversion from ASCII to BINARY cannot be performed.

Function

- Use SLMP communications to write in word units to the sequencer device made by Mitsubishi Electric on the connected station.
- In the image below, the Mitsubishi Electric Sequencers that can write-out are CPU Unit 11 and CPU Unit 21. CPU Units 12 to 14 and CPU Units 22 to 24 do not support write-out.



- Connect the device specified by the input variable *DeviceCode* from the input variable *Offset*, and write the input variable *Size* amount.
- The SLMP request message is based on the input variable *Mode*, and created using Write (Command: 1401, Sub-command: 0000).
- Write data is stored in the input/output variable *WriteDat*.
- The Offset range is as follows.

Device No. range	Mode	
	FALSE BINARY mode 3 bytes	TRUE ASCII mode 6 bytes
Hexadecimal	'0' - '16777215 (0xFFFFFF)'	'0' - 'FFFFFF'

Device No. range	Mode	
	FALSE BINARY mode 3 bytes	TRUE ASCII mode 6 bytes
Decimal	Same as above (For Decimal → Hexadecimal conversion)	'0' - '999999'

- The possible input range for the input variable *Size*, based on the input variable *Mode* and input variable *BitAccess*, and the SLMP request message write data length are as shown below.
  When Size is set to 0, the command is completed (Done=TRUE) without a command being sent.

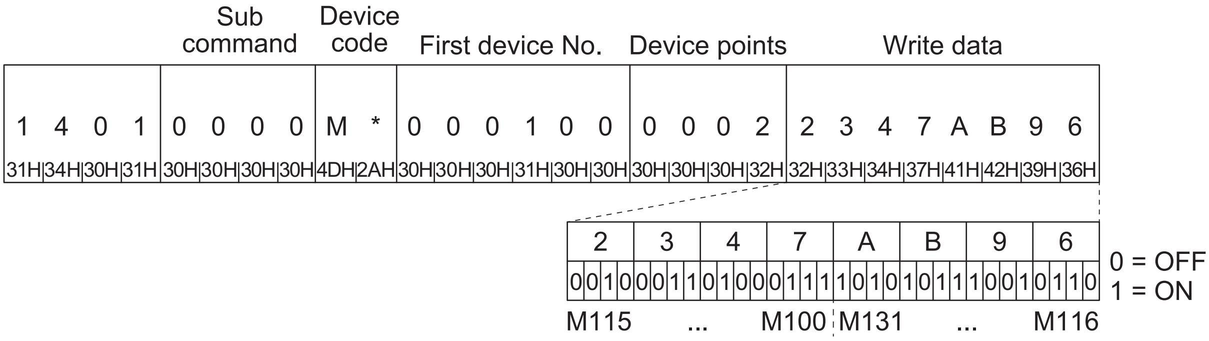
Item name	Mode	
	FALSE BINARY mode	TRUE ASCII mode
Size	1 - 960 points	1 - 480 points
SLMP request message write data length	2 - 1920 bytes	4 - 1920 bytes

Communicating Example (when writing in word units (bit device))

Write the value from M100 to M131 (2 word part).

● When Data Communications are in ASCII Code

(Request data)



	Sub command	Device code	First device No.	Device points	Write data
1 4 0 1	0 0 0 0	M *	0 0 0 1 0 0	0 0 0 2	2 3 4 7 A B 9 6
31H 34H 30H 31H	30H 30H 30H 30H	4DH 2AH	30H 30H 30H 31H 30H 30H	30H 30H 30H 32H	32H 33H 34H 37H 41H 42H 39H 36H

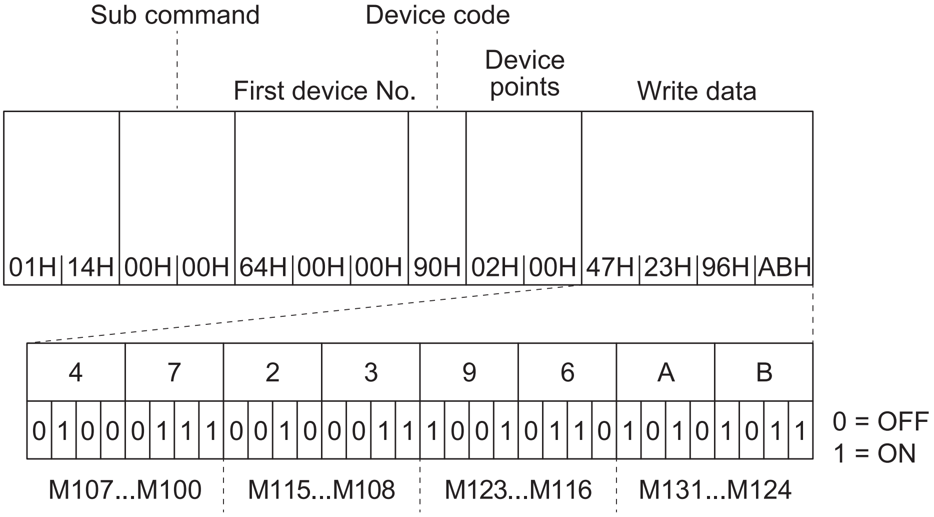
2	3	4	7	A	B	9	6
0 0 1 0	0 0 1 1	0 1 0 0	0 1 1 1	1 0 1 0	1 0 1 1	1 0 0 1	0 1 1 0

M115 ... M100 M131 ... M116

0 = OFF
1 = ON

- **When Data Communications are in Binary Code**

(Request data)

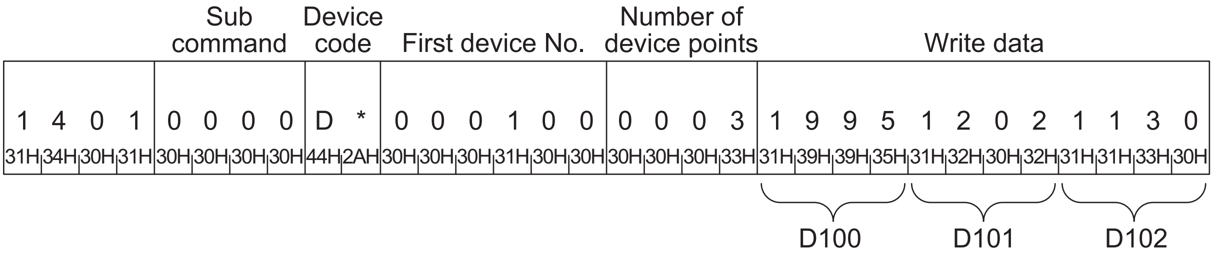


Communicating Example (when writing in word units (word device))

Write the value from D100 to D102 (2 word part).

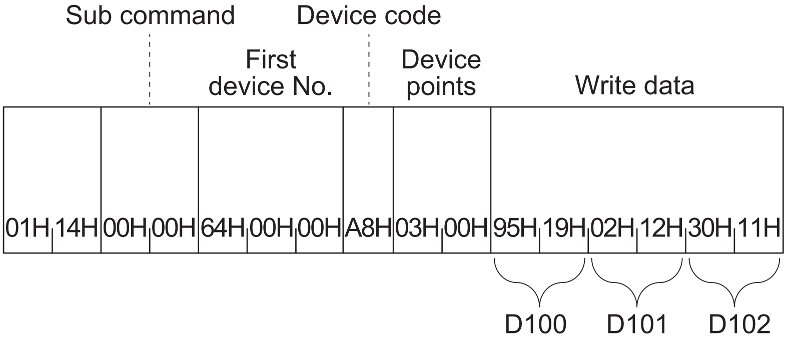
- **When Data Communications are in ASCII Code**

(Request data)


Sub command
Device code
First device No.
Number of device points
Write data
1 4 0 1 0 0 0 0 D * 0 0 0 1 0 0 0 0 0 3 1 9 9 5 1 2 0 2 1 1 3 0
31H 34H 30H 31H 30H 30H 30H 30H 44H 2AH 30H 30H 30H 31H 30H 30H 30H 30H 30H 33H 31H 39H 39H 35H 31H 32H 30H 32H 31H 31H 33H 30H
D100
D101
D102

- **When Data Communications are in Binary Code**

(Request data)



SLMP_DeviceWriteWord

4

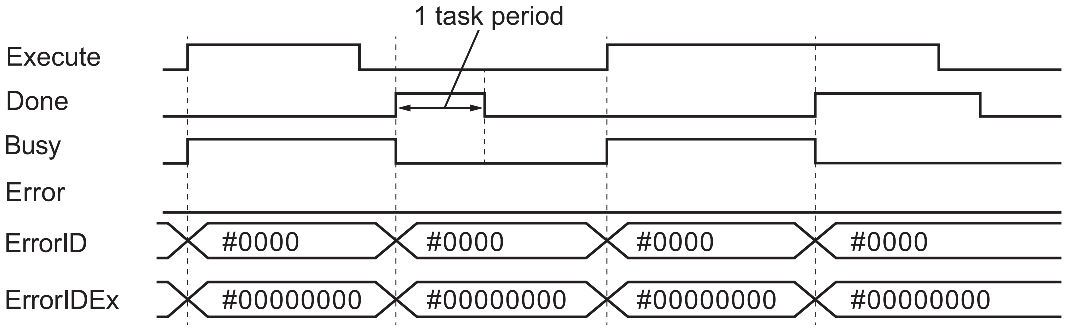
Function

Timing Chart

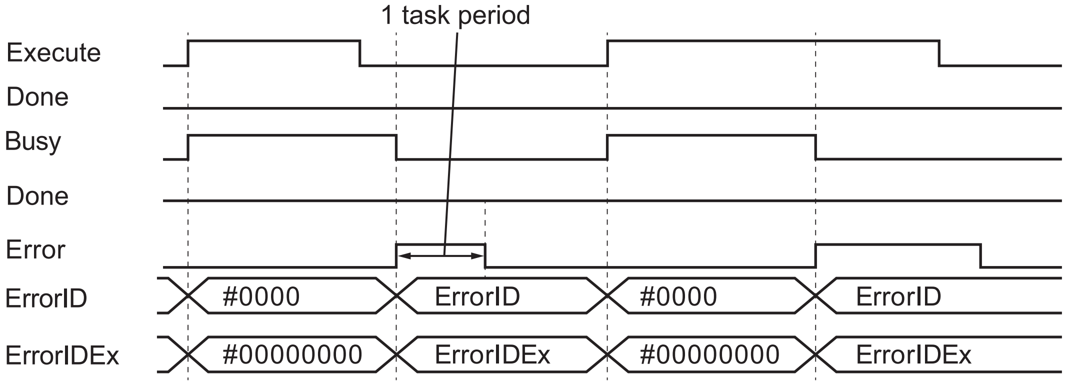
The timing charts are shown below.

- *Busy* (Executing) changes to TRUE when *Execute* (Execute) changes to TRUE.
- When a response message from the Mitsubishi Electric Sequencer is received successfully, and EndCode=0, *Done* changes to TRUE.
- If an error occurs when execution of the function block is in progress, *Error* changes to TRUE and *Busy* (Executing) changes to FALSE.
  You can find out the cause of the error by accessing the values output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code).
- If *Execute* (Execute) changes to FALSE before execution of the FB is ended, *Done* and *Error* are TRUE only for one task period.
- If *Execute* (Execute) remains TRUE even after execution of the function block is ended, the output values of *Done* and *Error* are retained.

- Timing Chart for Normal End



- Timing Chart for Error End



Additional Information

- For this FB, use the socket service function. Refer to *NJ/NX-series CPU Unit Built-in EtherNet/IP™ Port User's Manual (W506)* or *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Built-in EtherNet/IP™ Port User's Manual (W563)* for details of the socket service function.

Precautions for Correct Use

- Execution of this function block will be continued until processing is ended even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is ended. Use this to confirm normal ending of processing.
- The FB performs one execution for the SktTCPRcv command and the SktTCPSend command, respectively. For the number of simultaneous executions, refer to the *NJ/NX-series Instructions Reference Manual (W502)* or *NY-series Instructions Reference Manual (W560)* SktTCPRcv command and SktTCPSend command.

Precautions for Correct Use

- Refer to *2-2 Usage Method* on page 2 - 3.
- If "Write during RUN" is prohibited on the Mitsubishi Electric Sequencer side, confirm that the Mitsubishi Electric Sequencer is in STOP state, and then execute.

Troubleshooting

Error code	Expansion error code	Status	Description	Corrective action
16#0000	16#00000000	Normal end	—	—
16#2003	16#00000000	Socket status error	Status at time of socket service instruction execution is not suitable.	Refer to the *NJ/NX-series Instructions Reference Manual (W502)* or *NY-series Instructions Reference Manual (W560)*, and the *NJ/NX-series CPU Unit Built-in EtherNet/IP™ Port User's Manual (W506)* or *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Built-in EtherNet/IP™ Port User's Manual (W563)* and perform troubleshooting.
16#2006	16#00000000	Socket timeout	In socket service instruction, a timeout occurred.	
16#2007	16#00000000	Illegal socket handle	The handle specified in socket service instruction is illegal.	
16#2008	16#00000000	Socket communications resource overflow	Instructions were executed in excess of the socket service instruction resources that can be simultaneously executed.	
16#3CEB	16#00000001	Illegal device code	This function block specified an unsupported device code.	Check the FB input variable *DeviceCode*.
16#3CEB	16#00000002	Illegal first device No.	This FB specified an unsupported first device No.	Check the FB input variable *Offset*.
16#3CEB	16#00000003	Illegal number of device points	Using the input variable *Size*, the write data size exceeded the controller specification upper limit value.	Check the FB input variable *Size*.
16#3CEB	16#00000004	Insufficient WriteDat size	Array size allocated to WriteDat is less than the write data length specified in *Mode*, *Size*.	Check the FB input variables *Mode* and *Size*. Or, check the buffer size allocated to WriteDat.

Error code	Expansion error code	Status	Description	Corrective action
16#3CEB	16#00000005	WriteDat upper limit exceeded	The array size allocated to *WriteDat* exceeds the upper limit.	Check that the array size allocated to *WriteDat* is 1 - 960 (when in BINARY mode), 1 - 480 (when in ASCII mode).
16#3CEB	16#00000010	Response reception timeout	A response message could not be received within the time specified in *TimeOut*.	Check the network status and communications target Mitsubishi Electric Sequencer status. Or, increase the timeout value.
16#3CEB	16#FFFF0001 - 16#FFFFFFFF	SLMP end code	An error response message was received.	Reference the SLMP regulations, and perform troubleshooting.

Sample Programming

Precautions for Correct Use

- The sample programming shows only the portion of a program that uses the function or function block from the library.
- When programming actual applications, also program safety circuits, device interlocks, I/O with other devices, and other control procedures.
- Create a user program that will produce the intended device operation.
- Check the user program for proper execution before you use it for actual operation.

This is an example of a program writing a value from M100 to M131 (2 word part). It can be used with the MELSEC iQ-R Series, iQ-F Series, Q Series, and L Series.

Conditions

- The timeout time is 2 s (default).

Device Write Process

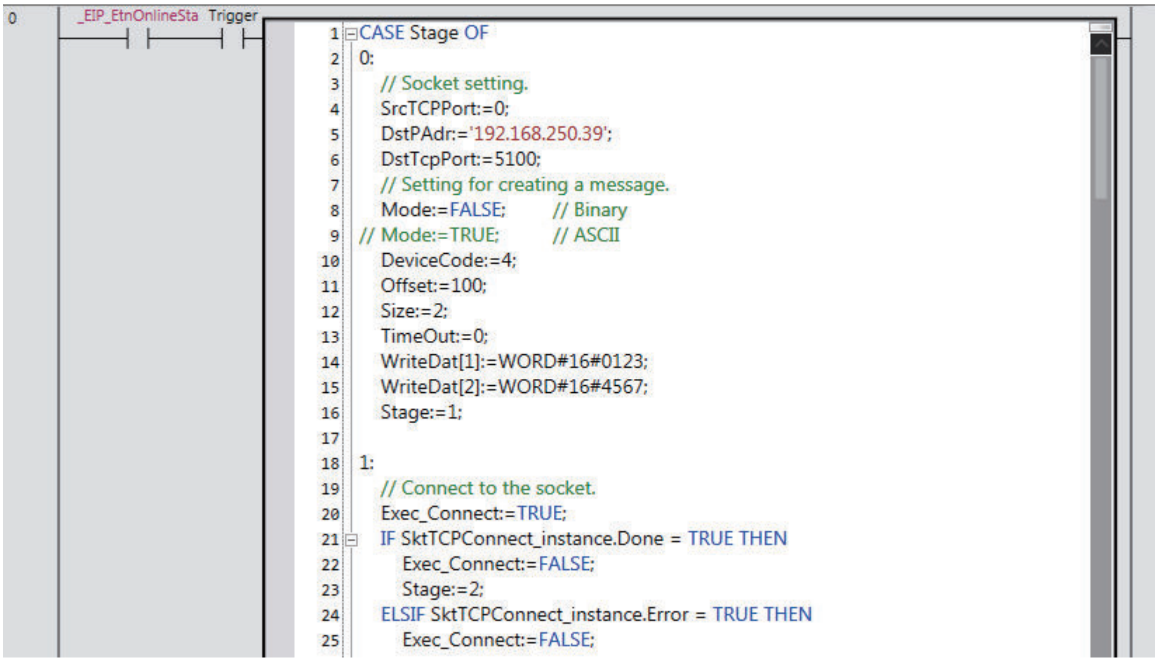
1. Execute the SktTCPConnect command, and connect to the partner TCP port.
2. Execute the SktClearBuf command, and clear the TCP socket receiving buffer.
3. Confirm that no other communications FB execution is in progress.
4. Start up the SLMP_DeviceWriteWord_instance.
5. Execute the SktClose command, and close the TCP socket.

Main Variables

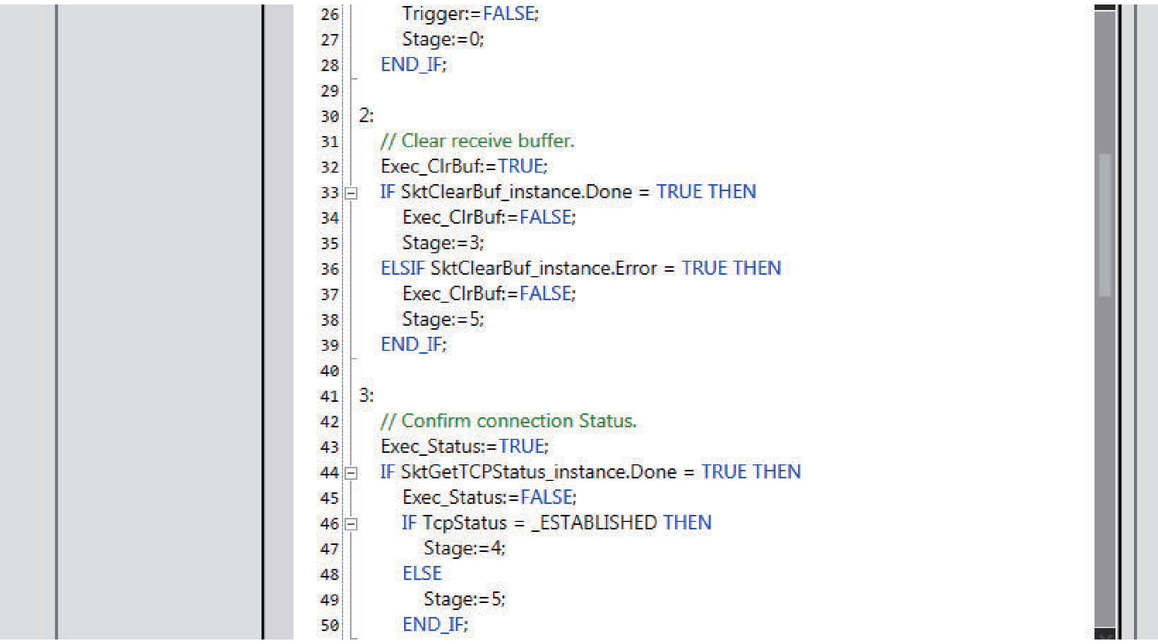
Variable	Data type	Default
Trigger	BOOL	FALSE

Variable	Data type	Default
Stage	USINT	0
SktTCPConnect_instance	SktTCPConnect	
SktClearBuf_instance	SktClearBuf	
SktGetTCPStatus_instance	SktGetTCPStatus	
SLMP_DeviceWriteWord_instance	OmronLib\SLMP_Comm\SLMP_DeviceRead	
SktClose_instance	SktClose	
Exec_Connect	BOOL	FALSE
Exec_ClrBuf	BOOL	FALSE
Exec_Status	BOOL	FALSE
Exec_Read	BOOL	FALSE
Exec_Close	BOOL	FALSE
WriteDat	ARRAY[1..2] OF WORD	
TcpStatus	_eCONNECTION_STATE	
Mode	BOOL	
DeviceCode	UINT	
Offset	UDINT	
Size	UINT	
TimeOut	UINT	
SrcTCPPort	UINT	
DstPAdr	STRING[64]	
DstTcpPort	UINT	
Socket	_sSOCKET	

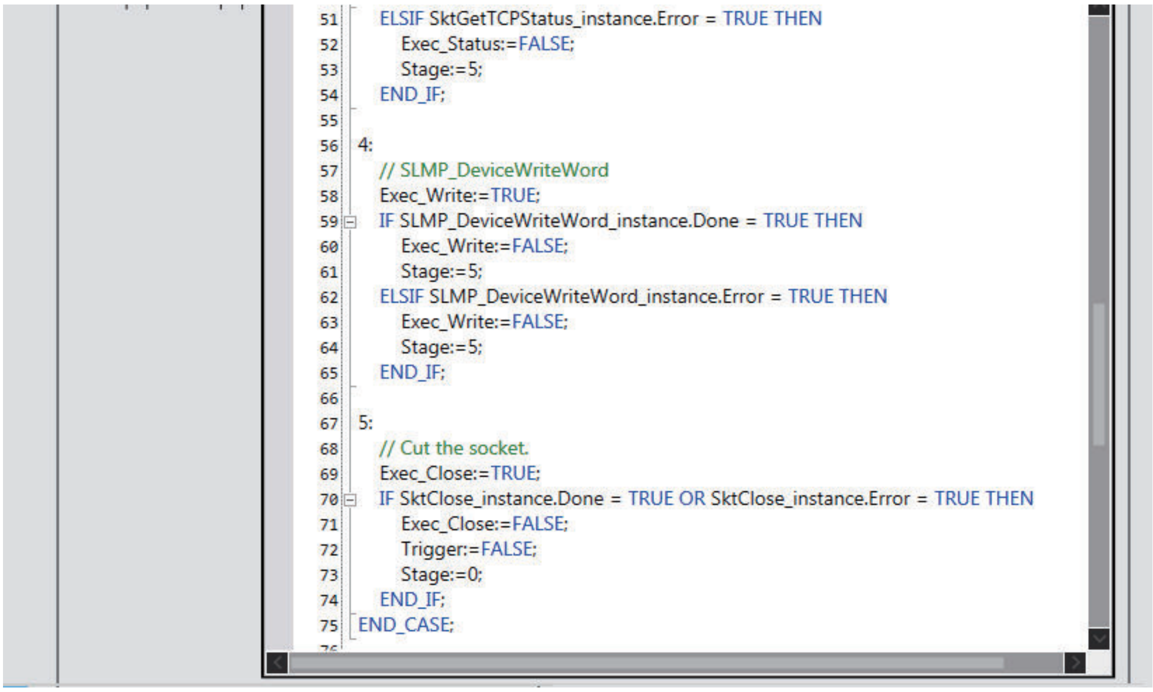
Ladder Diagram



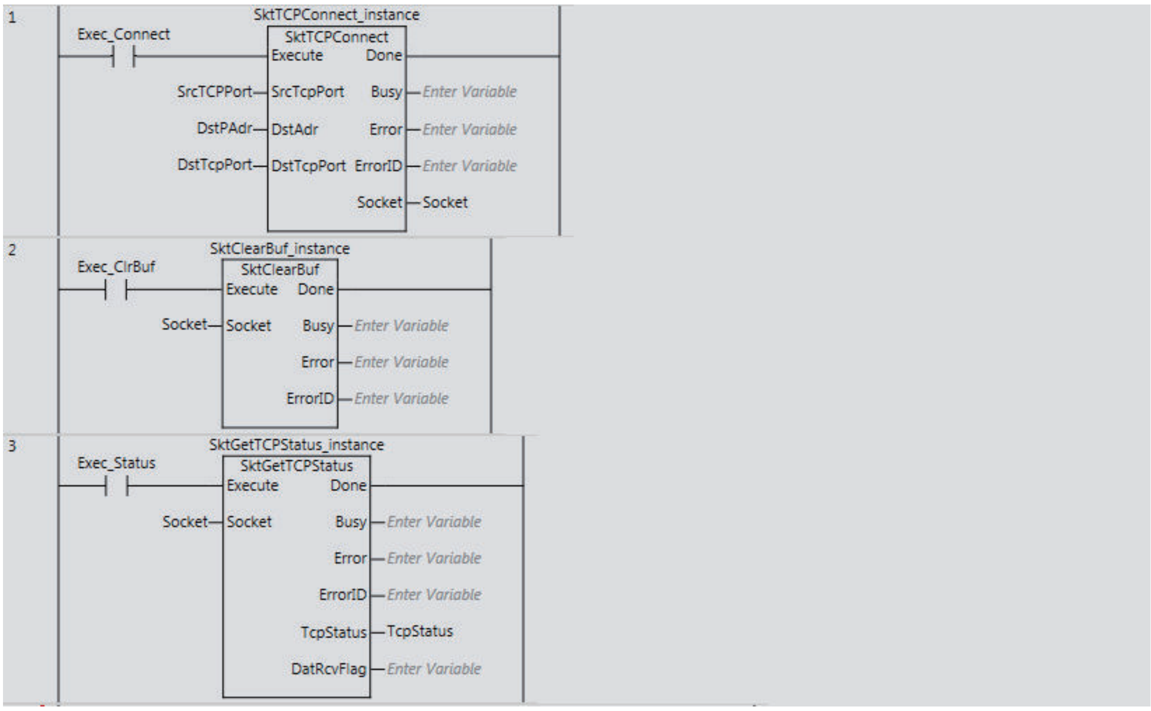
```
CASE Stage OF
0:
   // Socket setting.
   SrcTCPPort:=0;
   DstPAdr:='192.168.250.39';
   DstTcpPort:=5100;
   // Setting for creating a message.
   Mode:=FALSE;        // Binary
// Mode:=TRUE;         // ASCII
   DeviceCode:=4;
   Offset:=100;
   Size:=2;
   TimeOut:=0;
   WriteDat[1]:=WORD#16#0123;
   WriteDat[2]:=WORD#16#4567;
   Stage:=1;

1:
   // Connect to the socket.
   Exec_Connect:=TRUE;
   IF SktTCPConnect_instance.Done = TRUE THEN
      Exec_Connect:=FALSE;
      Stage:=2;
   ELSIF SktTCPConnect_instance.Error = TRUE THEN
      Exec_Connect:=FALSE;
```

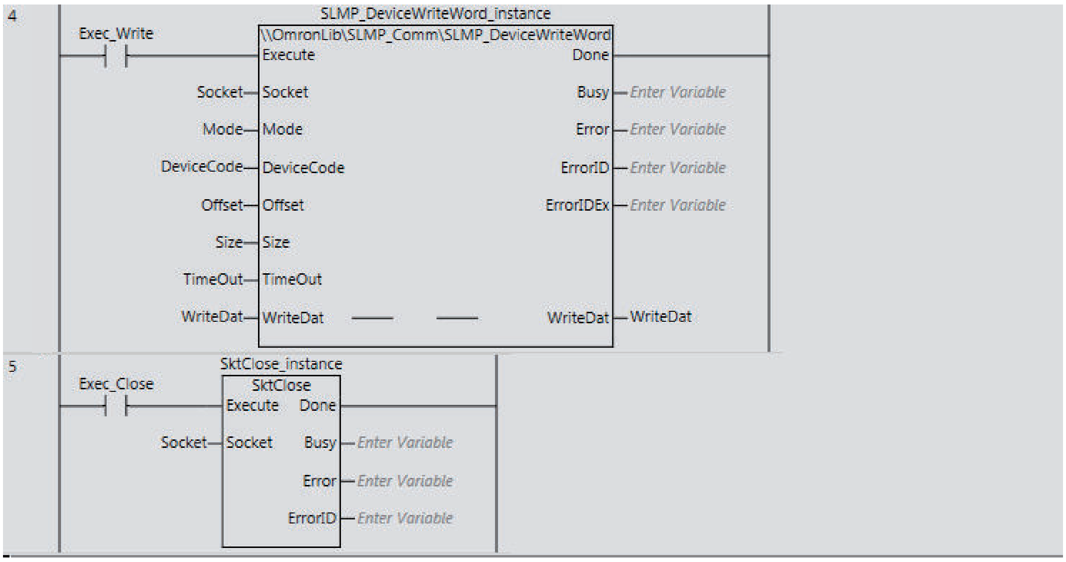


```
        Trigger:=FALSE;
        Stage:=0;
    END_IF;

2:
    // Clear receive buffer.
    Exec_ClrBuf:=TRUE;
    IF SktClearBuf_instance.Done = TRUE THEN
        Exec_ClrBuf:=FALSE;
        Stage:=3;
    ELSIF SktClearBuf_instance.Error = TRUE THEN
        Exec_ClrBuf:=FALSE;
        Stage:=5;
    END_IF;

3:
    // Confirm connection Status.
    Exec_Status:=TRUE;
    IF SktGetTCPStatus_instance.Done = TRUE THEN
        Exec_Status:=FALSE;
        IF TcpStatus = _ESTABLISHED THEN
            Stage:=4;
        ELSE
            Stage:=5;
        END_IF;
```



```
    ELSIF SktGetTCPStatus_instance.Error = TRUE THEN
        Exec_Status:=FALSE;
        Stage:=5;
    END_IF;

4:
    // SLMP_DeviceWriteWord
    Exec_Write:=TRUE;
    IF SLMP_DeviceWriteWord_instance.Done = TRUE THEN
        Exec_Write:=FALSE;
        Stage:=5;
    ELSIF SLMP_DeviceWriteWord_instance.Error = TRUE THEN
        Exec_Write:=FALSE;
        Stage:=5;
    END_IF;

5:
    // Cut the socket.
    Exec_Close:=TRUE;
    IF SktClose_instance.Done = TRUE OR SktClose_instance.Error = TRUE THEN
        Exec_Close:=FALSE;
        Trigger:=FALSE;
        Stage:=0;
    END_IF;
END_CASE;
```





ST

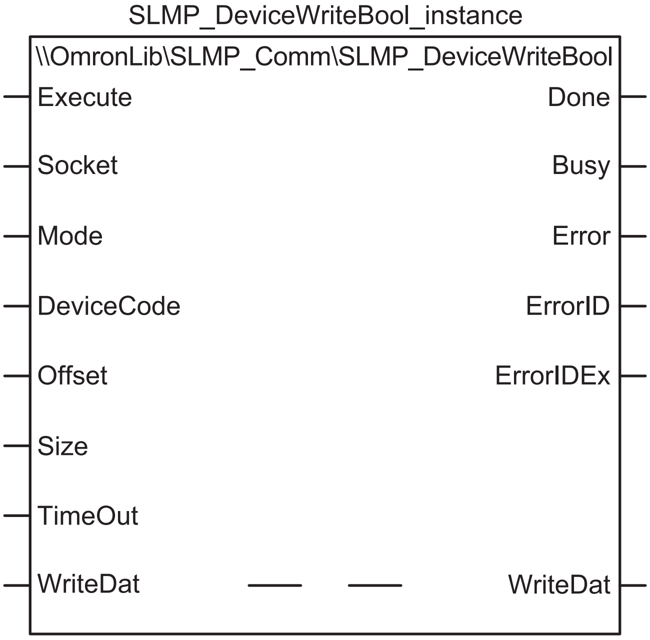
```
IF (Trigger = TRUE) AND (_EIP_EtnOnlineSta=TRUE) THEN
   CASE Stage OF
   0:
      // Socket setting.
      SrcTCPPort:=0;
      DstPAdr:='192.168.250.39';
      DstTcpPort:=5100;
      // Setting for creating a message.
      Mode:=FALSE;        // Binary
//    Mode:=TRUE;         // ASCII
```

```
   DeviceCode:=4;
   Offset:=100;
   Size:=2;
   TimeOut:=0;
   WriteDat[1]:=WORD#16#0123;
   WriteDat[2]:=WORD#16#4567;
   Stage:=1;

1:
   // Connect to the socket.
   Exec_Connect:=TRUE;
   IF SktTCPConnect_instance.Done = TRUE THEN
      Exec_Connect:=FALSE;
      Stage:=2;
   ELSIF SktTCPConnect_instance.Error = TRUE THEN
      Exec_Connect:=FALSE;
      Trigger:=FALSE;
      Stage:=0;
   END_IF;

2:
   // Clear receive buffer.
   Exec_ClrBuf:=TRUE;
   IF SktClearBuf_instance.Done = TRUE THEN
      Exec_ClrBuf:=FALSE;
      Stage:=3;
   ELSIF SktClearBuf_instance.Error = TRUE THEN
      Exec_ClrBuf:=FALSE;
      Stage:=5;
   END_IF;

3:
   // Confirm connection Status.
   Exec_Status:=TRUE;
   IF SktGetTCPStatus_instance.Done = TRUE THEN
      Exec_Status:=FALSE;
      IF TcpStatus = _ESTABLISHED THEN
         Stage:=4;
      ELSE
         Stage:=5;
      END_IF;
   ELSIF SktGetTCPStatus_instance.Error = TRUE THEN
      Exec_Status:=FALSE;
      Stage:=5;
   END_IF;

4:
      // SLMP_DeviceWriteWord
      Exec_Write:=TRUE;
      IF SLMP_DeviceWriteWord_instance.Done = TRUE THEN
         Exec_Write:=FALSE;
         Stage:=5;
      ELSIF SLMP_DeviceWriteWord_instance.Error = TRUE THEN
         Exec_Write:=FALSE;
         Stage:=5;
      END_IF;

   5:
      // Cut the socket.
      Exec_Close:=TRUE;
      IF SktClose_instance.Done = TRUE OR SktClose_instance.Error = TRUE THEN
         Exec_Close:=FALSE;
         Trigger:=FALSE;
         Stage:=0;
      END_IF;
   END_CASE;
ELSE
   Exec_Connect:=FALSE;
   Exec_ClrBuf:=FALSE;
   Exec_Status:=FALSE;
   Exec_Write:=FALSE;
   Exec_Close:=FALSE;
END_IF;

// FB execution.
SktTCPConnect_instance(
   Execute:=Exec_Connect,
   SrcTcpPort:=SrcTCPPort,
   DstAdr:=DstPAdr,
   DstTcpPort:=DstTcpPort,
   Socket=>Socket
   );
SktClearBuf_instance(
   Execute:=Exec_ClrBuf,
   Socket:=Socket
   );
SktGetTCPStatus_instance(
   Execute:=Exec_Status,
   Socket:=Socket,
   TcpStatus=>TcpStatus
   );
SLMP_DeviceWriteWord_instance(
   Execute:=Exec_Write,
   Socket:=Socket,
      Mode:=Mode,
      DeviceCode:=DeviceCode,
      Offset:=Offset,
      Size:=Size,
      TimeOut:=TimeOut,
      WriteDat:=WriteDat
      );
   SktClose_instance(
      Execute:=Exec_Close,
      Socket:=Socket
      );
```

SLMP_DeviceWriteBool

Use SLMP communications to write in bit units to the sequencer device made by Mitsubishi Electric.

Function block name	Name	FB/ FUN	Graphic expression	ST expression

SLMP_DeviceWriteBool	Internal memory batch write (Bit)	FB	 SLMP_DeviceWriteBool_instance
			\\OmronLib\SLMP_Comm\SLMP_DeviceWriteBool
			Execute
			Socket
			Mode
			DeviceCode
			Offset
			Size
			TimeOut
			WriteDat
			Done
			Busy
			Error
			ErrorID
			ErrorIDEx
			WriteDat
				SLMP_DeviceWriteBool_instance(
				Execute:=,
				Socket:=,
				Mode:=,
				DeviceCode:=,
				Offset:=,
				Size:=,
				TimeOut:=,
				Done=>,
				Busy=>,
				Error=>,
				ErrorID=>,
				ErrorIDEx=>,
				WriteDat:=);

Function Block and Function Information

Item	Description
Library file name	OmronLib_SLMP_Comm_Vx_x.slr (x shows the unit version.)
Namespace	OmronLib\SLMP_Comm
Function block and function number	00171
Source code published/not published	Not Published

Input Variables

Variable	Meaning	Data type	Description	Valid range	Unit	Default
Execute	Execute	BOOL	TRUE: Execute FALSE: Do not execute	TRUE, FALSE	—	FALSE
Socket	Socket	_sSOCKET[*1]	Socket	—	—	—
Mode	Mode	BOOL	SLMP frame communicating code TRUE: ASCII mode FALSE: BINARY mode	TRUE, FALSE	—	FALSE

Variable	Meaning	Data type	Description	Valid range	Unit	Default
Device-Code	Device code	UINT	Specifies the device code for the write target. Details are listed in *2-3-1 Device Code (Device Access for Sequencer CPU Made by Mitsubishi Electric)* on page 2 - 4.	Depends on data type.	—	UINT#0
Offset	First device No.	UDINT	Specifies the write device first No.	Depends on data type.	—	UDINT#0
Size	Number of device points	UINT	Specifies the write device number.	Depends on data type.	—	UINT#0
TimeOut	Timeout time	UINT	The timeout time for FB For "0", 2.0s	Depends on data type.	0.1s	UINT#0

*1. Refer to the SktTCPConnect instructions in the *NJ/NX-series Instructions Reference Manual (W502)* or *NY-series Instructions Reference Manual (W560)* for details on the Data Type function block.

Output Variables

Variable	Meaning	Data type	Description	Valid range	Unit	Default
Done	Done	BOOL	TRUE: Normal end FALSE: Error end, execution in progress, or execution condition not met	TRUE, FALSE	—	—
Busy	Executing	BOOL	TRUE: Executing. FALSE: Not executing.	TRUE, FALSE	—	—
Error	Error	BOOL	TRUE: Error end. FALSE: Normal end, executing, or execution condition not met	TRUE, FALSE	—	—
ErrorID	Error Code	WORD	This is the error ID for an error end. The value is 16#0 for a normal end.	*1	—	—
ErrorIDEx	Expansion Error Code	DWORD	This is the error ID for an Expansion Unit Hardware Error. The value is 16#0 for a normal end.	*1	—	—

*1. Refer to *Troubleshooting* on page 4 - 34 for details.

Input-Output Variables

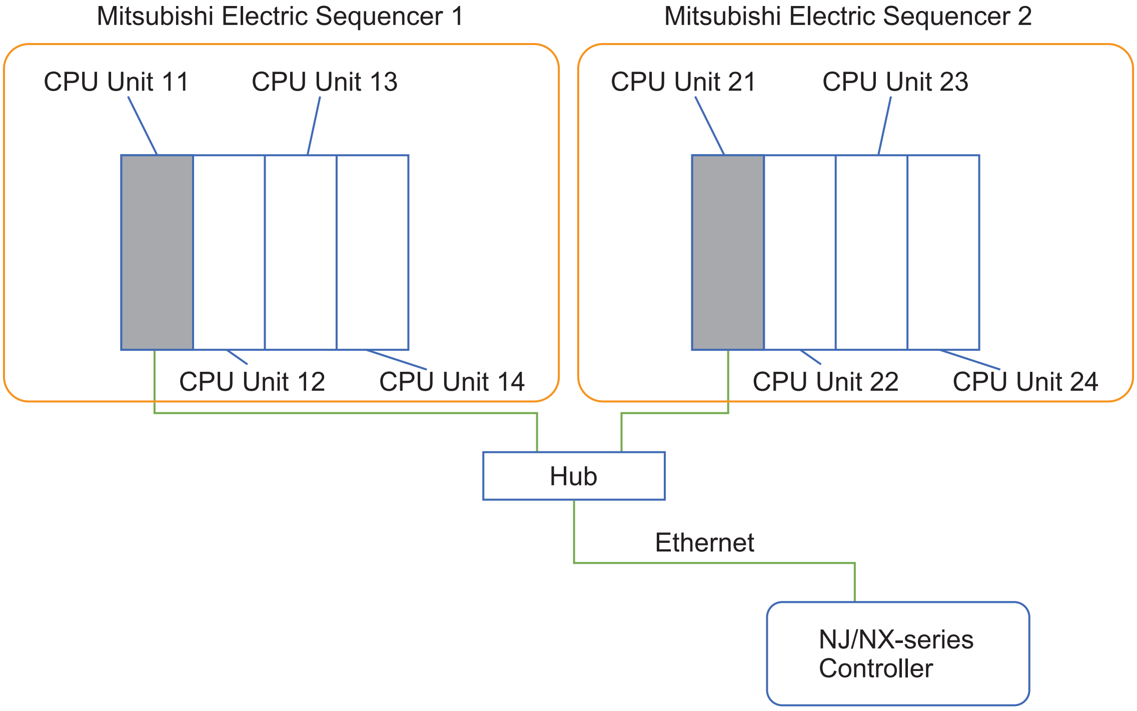
Variable	Meaning	Data type	Description	Valid range	Unit	Default
WriteDat	Write data	ARRAY[*] OF BOOL [*1]	Stores the write data. [*2]	Depends on data type.	—	—

[*1] You can input any desired Bool array length. Array length is in the range of 1 - 3840 (when in BINARY mode), 1 - 1920 (when in ASCII mode). Basic type Bool cannot be input. Bool arrays can start with either element 0 or with element n.

[*2] It must be stored in the communicating code specified in Mode. Conversion from BINARY to ASCII and conversion from ASCII to BINARY cannot be performed.

Function

- Use SLMP communications to write in bit units to the sequencer device made by Mitsubishi Electric on the connected station.
- In the image below, the Mitsubishi Electric Sequencers that can write-out are CPU Unit 11 and CPU Unit 21. CPU Units 12 to 14 and CPU Units 22 to 24 do not support write-out.



- Connect the device specified by the input variable *DeviceCode* from the input variable *Offset*, and write the input variable *Size* amount.
- The SLMP request message is based on the input variable *Mode*, and created using Write (Command: 1401, Sub-command: 0001).
- Write data is stored in the input/output variable *WriteDat*.
- The Offset range is as follows.

Device No. range	Mode	
	FALSE BINARY mode 3 bytes	TRUE ASCII mode 6 bytes
Hexadecimal	'0' - '16777215 (0xFFFFFF)'	'0' - 'FFFFFF'

Device No. range	Mode	
	FALSE BINARY mode 3 bytes	TRUE ASCII mode 6 bytes
Decimal	Same as above (For Decimal → Hexadecimal conversion)	'0' - '999999'

- The possible input range for the input variable *Size*, based on the Input variable *Mode*, and the SLMP request message write data length, are as shown below.
  When Size is set to 0, the command is completed (Done=TRUE) without a command being sent.

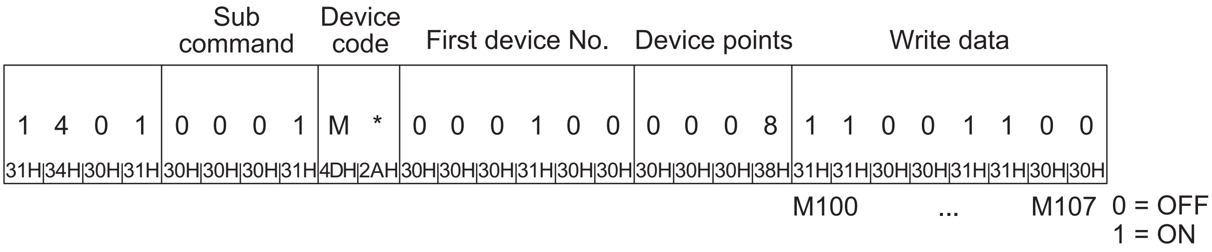
Item name	Mode	
	FALSE BINARY mode	TRUE ASCII mode
Size	1 - 3840 points	1 - 1920 points
SLMP request message write data length	1 - 1920 bytes	1 - 1920 bytes

Communicating Example (when writing in bit units)

Write the value from M100 to M107.

● When Data Communications are in ASCII Code

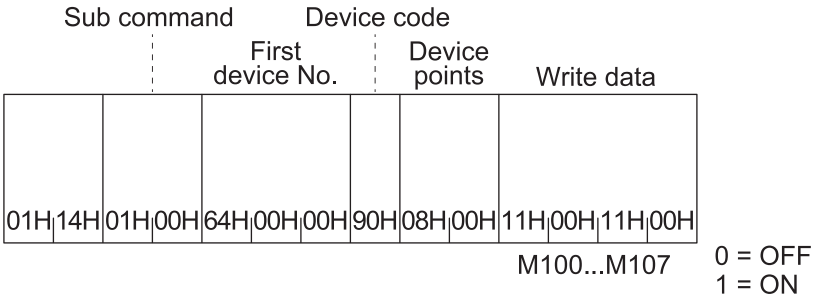
(Request data)



	Sub command	Device code	First device No.	Device points	Write data
1 4 0 1	0 0 0 1	M *	0 0 0 1 0 0	0 0 0 8	1 1 0 0 1 1 0 0
31H 34H 30H 31H	30H 30H 30H 31H	4DH 2AH	30H 30H 30H 31H 30H 30H	30H 30H 30H 38H	31H 31H 30H 30H 31H 31H 30H 30H

● When Data Communications are in Binary Code

(Request data)



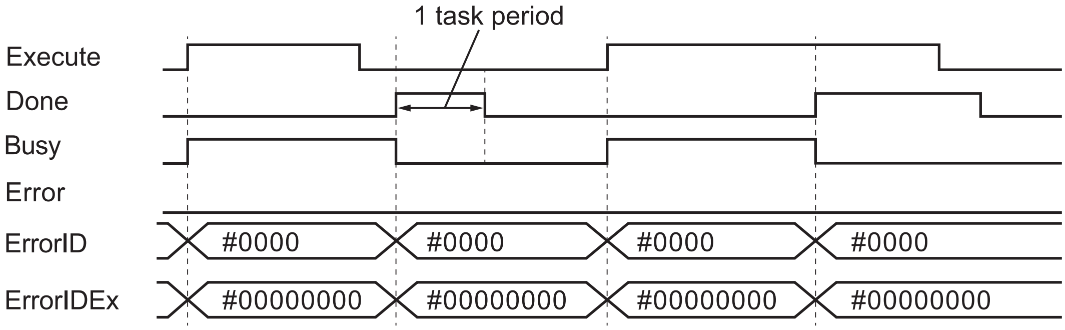
	Sub command	First device No.	Device code	Device points	Write data
01H 14H	01H 00H	64H 00H 00H	90H	08H 00H	11H 00H 11H 00H

Timing Chart

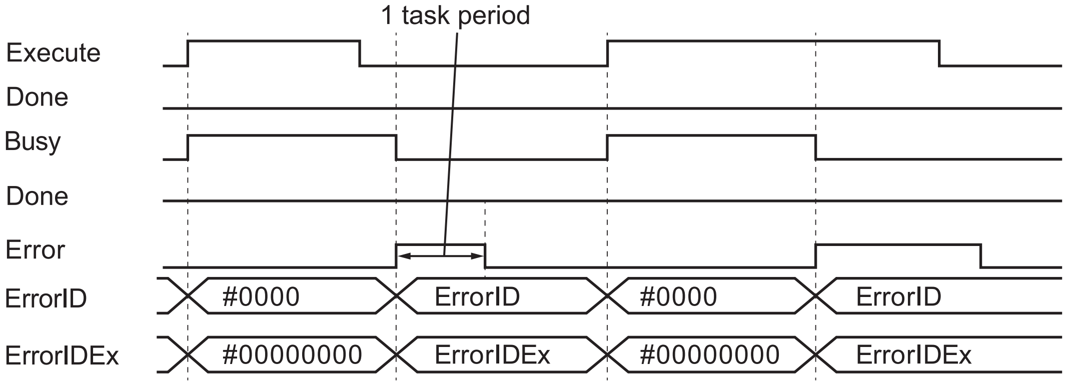
The timing charts are shown below.

- *Busy* (Executing) changes to TRUE when *Execute* (Execute) changes to TRUE.
- When a response message from the Mitsubishi Electric Sequencer is received successfully, and EndCode=0, *Done* changes to TRUE.
- If an error occurs when execution of the function block is in progress, *Error* changes to TRUE and *Busy* (Executing) changes to FALSE.
  You can find out the cause of the error by accessing the values output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code).
- If *Execute* (Execute) changes to FALSE before execution of the FB is ended, *Done* and *Error* are TRUE only for one task period after execution of the FB is ended.
- If *Execute* (Execute) remains TRUE even after execution of the function block is ended, the output values of *Done* and *Error* are retained.

- Timing Chart for Normal End



- Timing Chart for Error End



Additional Information

- For this FB, use the socket service function. Refer to *NJ/NX-series CPU Unit Built-in EtherNet/IP™ Port User's Manual (W506)* or *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Built-in EtherNet/IP™ Port User's Manual (W563)* for details of the socket service function.

Precautions for Correct Use

- Execution of this function block will be continued until processing is ended even if the value of *Execute* changes to FALSE or the execution time exceeds the task period. The value of *Done* changes to TRUE when processing is ended. Use this to confirm normal ending of processing.

- The FB performs one execution for the SktTCPRcv command and the SktTCPSend command, respectively. For the number of simultaneous executions, refer to the *NJ/NX-series Instructions Reference Manual (W502)* or *NY-series Instructions Reference Manual (W560)* SktTCPRcv command and SktTCPSend command.

Precautions for Correct Use

- Refer to *2-2 Usage Method* on page 2 - 3.
- If "Write during RUN" is prohibited on the Mitsubishi Electric Sequencer side, confirm that the Mitsubishi Electric Sequencer is in STOP state, and then execute.

Troubleshooting

Error code	Expansion error code	Status	Description	Corrective action
16#0000	16#00000000	Normal end	—	—
16#2003	16#00000000	Socket status error	Status at time of socket service instruction execution is not suitable.	Refer to the *NJ/NX-series Instructions Reference Manual (W502)* or *NY-series Instructions Reference Manual (W560)*, and the *NJ/NX-series CPU Unit Built-in EtherNet/IP™ Port User's Manual (W506)* or *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Built-in EtherNet/IP™ Port User's Manual (W563)* and perform troubleshooting.
16#2006	16#00000000	Socket timeout	In socket service instruction, a timeout occurred.	
16#2007	16#00000000	Illegal socket handle	The handle specified in socket service instruction is illegal.	
16#2008	16#00000000	Socket communications resource overflow	Instructions were executed in excess of the socket service instruction resources that can be simultaneously executed.	
16#3CEC	16#00000001	Illegal device code	This function block specified an unsupported device code.	Check the FB input variable *DeviceCode*.
16#3CEC	16#00000002	Illegal first device No.	This FB specified an unsupported first device No.	Check the FB input variable *Offset*.
16#3CEC	16#00000003	Illegal number of device points	Using the input variable *Size*, the write data size exceeded the controller specification upper limit value.	Check the FB input variable *Size*.
16#3CEC	16#00000004	Insufficient WriteDat size	Array size allocated to WriteDat is less than the write data length specified in *Mode*, *Size*.	Check the FB input variables *Mode* and *Size*. Or, check the buffer size allocated to *WriteDat*.
16#3CEC	16#00000005	WriteDat upper limit exceeded	The array size allocated to *WriteDat* exceeds the upper limit.	Check that the array size allocated to *WriteDat* is 1 - 3840 (when in BINARY mode), 1 - 1920 (when in ASCII mode).

Error code	Expansion error code	Status	Description	Corrective action
16#3CEC	16#00000010	Response reception timeout	A response message could not be received within the time specified in *TimeOut*.	Check the network status and communications target Mitsubishi Electric Sequencer status. Or, increase the timeout value.
16#3CEC	16#FFFF0001 - 16#FFFFFFFF	SLMP end code	An error response message was received.	Reference the SLMP regulations, and perform troubleshooting.

Sample Programming

Precautions for Correct Use

- The sample programming shows only the portion of a program that uses the function or function block from the library.
- When programming actual applications, also program safety circuits, device interlocks, I/O with other devices, and other control procedures.
- Create a user program that will produce the intended device operation.
- Check the user program for proper execution before you use it for actual operation.

This is an example of a program writing a value from M100 to M107. It can be used with the MELSEC iQ-R Series, iQ-F Series, Q Series, and L Series.

Conditions

- The timeout time is 2 s (default).

Device Write Process

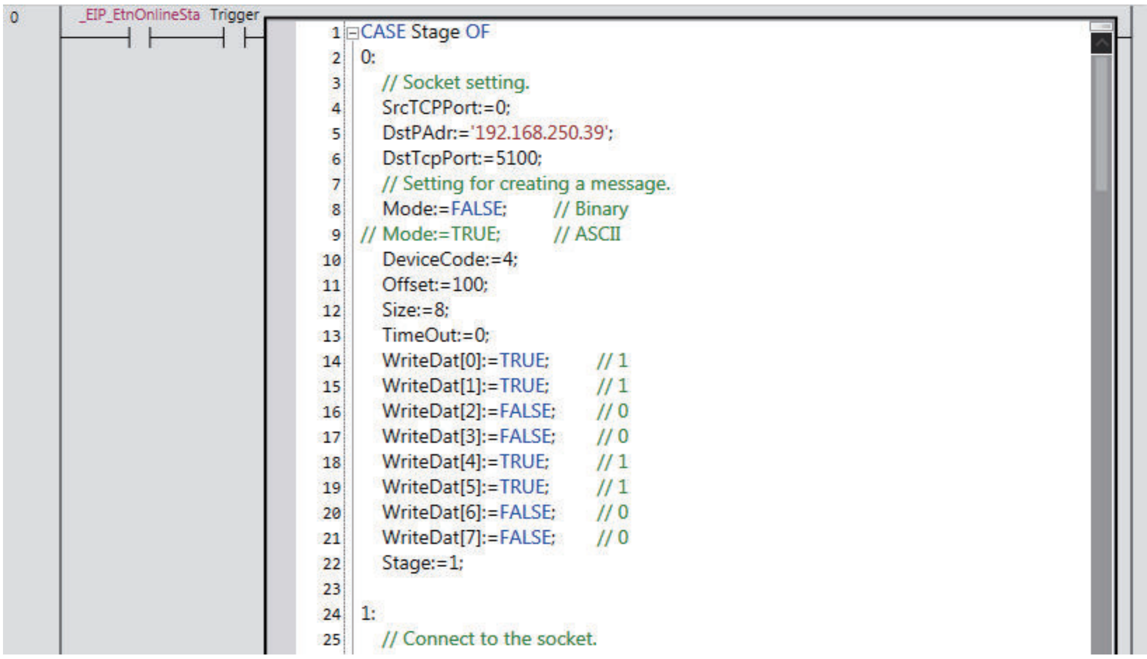
1. Execute the SktTCPConnect command, and connect to the partner TCP port.
2. Execute the SktClearBuf command, and clear the TCP socket receiving buffer.
3. Confirm that no other communications FB execution is in progress.
4. Start up the SLMP_DeviceWriteBool_instance.
5. Execute the SktClose command, and close the TCP socket.

Main Variables

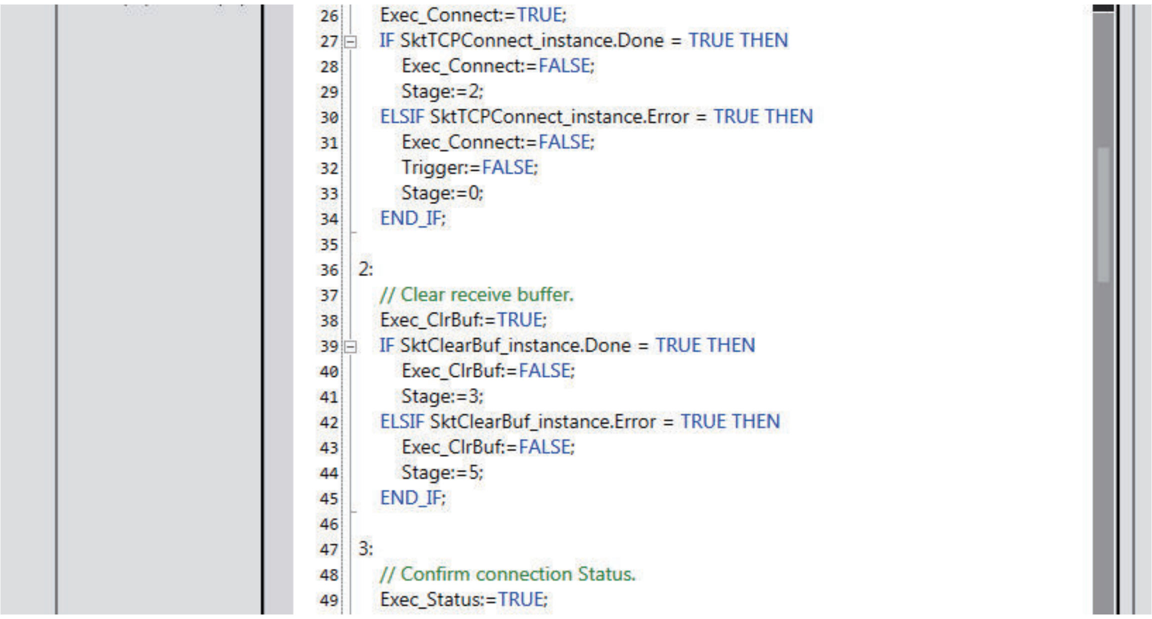
Variable	Data type	Default
Trigger	BOOL	FALSE
Stage	USINT	0
SktTCPConnect_instance	SktTCPConnect	
SktClearBuf_instance	SktClearBuf	
SktGetTCPStatus_instance	SktGetTCPStatus	
SLMP_DeviceWriteBool_instance	OmronLib\SLMP_Comm\SLMP_DeviceWriteBool	

Variable	Data type	Default
SktClose_instance	SktClose	
Exec_Connect	BOOL	FALSE
Exec_ClrBuf	BOOL	FALSE
Exec_Status	BOOL	FALSE
Exec_Read	BOOL	FALSE
Exec_Close	BOOL	FALSE
WriteDat	ARRAY[0..7] OF BOOL	
TcpStatus	_eCONNECTION_STATE	
Mode	BOOL	
DeviceCode	UINT	
Offset	UDINT	
Size	UINT	
TimeOut	UINT	
SrcTCPPort	UINT	
DstPAdr	STRING[64]	
DstTcpPort	UINT	
Socket	_sSOCKET	

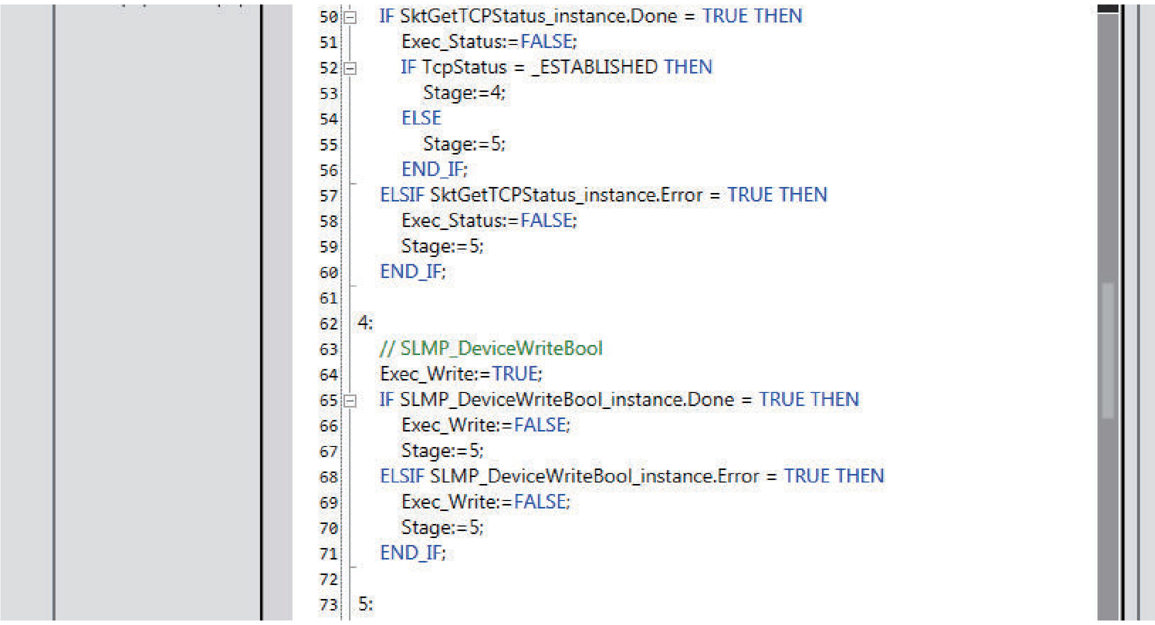
Ladder Diagram



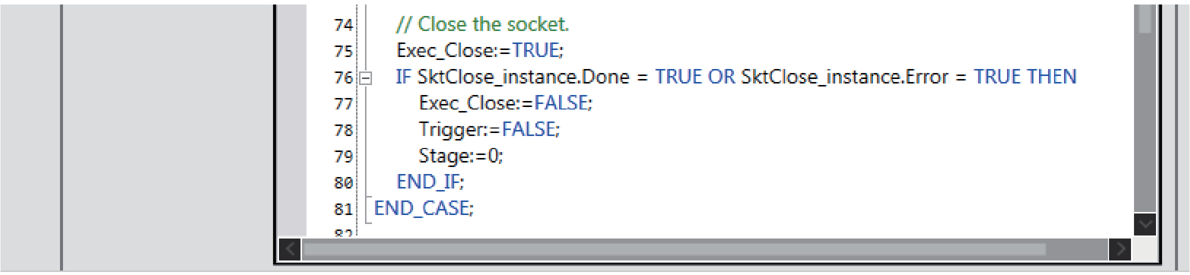
```
CASE Stage OF
0:
    // Socket setting.
    SrcTCPPort:=0;
    DstPAdr:='192.168.250.39';
    DstTcpPort:=5100;
    // Setting for creating a message.
    Mode:=FALSE;          // Binary
// Mode:=TRUE;            // ASCII
    DeviceCode:=4;
    Offset:=100;
    Size:=8;
    TimeOut:=0;
    WriteDat[0]:=TRUE;        // 1
    WriteDat[1]:=TRUE;        // 1
    WriteDat[2]:=FALSE;       // 0
    WriteDat[3]:=FALSE;       // 0
    WriteDat[4]:=TRUE;        // 1
    WriteDat[5]:=TRUE;        // 1
    WriteDat[6]:=FALSE;       // 0
    WriteDat[7]:=FALSE;       // 0
    Stage:=1;

1:
    // Connect to the socket.
```



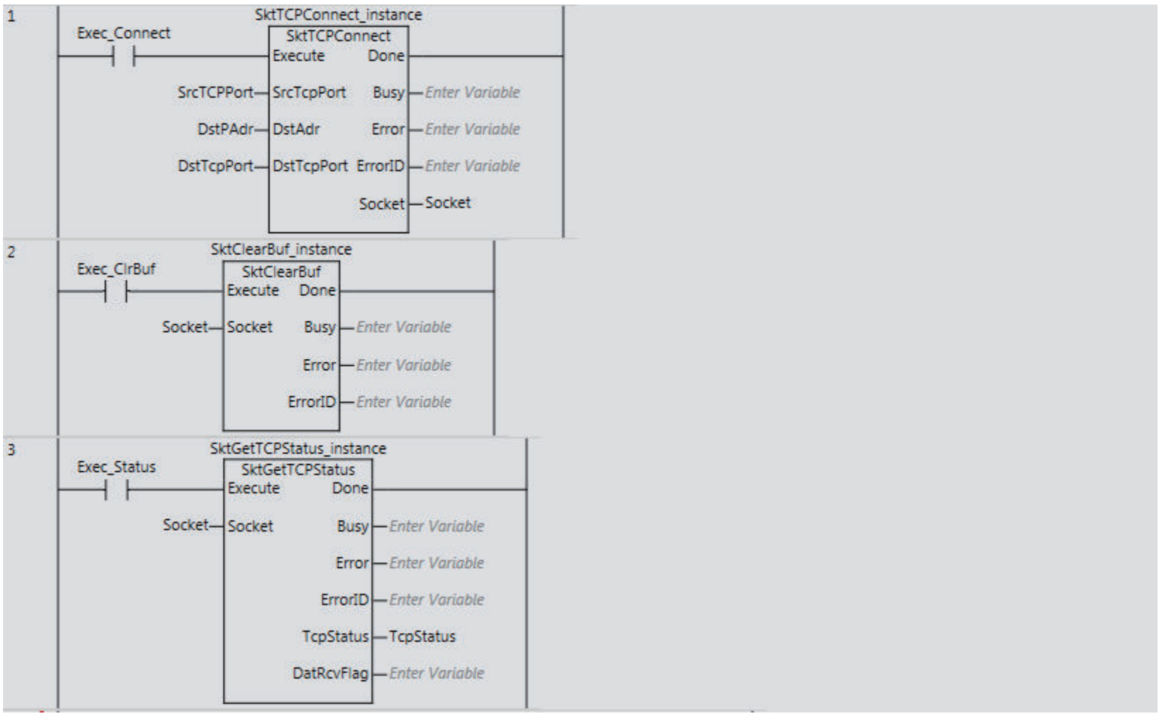
```
    Exec_Connect:=TRUE;
    IF SktTCPConnect_instance.Done = TRUE THEN
        Exec_Connect:=FALSE;
        Stage:=2;
    ELSIF SktTCPConnect_instance.Error = TRUE THEN
        Exec_Connect:=FALSE;
        Trigger:=FALSE;
        Stage:=0;
    END_IF;

2:
    // Clear receive buffer.
    Exec_ClrBuf:=TRUE;
    IF SktClearBuf_instance.Done = TRUE THEN
        Exec_ClrBuf:=FALSE;
        Stage:=3;
    ELSIF SktClearBuf_instance.Error = TRUE THEN
        Exec_ClrBuf:=FALSE;
        Stage:=5;
    END_IF;

3:
    // Confirm connection Status.
    Exec_Status:=TRUE;
```

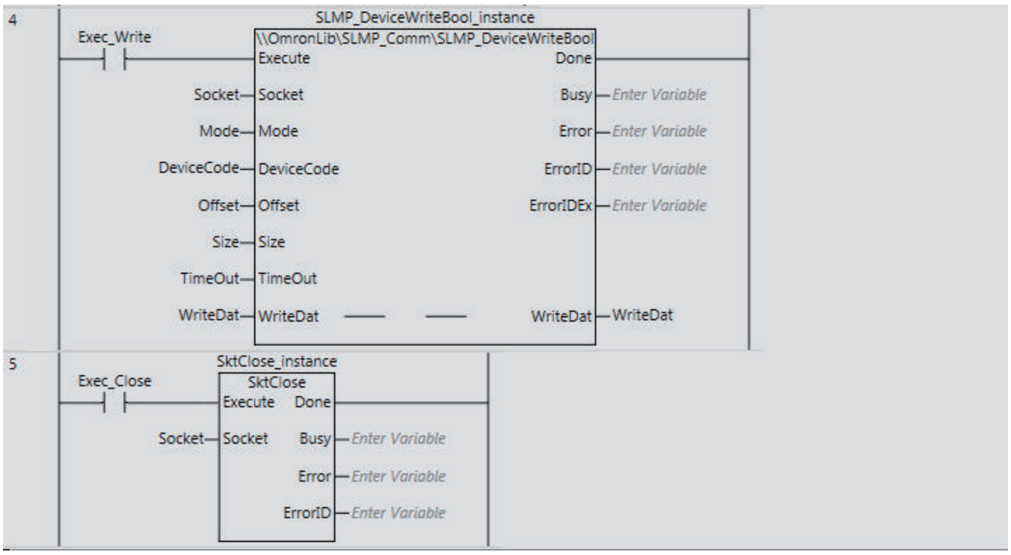


```
    IF SktGetTCPStatus_instance.Done = TRUE THEN
        Exec_Status:=FALSE;
        IF TcpStatus = _ESTABLISHED THEN
            Stage:=4;
        ELSE
            Stage:=5;
        END_IF;
    ELSIF SktGetTCPStatus_instance.Error = TRUE THEN
        Exec_Status:=FALSE;
        Stage:=5;
    END_IF;

4:
    // SLMP_DeviceWriteBool
    Exec_Write:=TRUE;
    IF SLMP_DeviceWriteBool_instance.Done = TRUE THEN
        Exec_Write:=FALSE;
        Stage:=5;
    ELSIF SLMP_DeviceWriteBool_instance.Error = TRUE THEN
        Exec_Write:=FALSE;
        Stage:=5;
    END_IF;

5:
```



```
    // Close the socket.
    Exec_Close:=TRUE;
    IF SktClose_instance.Done = TRUE OR SktClose_instance.Error = TRUE THEN
        Exec_Close:=FALSE;
        Trigger:=FALSE;
        Stage:=0;
    END_IF;
END_CASE;
```





ST

```
IF (Trigger = TRUE) AND (_EIP_EtnOnlineSta=TRUE) THEN
   CASE Stage OF
   0:
      // Socket setting.
      SrcTCPPort:=0;
      DstPAdr:='192.168.250.39';
      DstTcpPort:=5100;
      // Setting for creating a message.
      Mode:=FALSE;             // Binary
//    Mode:=TRUE;              // ASCII
```

```
   DeviceCode:=4;
   Offset:=100;
   Size:=8;
   TimeOut:=0;
   WriteDat[0]:=TRUE;         // 1
   WriteDat[1]:=TRUE;         // 1
   WriteDat[2]:=FALSE;        // 0
   WriteDat[3]:=FALSE;        // 0
   WriteDat[4]:=TRUE;         // 1
   WriteDat[5]:=TRUE;         // 1
   WriteDat[6]:=FALSE;        // 0
   WriteDat[7]:=FALSE;        // 0
   Stage:=1;

1:
   // Connect to the socket.
   Exec_Connect:=TRUE;
   IF SktTCPConnect_instance.Done = TRUE THEN
      Exec_Connect:=FALSE;
      Stage:=2;
   ELSIF SktTCPConnect_instance.Error = TRUE THEN
      Exec_Connect:=FALSE;
      Trigger:=FALSE;
      Stage:=0;
   END_IF;

2:
   // Clear receive buffer.
   Exec_ClrBuf:=TRUE;
   IF SktClearBuf_instance.Done = TRUE THEN
      Exec_ClrBuf:=FALSE;
      Stage:=3;
   ELSIF SktClearBuf_instance.Error = TRUE THEN
      Exec_ClrBuf:=FALSE;
      Stage:=5;
   END_IF;

3:
   // Confirm connection Status.
   Exec_Status:=TRUE;
   IF SktGetTCPStatus_instance.Done = TRUE THEN
      Exec_Status:=FALSE;
      IF TcpStatus = _ESTABLISHED THEN
         Stage:=4;
      ELSE
         Stage:=5;
      END_IF;
```

```
            ELSIF SktGetTCPStatus_instance.Error = TRUE THEN
                Exec_Status:=FALSE;
                Stage:=5;
            END_IF;

        4:
            // SLMP_DeviceWriteBool
            Exec_Write:=TRUE;
            IF SLMP_DeviceWriteBool_instance.Done = TRUE THEN
                Exec_Write:=FALSE;
                Stage:=5;
            ELSIF SLMP_DeviceWriteBool_instance.Error = TRUE THEN
                Exec_Write:=FALSE;
                Stage:=5;
            END_IF;

        5:
            // Close the socket.
            Exec_Close:=TRUE;
            IF SktClose_instance.Done = TRUE OR SktClose_instance.Error = TRUE THEN
                Exec_Close:=FALSE;
                Trigger:=FALSE;
                Stage:=0;
            END_IF;
        END_CASE;
    ELSE
        Exec_Connect:=FALSE;
        Exec_ClrBuf:=FALSE;
        Exec_Status:=FALSE;
        Exec_Write:=FALSE;
        Exec_Close:=FALSE;
    END_IF;

    // FB execution.
    SktTCPConnect_instance(
        Execute:=Exec_Connect,
        SrcTcpPort:=SrcTCPPort,
        DstAdr:=DstPAdr,
        DstTcpPort:=DstTcpPort,
        Socket=>Socket
        );
    SktClearBuf_instance(
        Execute:=Exec_ClrBuf,
        Socket:=Socket
        );
    SktGetTCPStatus_instance(
        Execute:=Exec_Status,
   Socket:=Socket,
   TcpStatus=>TcpStatus
   );
SLMP_DeviceWriteBool_instance(
   Execute:=Exec_Write,
   Socket:=Socket,
   Mode:=Mode,
   DeviceCode:=DeviceCode,
   Offset:=Offset,
   Size:=Size,
   TimeOut:=TimeOut,
   WriteDat:=WriteDat
   );
SktClose_instance(
   Execute:=Exec_Close,
   Socket:=Socket
   );
```

A

Appendix

This section describes information that is convenient to know, such as library information reference methods, FB or FUN source code reference methods, etc.

A-1 Referring to Library Information

When you make an inquiry to OMRON about a library, you can refer to the library information to identify the library to ask about.

The library information is useful in identifying the target library among the libraries provided by OMRON or created by the user.

The library information consists of the attributes of the library and the attributes of function blocks and functions contained in the library.

- Attributes of libraries
  Information for identifying the library itself
- Attributes of function blocks and functions
  Information for identifying the function block and function contained in the library

Use the Sysmac Studio to access the library information.

A-1-1 Library Attributes, and FB or FUN Attributes

The following attributes of libraries, function blocks, and functions are provided as library information.

Library Attributes

No.[*1]	Attribute	Description
(1)	Library file name	The name of the library file
(2)	Library version	The version of the library
(3)	Author	The name of the creator of the library
(4)	Comment	The description of the library[*2]

*1. These numbers correspond to the numbers shown on the screen images in the next section, *A-1-2 Referring to Attributes of Libraries, Function Blocks, and Functions* on page A - 3.

*2. It is provided in English and Japanese.

Attributes of Function Blocks and Functions

No.[*1]	Attribute	Description
(5)	FB/FUN name	The name of the function block or function
(6)	Name space	The name of the name space for the function block or function
(7)	FB/FUN version	The version of the function block or function
(8)	Author	The name of the creator of the function block or function
(9)	FB/FUN number	The function block number or function number
(10)	Comment	The description of the function block or function [*2]

*1. These numbers correspond to the numbers shown on the screen images in the next section, *A-1-2 Referring to Attributes of Libraries, Function Blocks, and Functions* on page A - 3.

*2. It is provided in English and Japanese.

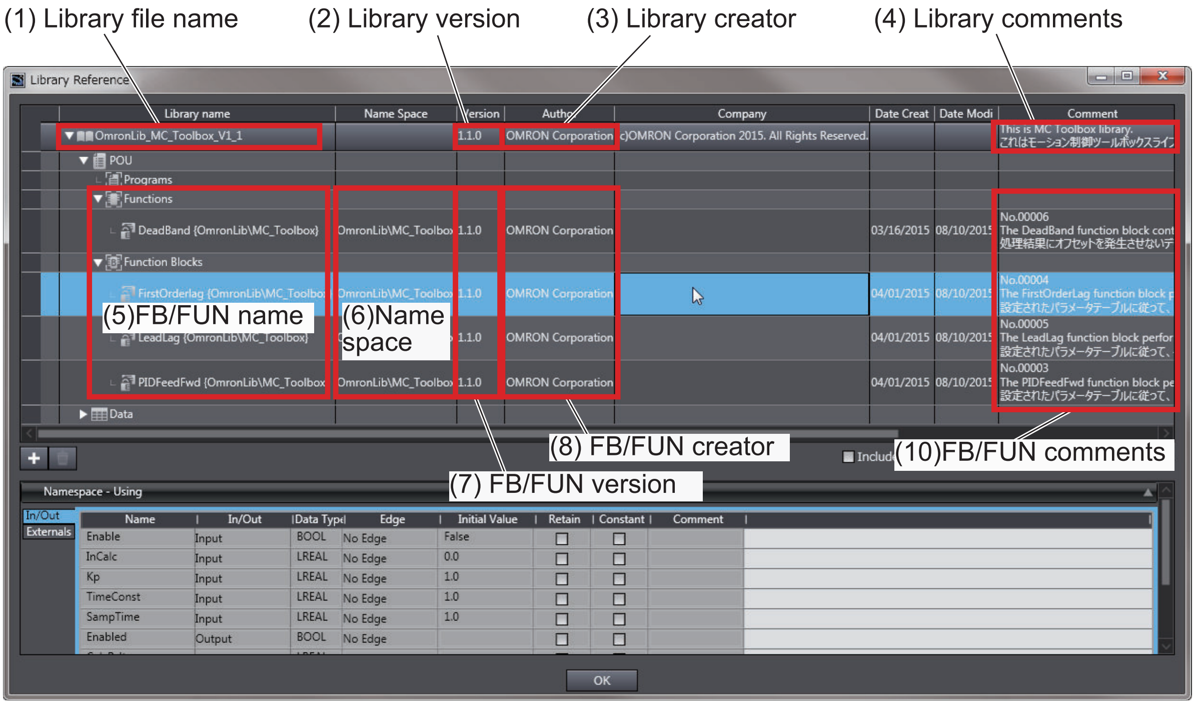
A-1-2 Referring to Attributes of Libraries, Function Blocks, and Functions

You can refer to the library attributes of library information, and FB or FUN attributes at the following Sysmac Studio locations.

- Library Reference Dialog Box
- Toolbox
- Programming screen

Library Reference Dialog Box

When you refer to the libraries, the library information is displayed at the locations shown below.

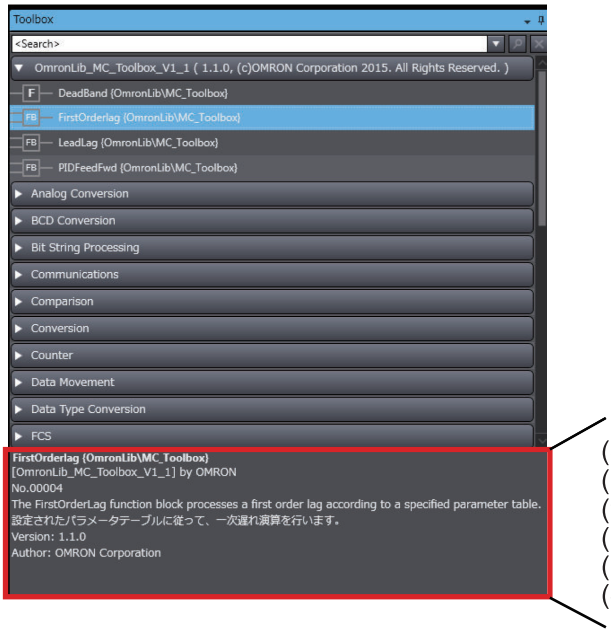


Library name	Name Space	Version	Author	Company	Date Creat	Date Modi	Comment
OmronLib_MC_Toolbox_V1_1		1.1.0	OMRON Corporation	c)OMRON Corporation 2015. All Rights Reserved.			This is MC Toolbox library. これはモーション制御ツールボックスライブ
POU							
Programs							
Functions							
DeadBand {OmronLib\MC_Toolbox}	OmronLib\MC_Toolbox	1.1.0	OMRON Corporation		03/16/2015	08/10/2015	No.00006 The DeadBand function block cont 処理結果にオフセットを発生させないデ
Function Blocks							
FirstOrderlag {OmronLib\MC_Toolbox}	OmronLib\MC_Toolbox	1.1.0	OMRON Corporation		04/01/2015	08/10/2015	No.00004 The FirstOrderLag function block p 設定されたパラメータテーブルに従って、
LeadLag {OmronLib\MC_Toolbox}		1.1.0	OMRON Corporation		04/01/2015	08/10/2015	No.00005 The LeadLag function block perfor 設定されたパラメータテーブルに従って、
PIDFeedFwd {OmronLib\MC_Toolbox	OmronLib\MC_Toolbox	1.1.0	OMRON Corporation		04/01/2015	08/10/2015	No.00003 The PIDFeedFwd function block pe 設定されたパラメータテーブルに従って、
Data							

Name	In/Out	Data Type	Edge	Initial Value	Retain	Constant	Comment
Enable	Input	BOOL	No Edge	False	□	□	
InCalc	Input	LREAL	No Edge	0.0	□	□	
Kp	Input	LREAL	No Edge	1.0	□	□	
TimeConst	Input	LREAL	No Edge	1.0	□	□	
SampTime	Input	LREAL	No Edge	1.0	□	□	
Enabled	Output	BOOL	No Edge		□	□	

Toolbox

Select a function block or function to display its library information at the bottom of the Toolbox Pane. The text "**by OMRON"** which is shown on the right of the library name (1) indicates that this library was provided by OMRON.

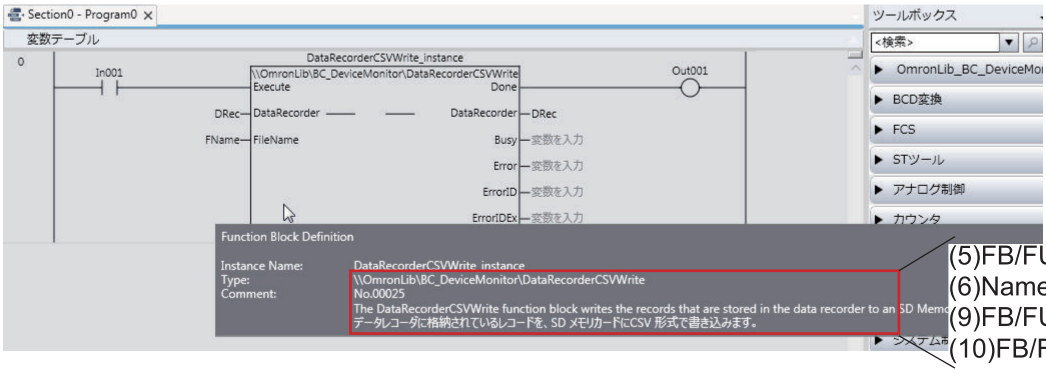

Toolbox
<Search>
OmronLib_MC_Toolbox_V1_1 (1.1.0, (c)OMRON Corporation 2015. All Rights Reserved.)
DeadBand (OmronLib\MC_Toolbox)
FirstOrderlag (OmronLib\MC_Toolbox)
LeadLag (OmronLib\MC_Toolbox)
PIDFeedFwd (OmronLib\MC_Toolbox)
Analog Conversion
BCD Conversion
Bit String Processing
Communications
Comparison
Conversion
Counter
Data Movement
Data Type Conversion
FCS

FirstOrderlag (OmronLib\MC_Toolbox)
[OmronLib_MC_Toolbox_V1_1] by OMRON
No.00004
The FirstOrderLag function block processes a first order lag according to a specified parameter table.
設定されたパラメータテーブルに従って、一次遅れ演算を行います。
Version: 1.1.0
Author: OMRON Corporation

(5)FB/FUN name (6)Name space
(1)Library file name
(9)FB/FUN number
(10)FB/FUN comment
(7)FB/FUN version
(8)FB/FUN author

Programming Screen

Place the mouse on a function block and function to display the library information in a tooltip.



Function Block Definition

Instance Name: DataRecorderCSVWrite_instance
Type: \\OmronLib\BC_DeviceMonitor\DataRecorderCSVWrite
Comment: No.00025
The DataRecorderCSVWrite function block writes the records that are stored in the data recorder to an SD Memo
データレコーダに格納されているレコードを、SD メモリカードにCSV 形式で書き込みます。

(5)FB/FUN name
(6)Name space
(9)FB/FUN number
(10)FB/FUN comment

A-2 Referring to Function Block and Function Source Codes

You can refer to the source codes of function blocks and functions provided by OMRON to customize them to suit the user's environment.

User function blocks and user functions can be created based on the copies of these source codes.

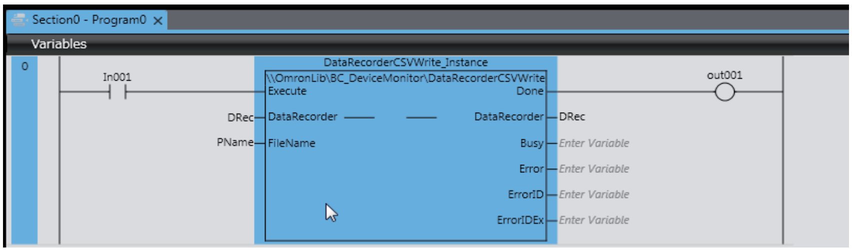
The following are the examples of items that you may need to customize.

- Customizing the "Array Size" to suit the memory capacity of the user's Controller
- Customizing the "Data Type" to suit the user-defined data types

Note that you can access only function blocks and functions whose Source code published/not published is set to "Published " in the library information shown in their individual specifications.

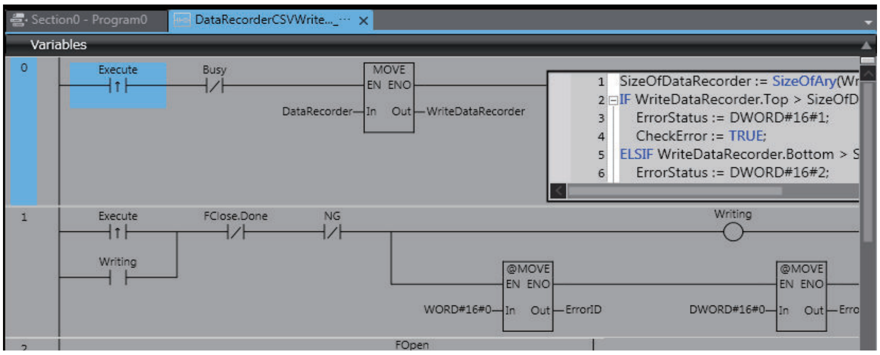
Use the following procedure to refer to the source codes of function blocks and functions.

1 Select a function block or function in the program.



2 Double-click or right-click and select **To Lower Layer** from the menu.

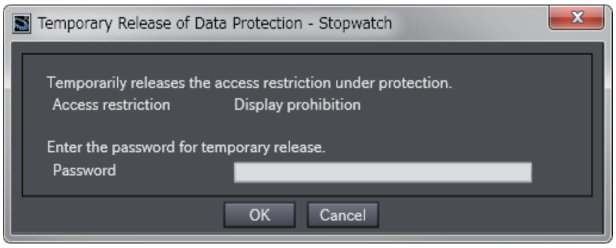
The source code is displayed.



```
SizeOfDataRecorder := SizeOfAry(Wr
IF WriteDataRecorder.Top > SizeOfD
    ErrorStatus := DWORD#16#1;
    CheckError := TRUE;
ELSIF WriteDataRecorder.Bottom > S
    ErrorStatus := DWORD#16#2;
```

Precautions for Correct Use

- For function blocks and functions whose source codes are not published, the following dialog box is displayed in the above step 2. Click the **Cancel** button.



Temporary Release of Data Protection - Stopwatch

Temporarily releases the access restriction under protection.
Access restriction Display prohibition

Enter the password for temporary release.
Password

Index

Index

OMRON Corporation **Industrial Automation Company**
Kyoto, JAPAN

Contact: www.ia.omron.com

Regional Headquarters

OMRON EUROPE B.V.
Wegalaan 67-69, 2132 JD Hoofddorp
The Netherlands
Tel: (31)2356-81-300/Fax: (31)2356-81-388

OMRON ASIA PACIFIC PTE. LTD.
No. 438A Alexandra Road # 05-05/08 (Lobby 2),
Alexandra Technopark,
Singapore 119967
Tel: (65) 6835-3011/Fax: (65) 6835-2711

OMRON ELECTRONICS LLC
2895 Greenspoint Parkway, Suite 200
Hoffman Estates, IL 60169 U.S.A.
Tel: (1) 847-843-7900/Fax: (1) 847-843-7787

OMRON (CHINA) CO., LTD.
Room 2211, Bank of China Tower,
200 Yin Cheng Zhong Road,
PuDong New Area, Shanghai, 200120, China
Tel: (86) 21-5037-2222/Fax: (86) 21-5037-2200

Authorized Distributor:

© OMRON Corporation 2018-2019 All Rights Reserved.
In the interest of product improvement,
specifications are subject to change without notice.

Cat. No. W597-E1-02 0119