OMRON

Machine Automation Controller

NJ/NX-series

CPU Unit Built-in EtherNet/IP™ Port

User's Manual

NX701-1□□□
NX102-1□□□
NX102-90□□
NX1P2-1□□□□□
NX1P2-9□□□□□
NJ501-□□□□
NJ301-1□□□
NJ101-10□□
NJ101-90□□

CPU Unit

SYSMAC
always in control

W506-E1-31

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, Excel, Visual Basic, and Microsoft Edge are either registered trademarks or trademarks of Microsoft Corporation in the United States and other countries.
- EtherCAT® is registered trademark and patented technology, 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 used with permission from Microsoft.
- This product incorporates certain third party software. The license and copyright information associated with this software is available at http://www.fa.omron.co.jp/nj_info_e/.

Introduction

Thank you for purchasing an NJ/NX-series CPU Unit.
This manual contains information that is necessary to use the NJ/NX-series CPU Unit. Please read this manual and make sure you understand the functionality and performance of the NJ/NX-series CPU Unit before you attempt to use it in a control system.
Keep this manual in a safe place where it will be available for reference during operation.

Intended Audience

This manual is intended for the following personnel, who must also have knowledge of electrical systems (electrical engineers 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

This manual covers the following products.

- NX-series CPU Units
  - NX701-17□□
  - NX701-16□□
  - NX102-12□□
  - NX102-11□□
  - NX102-10□□
  - NX102-90□□
  - NX1P2-11□□□□
  - NX1P2-11□□□□1
  - NX1P2-10□□□□
  - NX1P2-10□□□□1
  - NX1P2-90□□□□
  - NX1P2-90□□□□1
  - NX1P2-9B□□□□
  - NX1P2-9B□□□□1
- NJ-series CPU Units
  - NJ501-□5□□
  - NJ501-□4□□
  - NJ501-□3□□
  - NJ301-12□□
  - NJ301-11□□
  - NJ101-10□□
  - NJ101-90□□

Part of the specifications and restrictions for the CPU Units are given in other manuals.
Refer to *Relevant Manuals* on page 2 and *Related Manuals* on page 24.

Relevant Manuals

The following table provides the relevant manuals for the NJ/NX-series CPU Units. Read all of the manuals that are relevant to your system configuration and application before you use the NJ/NX-series CPU Unit.

Most operations are performed from the Sysmac Studio Automation Software. Refer to the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for information on the Sysmac Studio.

Purpose of use	Manual																		
	Basic information																		
	NX-series CPU Unit Hardware User's Manual	NX-series NX102 CPU Unit Hardware User's Manual	NX-series NX1P2 CPU Unit Hardware User's Manual	NJ-series CPU Unit Hardware User's Manual	NJ/NX-series CPU Unit Software User's Manual	NX-series NX1P2 CPU Unit Built-in I/O and Option Board User's Manual	NJ/NX-series Instructions Reference Manual	NJ/NX-series CPU Unit Motion Control User's Manual	NJ/NX-series Motion Control Instructions Reference Manual	NJ/NX-series CPU Unit Built-in EtherCAT Port User's Manual	NJ/NX-series CPU Unit Built-in EtherNet/IP Port User's Manual	NJ/NX-series CPU Unit User's Manual	NX-series CPU Unit FINS User's Manual	NJ/NX-series Database Connection CPU Units User's Manual	NJ-series SECS/GEM CPU Units User's Manual	NJ-series Robot Integrated CPU Unit User's Manual	NJ-series NJ Robotics CPU Unit User's Manual	NJ/NY-series NC Integrated Controller User's Manual	NJ/NX-series Troubleshooting Manual
Introduction to NX701 CPU Units	○																		
Introduction to NX102 CPU Units		○																	
Introduction to NX1P2 CPU Units			○																
Introduction to NJ-series Controllers				○															
Setting devices and hardware	○	○	○	○															
Using motion control								○											
Using EtherCAT										○									
Using EtherNet/IP											○								
Using robot control for OMRON robots																○			

Purpose of use	Manual																			
	Basic information																			
	NX-series CPU Unit Hardware User's Manual	NX-series NX102 CPU Unit Hardware User's Manual	NX-series NX1P2 CPU Unit Hardware User's Manual	NJ-series CPU Unit Hardware User's Manual	NJ/NX-series CPU Unit Software User's Manual	NX-series NX1P2 CPU Unit Built-in I/O and Option Board User's Manual	NJ/NX-series Instructions Reference Manual	NJ/NX-series CPU Unit Motion Control User's Manual	NJ/NX-series Motion Control Instructions Reference Manual	NJ/NX-series CPU Unit Built-in EtherCAT Port User's Manual	NJ/NX-series CPU Unit Built-in EtherNet/IP Port User's Manual	NJ/NX-series CPU Unit User's Manual	NX-series CPU Unit FINS User's Manual	NJ/NX-series Database Connection CPU Units User's Manual	NJ-series SECS/GEM CPU Units User's Manual	NJ-series Robot Integrated CPU Unit User's Manual	NJ-series NJ Robotics CPU Unit User's Manual	NJ/NY-series NC Integrated Controller User's Manual	NJ/NX-series Troubleshooting Manual
Software settings					○															
Using motion control								○												
Using EtherCAT										○										
Using EtherNet/IP											○									
Using OPC UA												○								
Using FINS													○							
Using the database connection service														○						
Using the GEM Services															○					
Using robot control for OMRON robots																○				
Using robot control by NJ Robotics function																	○			
Using numerical control																		○		
Using the NX1P2 CPU Unit functions						○														
Writing the user program					○		○													
Using motion control								○	○											
Using EtherCAT										○										
Using EtherNet/IP											○									
Using OPC UA												○								
Using FINS													○							
Using the database connection service														○						
Using the GEM Services					○		○								○					
Using robot control for OMRON robots																○				
Using robot control by NJ Robotics function																	○			
Using numerical control																		○		
Programming error processing																				○
Using the NX1P2 CPU Unit functions						○														

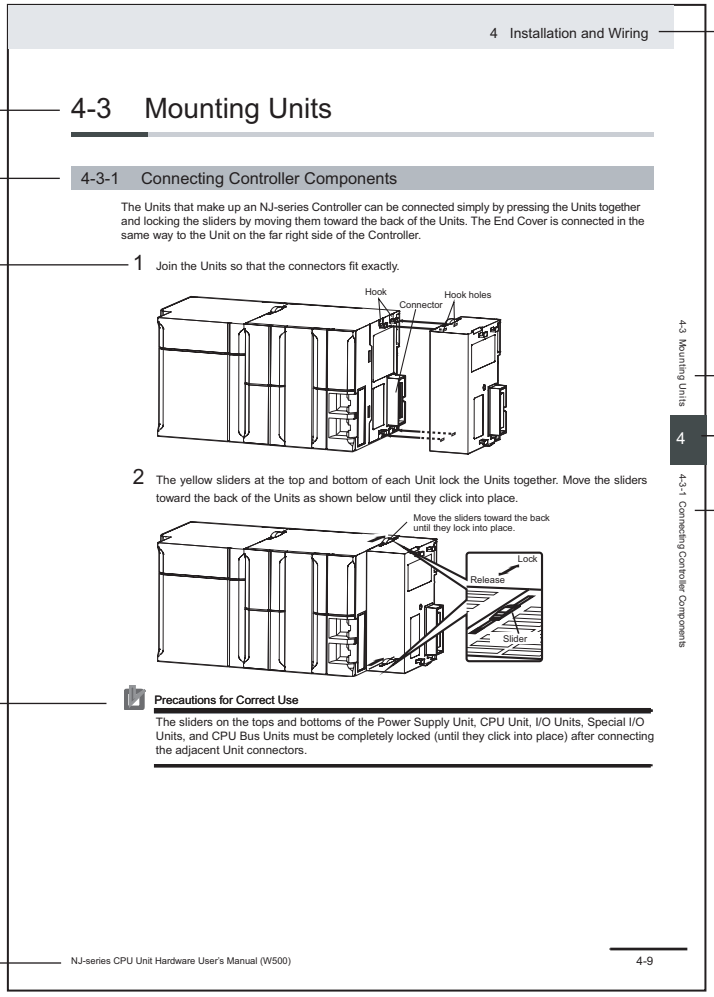
Purpose of use	Manual																			
	Basic information																			
	NX-series CPU Unit Hardware User's Manual	NX-series NX102 CPU Unit Hardware User's Manual	NX-series NX1P2 CPU Unit Hardware User's Manual	NJ-series CPU Unit Hardware User's Manual	NJ/NX-series CPU Unit Software User's Manual	NX-series NX1P2 CPU Unit Built-in I/O and Option Board User's Manual	NJ/NX-series Instructions Reference Manual	NJ/NX-series CPU Unit Motion Control User's Manual	NJ/NX-series Motion Control Instructions Reference Manual	NJ/NX-series CPU Unit Built-in EtherCAT Port User's Manual	NJ/NX-series CPU Unit Built-in EtherNet/IP Port User's Manual	NJ/NX-series CPU Unit OPC UA User's Manual	NX-series CPU Unit FINS User's Manual	NJ/NX-series Database Connection CPU Units User's Manual	NJ-series SECS/GEM CPU Units User's Manual	NJ-series Robot Integrated CPU Unit User's Manual	NJ-series NJ Robotics CPU Unit User's Manual	NJ/NY-series NC Integrated Controller User's Manual	NJ/NX-series Troubleshooting Manual
Testing operation and debugging																				
Using motion control								○												
Using EtherCAT										○										
Using EtherNet/IP											○									
Using OPC UA												○								
Using FINS													○							
Using the database connection service					○									○						
Using the GEM Services															○					
Using robot control for OMRON robots																○				
Using robot control by NJ Robotics function																	○			
Using numerical control																		○		
Using the NX1P2 CPU Unit functions						○														
Learning about error management and corrections*1												△	△	△	△	△	△	△	△	○
Maintenance																				
Using motion control	○	○	○	○				○												
Using EtherCAT										○										
Using EtherNet/IP											○									

*1. Refer to the *NJ/NX-series Troubleshooting Manual (Cat. No. W503)* for the error management concepts and the error items. However, refer to the manuals that are indicated with triangles for details on errors corresponding to the products with the manuals that are indicated with triangles.

Manual Structure

Page Structure

The following page structure is used in this manual.



Level 2 heading

Level 3 heading

A step in a procedure

Indicates a procedure.

Special information

Icons indicate precautions, additional information, or reference information.

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 or make operation easier.

 **Version Information**

Information on differences in specifications and functionality for Controller with different unit versions and for different versions of the Sysmac Studio is given.

Precaution on Terminology

In this manual, "download" refers to transferring data from the Sysmac Studio to the physical Controller and "upload" refers to transferring data from the physical Controller to the Sysmac Studio.
For the Sysmac Studio, "synchronization" is used to both "upload" and "download" data. Here, "synchronize" means to automatically compare the data for the Sysmac Studio on the computer with the data in the physical Controller and transfer the data in the direction that is specified by the user.

Sections in this Manual

CONTENTS

Section 1 Introduction

Section 6 Tag Data Link Functions

Section 7 CIP Message Communications

Section 8 Socket Service

Section 9 Modbus TCP Master Function

Section 10 FTP Server

Section 11 FTP Client

Section 12 Automatic Clock Adjustment

Section 13 SNMP Agent

Section 14 Communications Performance and Communications Load

Section 15 Troubleshooting

Appendices

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.

Statement of security responsibilities for assumed use cases and against threats

OMRON SHALL NOT BE RESPONSIBLE AND/OR LIABLE FOR ANY LOSS, DAMAGE, OR EXPENSES DIRECTLY OR INDIRECTLY RESULTING FROM THE INFECTION OF OMRON PRODUCTS, ANY SOFTWARE INSTALLED THEREON OR ANY COMPUTER EQUIPMENT, COMPUTER PROGRAMS, NETWORKS, DATABASES OR OTHER PROPRIETARY MATERIAL CONNECTED THERETO BY DISTRIBUTED DENIAL OF SERVICE ATTACK, COMPUTER VIRUSES, OTHER TECHNOLOGICALLY HARMFUL MATERIAL AND/OR UNAUTHORIZED ACCESS.

It shall be the users sole responsibility to determine and use adequate measures and checkpoints to satisfy the users particular requirements for (i) antivirus protection, (ii) data input and output, (iii) maintaining a means for reconstruction of lost data, (iv) preventing Omron Products and/or software installed thereon from being infected with computer viruses and (v) protecting Omron Products from unauthorized access.

Safety Precautions

Refer to the following manuals for safety precautions.

- *NX-series CPU Unit Hardware User's Manual (Cat. No. W535)*
- *NX-series NX102 CPU Unit Hardware User's Manual (Cat. No. W593)*
- *NX-series NX1P2 CPU Unit Hardware User's Manual (Cat. No. W578)*
- *NJ-series CPU Unit Hardware User’s Manual (Cat No. W500)*

Precautions for Safe Use

Refer to the following manuals for precautions for safe use.

- *NX-series CPU Unit Hardware User's Manual (Cat. No. W535)*
- *NX-series NX102 CPU Unit Hardware User's Manual (Cat. No. W593)*
- *NX-series NX1P2 CPU Unit Hardware User's Manual (Cat. No. W578)*
- *NJ-series CPU Unit Hardware User’s Manual (Cat No. W500)*

Precautions for Correct Use

Refer to the following manuals for precautions for correct use.

- *NX-series CPU Unit Hardware User's Manual (Cat. No. W535)*
- *NX-series NX102 CPU Unit Hardware User's Manual (Cat. No. W593)*
- *NX-series NX1P2 CPU Unit Hardware User's Manual (Cat. No. W578)*
- *NJ-series CPU Unit Hardware User’s Manual (Cat No. W500)*

Regulations and Standards

Refer to the following manuals for regulations and standards.

- *NX-series CPU Unit Hardware User's Manual (Cat. No. W535)*
- *NX-series NX102 CPU Unit Hardware User's Manual (Cat. No. W593)*
- *NX-series NX1P2 CPU Unit Hardware User's Manual (Cat. No. W578)*
- *NJ-series CPU Unit Hardware User's Manual (Cat No. W500)*

Software Licenses and Copyrights

The products supporting secure socket services incorporate the following third party software. The license and copyright information associated with this software is available at http://www.fa.omron.co.jp/nj_info_e/.

- OpenSSL
  This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/).
  Copyright (C) 1998-2019 The OpenSSL Project. All rights reserved.
  Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com) All rights reserved.
  This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)

Refer to *8-9 Secure Socket Services* on page 8-35 for models that support secure socket services.

Versions

Hardware revisions and unit versions are used to manage the hardware and software in NJ/NX-series Units and EtherCAT slaves. The hardware revision or unit version is updated each time there is a change in hardware or software specifications. Even when two Units or EtherCAT slaves have the same model number, they will have functional or performance differences if they have different hardware revisions or unit versions.

Refer to the following manuals for versions.

- *NX-series CPU Unit Hardware User's Manual (Cat. No. W535)*
- *NX-series NX102 CPU Unit Hardware User's Manual (Cat. No. W593)*
- *NX-series NX1P2 CPU Unit Hardware User's Manual (Cat. No. W578)*
- *NJ-series CPU Unit Hardware User's Manual (Cat No. W500)*

Unit Versions of CPU Units and Sysmac Studio Versions

The functions that are supported depend on the unit version of the NJ/NX-series CPU Unit. The version of Sysmac Studio that supports the functions that were added for an upgrade is required to use those functions.
Refer to the *NJ/NX-series CPU Unit Software User's Manual* (Cat. No. W501) for the relationship between the unit versions of CPU Units and the Sysmac Studio versions, and for the functions that are supported by each unit version.

Unit Versions of CPU Units and Peripheral Tool Versions

When you set tag data links for the built-in EtherNet/IP port on an NJ/NX-series CPU Unit, use the versions of the Network Configurator and the Sysmac Studio that are given in the following table.

OK: Supported, ---: Not supported

CPU Unit		Network Configurator for EtherNet/IP							Sysmac Studio				
Model	Version	Ver. 3.3x or lower	Ver. 3.40	Ver.3. 50 or 3.51	Ver. 3.53 to 3.58	Ver. 3.59 to 3.60	Ver. 3.61 to 3.63	Ver. 3.64 or higher	Ver. 1.09 or lower	Ver. 1.10 to 1.12	Ver. 1.13 to 1.16	Ver. 1.17 to 1.22	Ver. 1.23 or higher
NJ501	Ver. 1.00 to 1.02	---	OK	OK	OK	OK	OK	OK	---	OK	OK	OK	OK
NJ301	Ver. 1.01 to 1.02	---	---	OK	OK	OK	OK	OK	---	OK	OK	OK	OK
NJ501 NJ301	Ver. 1.03 or later	---	---	---	OK	OK	OK	OK	---	OK	OK	OK	OK
NJ101 NX701	Ver. 1.10 or later	---	---	---	---	OK	OK	OK	---	---	OK	OK	OK

CPU Unit		Network Configurator for EtherNet/IP							Sysmac Studio				
Model	**Version**	**Ver. 3.3x or lower**	**Ver. 3.40**	**Ver.3. 50 or 3.51**	**Ver. 3.53 to 3.58**	**Ver. 3.59 to 3.60**	**Ver. 3.61 to 3.63**	**Ver. 3.64 or higher**	**Ver. 1.09 or lower**	**Ver. 1.10 to 1.12**	**Ver. 1.13 to 1.16**	**Ver. 1.17 to 1.22**	**Ver. 1.23 or higher**
NX1P2	Ver. 1.13 or later	---	---	---	---	---	OK	OK	---	---	---	OK	OK*1
NX102	Ver. 1.30 or later	---	---	---	---	---	---	OK	---	---	---	---	OK

*1. Use an NX1P2-9B□□□□□□ CPU Unit with Sysmac Studio version 1.30 or higher.

Related Manuals

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

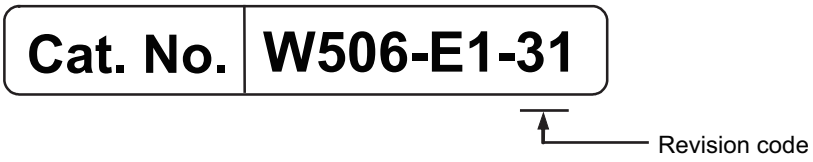
Manual name	Cat. No.	Model numbers	Application	Description
NX-series CPU Unit Hardware User's Manual	W535	NX701-□□□□	Learning the basic specifications of the NX701 CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction 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
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
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	Cat. No.	Model numbers	Application	Description
NX-series NX1P2 CPU Unit Built-in I/O and Option Board User's Manual	W579	NX1P2-□□□□	Learning about the details of functions only for an NX-series NX1P2 CPU Unit and an introduction of functions for an NJ/NX-series CPU Unit.	Of the functions for an NX1P2 CPU Unit, the following information is provided. • Built-in I/O • Serial Communications Option Boards • Analog I/O Option Boards An introduction of following functions for an NJ/NX-series CPU Unit is also provided. • Motion control functions • EtherNet/IP communications functions • EtherCAT communications functions
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.
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.
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.
NJ/NX-series CPU Unit Built-in EtherCAT® Port User's Manual	W505	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Using the built-in EtherCAT port on an NJ/NX-series CPU Unit.	Information on the built-in EtherCAT port is provided. This manual provides an introduction and provides information on the configuration, features, and setup.
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.
NJ/NX-series CPU Unit OPC UA User's Manual	W588	NX701-□□□□ NX102-□□□□ NJ501-1□00	Using the OPC UA.	Describes the OPC UA.
NX-series CPU Unit FINS Function User's Manual	W596	NX701-□□20 NX102-□□□□	Using the FINS function of an NX-series CPU Unit.	Describes the FINS function of an NX-series CPU Unit.
NJ/NX-series Database Connection CPU Units User's Manual	W527	NX701-□□20 NX102-□□20 NJ501-□□20 NJ101-□□20	Using the database connection service with NJ/NX-series Controllers.	Describes the database connection service.
NJ-series SECS/GEM CPU Units User's Manual	W528	NJ501-1340	Using the GEM Services with NJ-series Controllers.	Provides information on the GEM Services.
NJ-series Robot Integrated CPU Unit User's Manual	O037	NJ501-R□□□	Using the NJ-series Robot Integrated CPU Unit.	Describes the settings and operation of the CPU Unit and programming concepts for OMRON robot control.

Manual name	Cat. No.	Model numbers	Application	Description
Sysmac Studio Robot Integrated System Building Function with Robot Integrated CPU Unit Operation Manual	W595	SYSMAC-SE2□□□ SYSMAC-SE200D-64	Learning about the operating procedures and functions of the Sysmac Studio to configure Robot Integrated System using Robot Integrated CPU Unit.	Describes the operating procedures of the Sysmac Studio for Robot Integrated CPU Unit.
Sysmac Studio Robot Integrated System Building Function with IPC Application Controller Operation Manual	W621	SYSMAC-SE2□□□ SYSMAC-SE200D-64	Learning about the operating procedures and functions of the Sysmac Studio to configure Robot Integrated System using IPC Application Controller.	Describes the operating procedures of the Sysmac Studio for IPC Application Controller.
Sysmac Studio 3D Simulation Function Operation Manual	W618	SYSMAC-SE2□□□ SYSMAC-SA4□□□-64	Learning about an outline of the 3D simulation function of the Sysmac Studio and how to use the function.	Describes an outline, execution procedures, and operating procedures for the 3D simulation function of the Sysmac Studio.
NJ-series NJ Robotics CPU Unit User's Manual	W539	NJ501-4□□□ NJ501-R□□□	Controlling robots with NJ-series CPU Units.	Describes the functionality to control robots.
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.
NJ/NX-series Troubleshooting Manual	W503	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning about the errors that may be detected in an NJ/NX-series Controller.	Concepts on managing errors that may be detected in an NJ/NX-series Controller and information on individual errors 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.
NX-series Safety Control Unit User's Manual	Z930	NX-SL□□□□ NX-SI□□□□ NX-SO□□□□	Learning how to use NX-series Safety Control Units.	Describes the hardware, setup methods, and functions of the NX-series Safety Control Units.
Sysmac Library User's Manual for MQTT Communications Library	W625	SYSMAC-XR020	Learning how to perform Pub/Sub message communications through MQTT broker.	Describes the specifications and procedures to use the function block of MQTT communications library.

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	July 2011	Original production
02	March 2012	• Added information on the NJ301-□□□□. • Added *A-8 Accesing Variables with CIP Message Communications*. • Added information on the functions supported by unit version 1.01 of the CPU Units. • Corrected mistakes.
03	May 2012	• Added information on the functions supported by unit version 1.02 of the CPU Units. • Corrected mistakes.
04	August 2012	• Added information on the functions supported by unit version 1.03 of the CPU Units. • Corrected mistakes.
05	February 2013	• Added information on the functions supported by unit version 1.04 of the CPU Units. • Corrected mistakes.
06	April 2013	• Corrected mistakes.
07	June 2013	• Added information on the functions supported by unit version 1.06 of the CPU Units.
08	December 2013	• Added information on the functions supported by unit version 1.08 of the CPU Units. • Corrected mistakes.
09	July 2014	• Added information on the functions supported by unit version 1.09 of the CPU Units. • Corrected mistakes.
10	January 2015	• Added information on the functions supported by unit version 1.10 of the CPU Units. • Corrected mistakes.
11	April 2015	• Added information on the NX701-□□□□. • Added information on the NJ101-□□□□. • Corrected mistakes.
12	October 2015	• Added information on the hardware revision. • Corrected mistakes.
13	April 2016	• Added information on the functions supported by unit version 1.11 of the CPU Units. • Corrected mistakes.

Revision code	Date	Revised content
14	July 2016	• Added information on the functions supported by unit version 1.12 of the CPU Units. • Corrected mistakes.
15	October 2016	• Added information on the NX1P2-□□□□□□. • Added information on the functions supported by unit version 1.13 of the CPU Units. • Corrected mistakes.
16	April 2017	• Added information on the functions supported by unit version 1.14 of the CPU Units. • Corrected mistakes.
17	October 2017	• Corrected mistakes.
18	January 2018	• Added information on the functions supported by unit version 1.17 of the CPU Units. • Corrected mistakes.
19	April 2018	• Added information on the NX102-□□□□. • Added information on the functions supported by unit version 1.30 of the CPU Units. • Consolidated descriptions related to event codes and errors into the *NJ/NX-series Troubleshooting Manual*. • Corrected mistakes.
20	July 2018	• Added information on the functions supported by unit version 1.31 of the NX102-□□□□.
21	April 2019	• Added information on the functions supported by unit version 1.32 of NX102-□□□□. • Added information on the functions supported by unit version 1.21 of the NX1P2-□□□□□□, NJ501-1□00, NJ301-□□□□, and NJ101-□□00. • Corrected mistakes.
22	July 2019	• Added information on the functions supported by unit version 1.21 of the NX701-□□□□, NJ501-4□00, NJ501-4□10, NJ501-1340 and NJ501-5300. • Corrected mistakes.
23	October 2019	• Added information on the NX1P2-9B□□□□. • Corrected mistakes.
24	August 2020	• Made changes accompanying the addition of NJ501-R□□□. • Corrected mistakes.
25	July 2021	• Added information on the functions supported by unit version 1.24 of the NX701-1□□0. • Added information on the functions supported by unit version 1.36 of the NX102-1□20. • Added information on the functions supported by unit version 1.45 of the NX1P2-□□00, NJ301-□□00, and NJ101-□□00. • Added information on the functions supported by unit version 1.25 of the NJ501-1□20, NJ501-1340, NJ501-4□□□, NJ501-5300, and NJ101-1□20. • Added information on the functions supported by unit version 1.43 of the NX102-□□00, NJ501-1□00, and NJ501-R□00. • Made changes on the information of the SD Memory Card. • Corrected mistakes.
26	October 2021	• Added information related to the hardware revision A of the NX701-□□□□. • Corrected mistakes.

Revision code	Date	Revised content
27	November 2021	• Added information related to the hardware revision D of the NJ-series CPU Unit.
28	April 2022	• Added information to Terms and Conditions Agreement.
29	June 2022	• Added information related to the hardware revision B of the NX701-□□□□.
30	November 2022	• Added information on the functions supported by unit version 1.60 of the NJ-series, NX102, and NX1P2 CPU Units. • Added information on the functions supported by unit version 1.32 of the NX701 CPU Units.
31	January 2023	• Corrected mistakes.

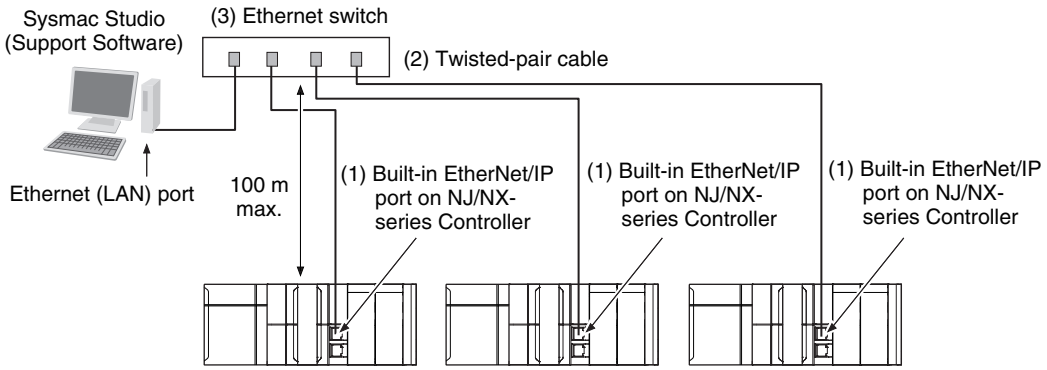
1

Introduction

1-1 Introduction

1-1-1 EtherNet/IP Features

EtherNet/IP is an industrial multi-vendor network that uses Ethernet.
The EtherNet/IP specifications are open standards managed by the ODVA (Open DeviceNet Vendor Association), just like DeviceNet.
EtherNet/IP is not just a network between Controllers. It is also used as a field network. Because EtherNet/IP uses standard Ethernet technology, various general-purpose Ethernet devices can be used in the network.



EtherNet/IP System Configuration Example

● High-speed, High-capacity Data Exchange through Tag Data Links

The EtherNet/IP protocol supports implicit communications, which allows cyclic communications (called tag data links in this manual) with EtherNet/IP devices.

● Tag Data Link (Cyclic Communications) Cycle Time

Tag data links (cyclic communications) operate at the cyclic period specified for each application, regardless of the number of nodes. Data is exchanged over the network at the refresh cycle set for each connection, so the communications refresh cycle will not increase even if the number of nodes is increased, i.e., the concurrency of the connection's data is maintained.
Because the refresh cycle can be set for each connection, each application can communicate at its ideal refresh cycle. For example, interprocess interlocks can be transferred at high speed, while the production commands and the status monitor information are transferred at low speed.

1-1-2 Features of Built-in EtherNet/IP Port on NJ/NX-series CPU Units

● Tag Data Links

Cyclic communications between Controllers or between a Controller and other devices are possible on an EtherNet/IP network.
High-speed data exchange can be performed through tag data links.

● CIP Message Communications

You can send CIP commands to devices on the EtherNet/IP network when required by executing CIP communications instructions in a program.
As a result, it is possible to send and receive data with the devices on the EtherNet/IP network.

● BOOTP Client

If the built-in EtherNet/IP port on an NJ/NX-series CPU Unit is set in the BOOTP settings, the BOOTP client operates when the Controller power is turned ON, and the IP address is obtained from the BOOTP server.
It is possible to set all of the IP addresses of multiple built-in EtherNet/IP ports at the same time.

● FTP Server for File Transfers to and from Host Computers

An FTP server is built into the Controller. You can use it to read and write data within the Controller as files from workstations and computers with FTP clients.
The FTP server enables the transfer of large amounts of data from a client without any additional ladder programming.

● FTP Client for File Transfers to and from Host Computers

An FTP client is built into the Controller, so you can read and write files on workstations and computers that have an FTP server from the Controller.
You can use the FTP client communications instructions to transfer one or more files between the Controller and an FTP server.

● NTP Client for Automatic Controller Clock Adjustment

The clocks built into Controllers connected to Ethernet can be automatically adjusted to the time of the clock in the NTP server. If all of the clocks in the system are automatically adjusted to the same time, time stamps can be used to analyze production histories.
*1. A separate NTP server is necessary to automatically adjust the Controller clocks.

● Socket Services

Socket services can be used to send and receive data between general-purpose applications and Controllers.
Through the communications services with sockets, you can send and receive data to and from remote nodes, i.e., between the host computer and Controllers or between Controllers.
You can execute socket communications instructions in order in a program to execute communications processes with the socket services.
There are two socket services, the UDP socket service and TCP socket service.
In addition, secure socket services which perform encrypted communications using TLS are available.
Secure socket service instructions can be used for secure socket communications with external cloud or on-premises servers.
In addition, the MQTT communications library can be used for secure socket communications with a MQTT broker .

Additional Information

Function Blocks (FBs) for MQTT communications are available for the secure socket communications between a CPU Unit and a MQTT broker.
Refer to the *Sysmac Library User's Manual for MQTT Communications Library (Cat. No. W625)* for more information on FBs for MQTT communications.

● DNS Client for Specifying Host Names

When you specify an NTP server, SNMP manager, or the destination of socket instructions or CIP communications instructions, you can use the host name, as well as its IP address (DNS client or hosts settings).

This will help identify the IP address automatically even after the IP addresses of relevant servers are changed due to system revisions.

*1. A separate DNS server is necessary when you use host names with the DNS client.
*2. The DNS server is specified directly using its IP address.

● Network Management with an SNMP Manager

The SNMP agent passes internal status information from the built-in EtherNet/IP port to network management software that uses an SNMP manager.

*1. A separate SNMP manager is necessary for network management.

● Complete Troubleshooting Functions

A variety of functions are provided to quickly identify and handle errors.

- Self-diagnosis at startup
- Event log that records the time of occurrence and other error details

● Two EtherNet/IP Communications Ports as a Standard Feature, Equipped with IP Routing Function (Only with the NX701 and NX102 CPU Units)

These CPU Units are equipped with two EtherNet/IP ports for EtherNet/IP communications as standard.

This feature allows you to separate the information network from the control network. In addition, the built-in EtherNet/IP ports support the IP routing function to send IP packets to devices on other IP network segments.

*1. In order to use the function, you must appropriately set the IP router table and default gateway settings for each device on the network according to your network configuration. For details on the settings, refer to *4-1 **TCP/IP Settings** Display* on page 4-2.

● CIP Safety on EtherNet/IP Compatible (Only with the NX102 CPU Units)

Combined with the NX-SL5□□□ Safety Control Unit, you can build a system which uses CIP Safety on EtherNet/IP communications in networks between Controllers and field networks. Safety communications by CIP Safety is enabled with devices that support CIP Safety on EtherNet/IP and other Safety CPU Units.

Additional Information

CIP (Common Industrial Protocol)

CIP is a shared industrial protocol for the OSI application layer. The CIP is used in networks such as EtherNet/IP, CompoNet, and DeviceNet.

Data can be routed easily between networks that are based on the CIP. You can therefore easily configure a transparent network from the field device level to the host level.

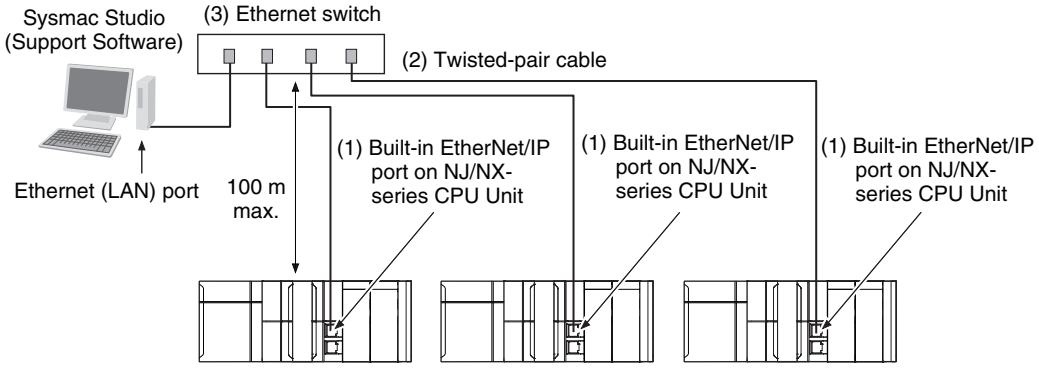
The CIP has the following advantages.

- Destination nodes are specified by a relative path, without fixed routing tables.
- The CIP uses the producer/consumer model. Nodes in the network are arranged on the same level and it is possible to communicate with required devices whenever it is necessary.
  The consumer node will receive data sent from a producer node when the connection ID in the packet indicates that the node requires the data. Because the producer can send the same data with the same characteristics in a multicast format, the time required for the transfer is fixed and not dependent on the number of consumer nodes. (Either multicast or unicast can be selected.)

1-2 System Configuration and Configuration Devices

1-2-1 Devices Required to Construct a Network

The basic configuration for an EtherNet/IP system includes one Ethernet switch to which nodes are attached in star configuration using twisted-pair cable.



The following products are also required to build a network. Obtain them in advance.

Network device	Function
Per Node • NJ-series CPU Unit (built-in EtherNet/IP port) (NJ501-□□□□/NJ301-□□□□/NJ101-□□□□) • NX-series CPU Unit (built-in EtherNet/IP port) (NX701-□□□□/NX102-□□□□□□/NX1P2-□□□□□□) • Other OMRON PLCs CJ2 CPU Units (built-in EtherNet/IP port) (CJ2H-CPU□□-EIP/CJ2M-CPU3□) CJ-series EtherNet/IP Unit*1 (CJ1W-EIP21) CS-series EtherNet/IP Unit (CS1W-EIP21)	These Units are used to connect to an EtherNet/IP network.
(2)Twisted-pair cable	The twisted-pair cable has an RJ45 Modular Connector at each end. This cable is used to connect the built-in EtherNet/IP port or EtherNet/IP Unit to an Ethernet switch. Use an STP (shielded twisted-pair) cable of category 5, 5e, or higher.
(3)Ethernet switch	This is a relay device that connects multiple nodes in a star LAN. For details on recommended devices to configure a network, refer to *2-1-1 Recommended Network Devices* on page 2-2.

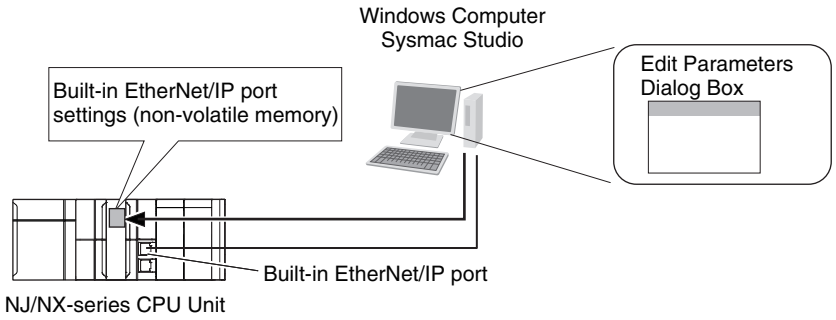
*1. The CJ1W-EIP21 EtherNet/IP Unit can be mounted only to an NJ-series CPU Unit. The unit version of the NJ-series CPU Unit should be 1.01 or later, and the Sysmac Studio version should be 1.02 or higher.

1-2-2 Support Software Required to Construct a Network

This section describes the Support Software that is required to construct an EtherNet/IP network.
The built-in EtherNet/IP port has Ethernet Settings and Tag Data Link Settings, which are both stored in the non-volatile memory of the CPU Unit.
Support Software is provided for each, as described below.

● Built-in EtherNet/IP Port Settings: Sysmac Studio

Use the Sysmac Studio to set the basic settings, such as the local IP address and subnet mask of the built-in EtherNet/IP port.
The Sysmac Studio can also be used to check if data I/O is being performed correctly for tag data links.

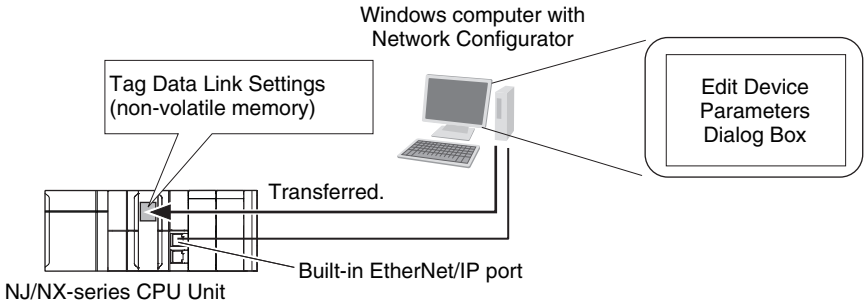


Refer to the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for details on the Sysmac Studio.

● Tag Data Link Settings: Network Configurator

Use the Network Configurator to set the tag data links for the built-in EtherNet/IP port. (The Network Configurator is included in the Sysmac Studio Standard Edition.) The main functions of the Network Configurator are given below.

- Setting and Monitoring Tag Data Links (Connections)
  The network device configuration and tag data links (connections) can be created and edited. After connecting to the network, the device configuration and tag data link settings can be uploaded and monitored.
- Multi-vendor Device Connections
  EDS files can be installed and deleted so that you can construct, set, and manage networks that contain EtherNet/IP devices from other companies. The IP addresses of EtherNet/IP devices can also be changed.



For details on the Network Configurator, refer to *Section 6 Tag Data Link Functions* on page 6-1.

Additional Information

You can also use the Sysmac Studio to set the tag data links.
Refer to *A-2 Use the Sysmac Studio to Set the Tag Data Links (EtherNet/IP Connections)* on page A-4 for details on setting the tag data links on the Sysmac Studio.

1-3 Built-in EtherNet/IP Port

1-3-1 Specifications

Item		Specifications					
		NX701-□□□□	NX102-□□□□	NX1P2-□□□□□	NJ501-□□□□ NJ301-□□□□		NJ101-□□□□
		Unit version 1.10 or later	Unit version 1.30 or later	Unit version 1.13 or later	Unit version 1.00 to 1.02	Unit version 1.03 or later	Unit version 1.10 or later
Communications protocol		TCP/IP or UDP/IP					
Supported services		Sysmac Studio connection, tag data link, CIP message communications, socket services, FTP server, FTP client, automatic clock adjustment (NTP client), SNMP agent, DNS client, BOOTP client, and Packet Filter*1					
		----	Packet Filter	----			
Number of ports		2 (IP routing function supported)		1			
Physical layer		100Base-TX, 10Base-T, or 1000Base-T (1000Base-T or 100Base-TX is recommended.) *2	100Base-TX or 10Base-T (100Base-TX is recommended.) *2				
Transmission specifications	Media access method	CSMA/CD					
	Modulation	Baseband					
	Transmission paths	Star form					
	Baud rate	1,000 Mbps (1000Base-T)	100 Mbps (100Base-TX)				
	Transmission media	Shielded twisted-pair (STP) cable, Category 5, 5e, or higher					
	Transmission distance	100 m max. (distance between hub and node)					
	Number of cascade connections	There is no limitation when an Ethernet switch is used.					
CIP service: Tag data links (cyclic communications)	Number of connections	256 per port (total of 512 with two ports)	32 per port (total of 64 with two ports)	32			
	Packet interval (refresh cycle)	0.5 to 10,000 ms in 0.5-ms increments	1 to 10,000 ms in 1-ms increments	2 to 10,000 ms in 1-ms increments	10 to 10,000 ms in 1-ms increments	1 to 10,000 ms in 1-ms increments	
		Packet intervals can be set independently for each connection. (Data is refreshed over the network at preset intervals and the refresh cycle does not depend on the number of nodes.)					

Item		Specifications					
		NX701-□□□□	NX102-□□□□	NX1P2-□□□□□	NJ501-□□□□ NJ301-□□□□		NJ101-□□□□
		Unit version 1.10 or later	Unit version 1.30 or later	Unit version 1.13 or later	Unit version 1.00 to 1.02	Unit version 1.03 or later	Unit version 1.10 or later
CIP service: Tag data links (cyclic communications)	Allowed communications bandwidth per Unit	40,000 pps[*3*4]	12,000 pps[*3*4]	3,000 pps[*3]	1,000 pps[*3]	3,000 pps[*3]	
		Note: The heartbeat is included.	Note: The heartbeat and the CIP Safety routing are included. [*5]	Note: The heartbeat is included.			
	Number of registrable tags	256 per port (total of 512 with two ports)		256			
	Tag types	Network variable CIO, Work, Holding, DM, or EM Areas cannot be used.	Network variable CIO, Work, Holding, DM, or EM Areas can be used.	Network variable CIO, Work, Holding, or DM Areas can be used.	Network variable CIO, Work, Holding, DM, or EM Areas can be used.		
	Number of tags per connection (= 1 tag set)	8 (7 tags when the tag set includes the Controller status)					
	Maximum link data size per node	739,328 bytes	38,400 bytes	19,200 bytes			
	Maximum data size per connection	1,444 bytes[*6]	600 bytes[*6]				
		Data concurrency is maintained within each connection. Refer to *6-1-7 Concurrency of Tag Data Link Data* on page 6-12 for methods to maintain concurrency.					
	Number of registrable tag sets	256 per port (1 connection = 1 tag set) (total of 512 with two ports)	32 per port (1 connection = 1 tag set) (total of 40 with two ports)[*7]	32 (1 connection = 1 tag set)			
	Maximum size of 1 tag set	722 words (The Controller status uses 1 word when the tag set includes the Controller status.)	300 words (The Controller status uses 1 word when the tag set includes the Controller status.)				
	Changing tag data link parameters when Controller is in RUN mode	Supported[*8]					
	Multi-cast packet filter[*9]	Supported					

Item		Specifications					
		NX701-□□□□	NX102-□□□□	NX1P2-□□□□□	NJ501-□□□□ NJ301-□□□□		NJ101-□□□□
		Unit version 1.10 or later	Unit version 1.30 or later	Unit version 1.13 or later	Unit version 1.00 to 1.02	Unit version 1.03 or later	Unit version 1.10 or later
CIP message service: Explicit messages*10	Class 3 (number of connections)	Connections: 128 per port (total of 256 with two ports) (clients plus server)	Connections: 32 per port (total of 64 with two ports) (clients plus server)	Connections: 32 (clients plus servers)			
	UCMM (unconnected)	Number of clients that can communicate at one time: 32 max. Number of servers that can communicate at one time: 32 max.					
	CIP routing*11	Supported CIP routing is supported for the following remote Units: NX701-□□□□, NX102-□□□□, NX1P2-□□□□, NJ501-□□□□, NJ301-□□□□, NJ101-□□□□, CS1W-EIP21, CJ1W-EIP21, CJ2H-CPU□□-EIP, and CJ2M-CPU3□. Using a combination of any Units above, communication can be extended up to a maximum of 8 levels.					
SNMP	Agents	SNMPv1 or SNMPv2c					
	MIB	MIB-II					
EtherNet/IP conformance test		Conforms to CT18	Conforms to CT14	Conforms to CT13	Conforms to CT18		
Ethernet interface		10Base-T, 100Base-TX, or 1000Base-T Auto negotiation or fixed settings	10Base-T or 100Base-TX Auto negotiation or fixed settings				

*1. The Packet Filter can be used in CPU Units with the following unit versions.
- NJ-series, NX102, NX1P2 CPU Unit: Version 1.49 or later
- NX701 CPU Unit: Version 1.29 or later

*2. If tag data links are being used, use 100Base-TX or 1000Base-T.

*3. Here, pps means “packets per second” and indicates the number of packets that can be processed in one second.

*4. If the two built-in EtherNet/IP ports are used simultaneously, the maximum communications data size means the maximum data size of the total of the two ports.

*5. An NX102 CPU Unit with unit version 1.31 or later is required to use the CIP Safety routing.

*6. To use a data size of 505 bytes or larger, the system must support a large forward open (an optional CIP specification). The CS, CJ, NJ, and NX-series Units support a large forward open, but before connecting to nodes of other companies, confirm that the devices also support it.

*7. When tag sets that exceed total of 40 are set, a Number of Tag Sets for Tag Data Links Exceeded (840E0000 hex) event occurs.

*8. If the parameters of the built-in EtherNet/IP port are changed, the port is restarted. When other nodes are in communications with the affected node, the communications will temporarily time out and automatically recover after the restart.

*9. Because the built-in EtherNet/IP port is equipped with an IGMP client (version 2), unnecessary multicast packets can be filtered out by an Ethernet switch that supports IGMP snooping.

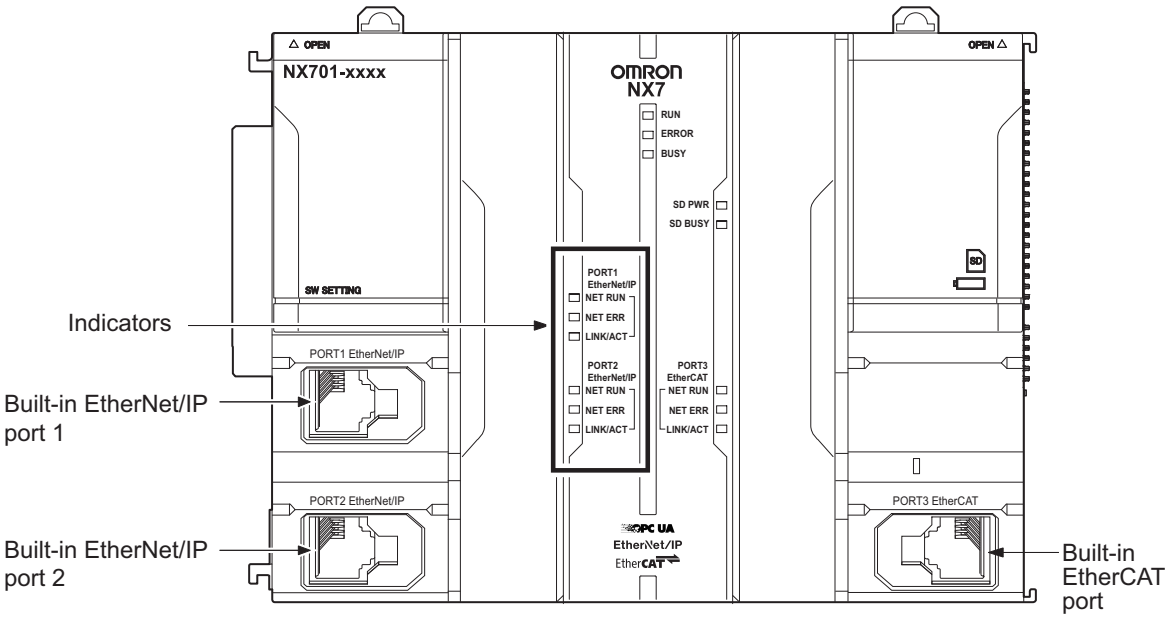
*10. The built-in EtherNet/IP port uses the TCP/UDP port numbers shown in *5-2 TCP/ UDP Port Numbers Used for the Built-in EtherNet/IP Port* on page 5-15.
Do not set the same port number for more than one TCP/UDP service.

*11. A CPU Unit with unit version 1.01 or later and Sysmac Studio version 1.02 or higher are required to use CIP routing.

1-3-2 Part Names and Functions

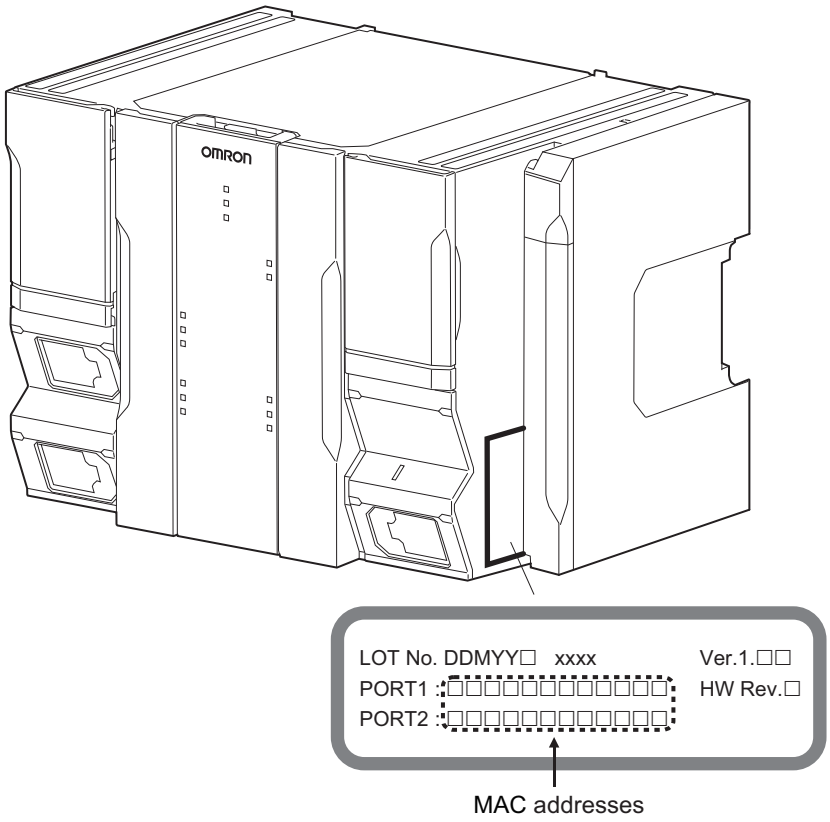
Parts and Names

- **NX701 CPU Unit**

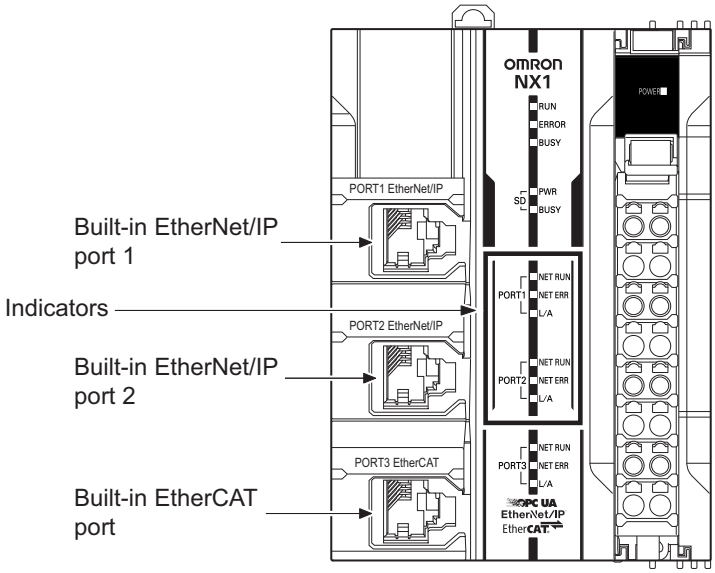


MAC Address Notation

A MAC address is uniquely allocated to each device connected to the Ethernet network. The MAC address of each built-in EtherNet/IP port is represented in 12-digit hexadecimal format and listed in the place of the Unit as shown below.

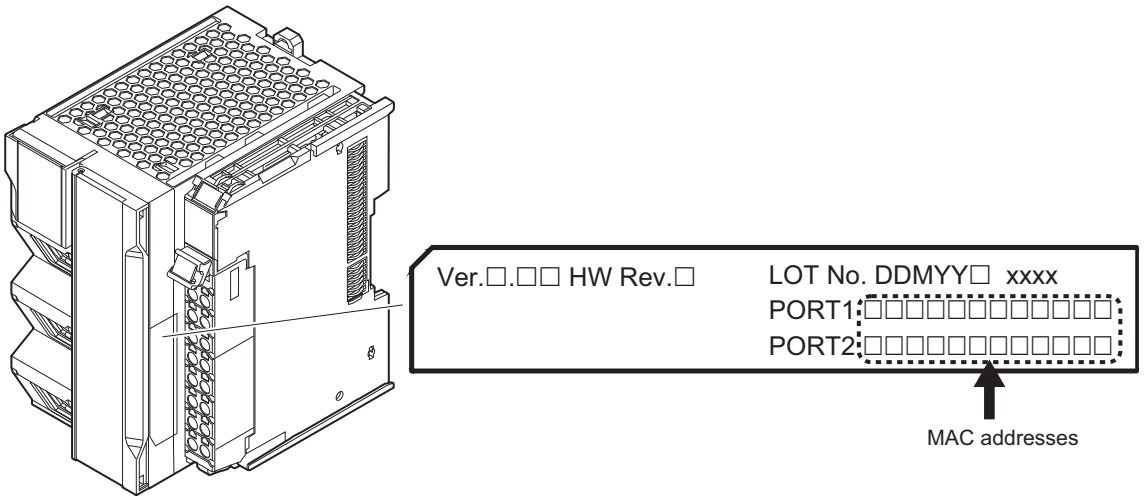


● NX102 CPU Unit

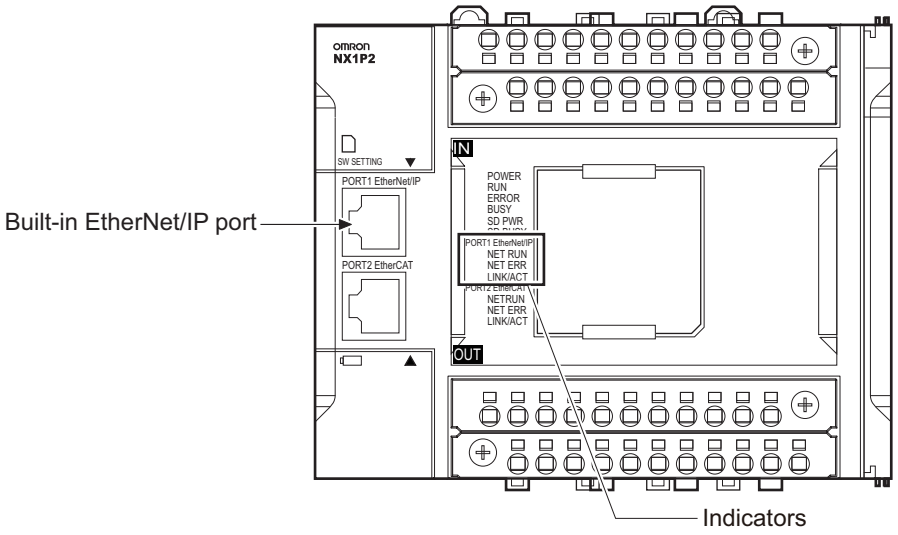


MAC Address Notation

A MAC address is uniquely allocated to each device connected to the Ethernet network. The MAC address of each built-in EtherNet/IP port is represented in 12-digit hexadecimal format and listed in the place of the Unit as shown below.

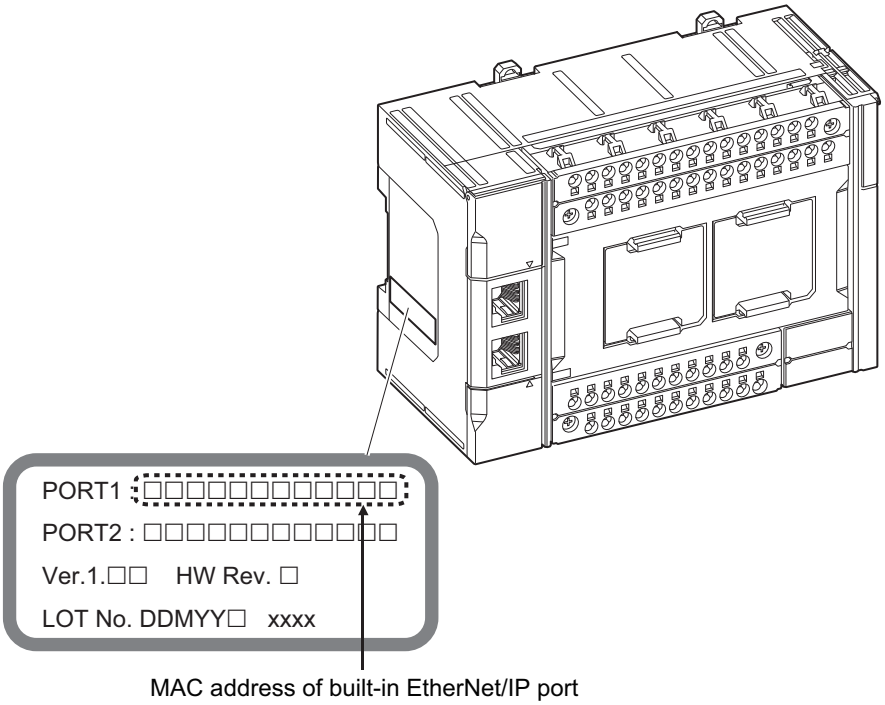

Ver.□.□□ HW Rev.□
LOT No. DDMYY□ xxxx
PORT1:□□□□□□□□□□□□
PORT2:□□□□□□□□□□□□
MAC addresses

● NX1P2 CPU Unit

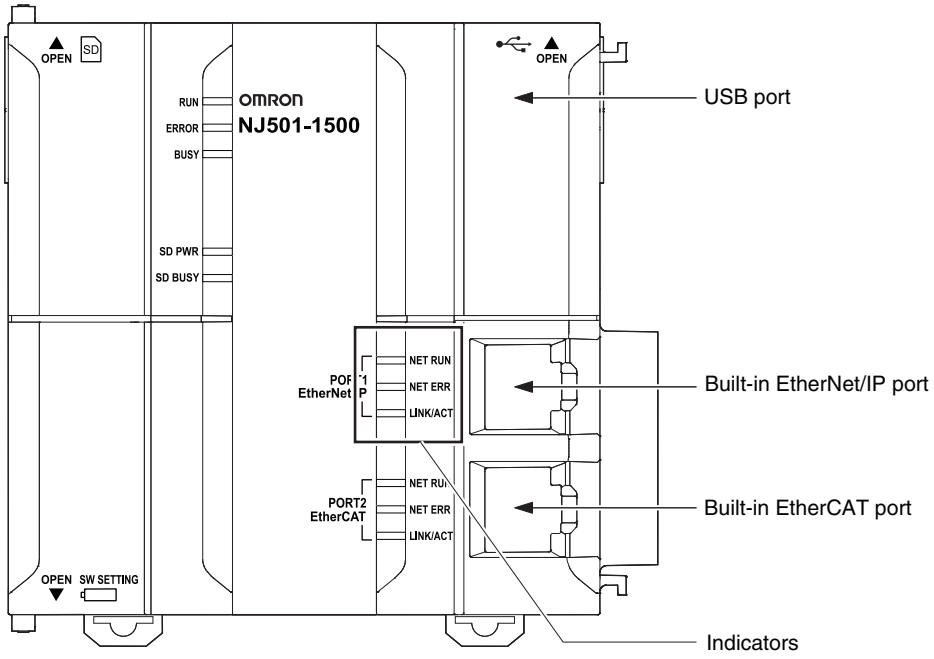


MAC Address Notation

A MAC address is uniquely allocated to each device connected to the Ethernet network. The MAC address of the built-in EtherNet/IP port is represented in 12-digit hexadecimal format and listed in the place of the Unit as shown below.

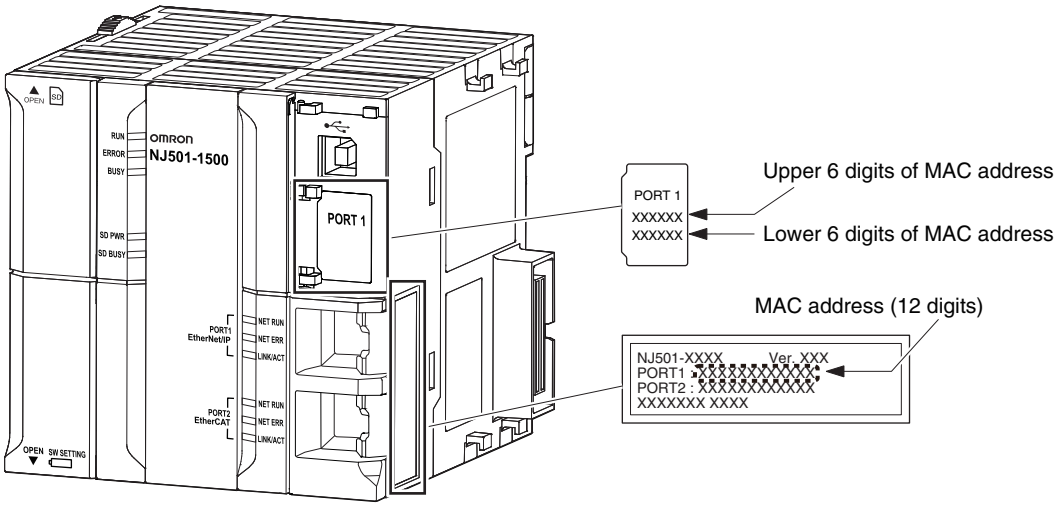


● NJ-series CPU Unit

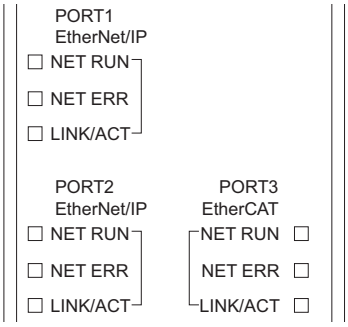


MAC Address Notation

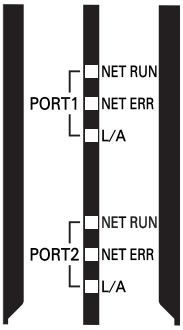
A MAC address is uniquely allocated to each device connected to the Ethernet network. The MAC address of the built-in EtherNet/IP port is represented in 12-digit hexadecimal format and listed in the two places of the Unit as shown below.



Indicators (LEDs)

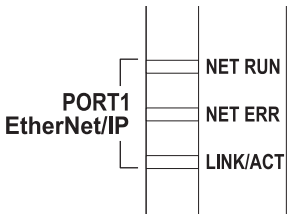


NX701 CPU Unit



NX102 CPU Unit

NX1P2 CPU Unit



NJ-series CPU Unit

● NET RUN, NET ERR, and LINK/ACT

- NET RUN indicator
  This shows the status of the CIP connection (tag data links, Class 3 messages).
- NET ERR indicator
  This shows the network communications error status. Refer to *Section 15 Troubleshooting* on page 15-1 for details.
- LINK/ACT indicator
  This shows the Ethernet communications status.

Indicator	Color	Status	Operating status
NET RUN	---	Not lit	Ethernet communications are not possible. • The power supply is OFF or the Controller is reset. • A MAC address error or communications Controller error is occurring. • The same IP address is assigned to more than one node.
	Green	Flashing	Ethernet communications are in progress. • Tag data link connection establishment in progress (originator operation) • IP address acquisition with BOOTP in progress.
		Lit	Normal If only the target is set for the tag data link, this indicator is lit regardless of whether the connection from the originator is established. It remains lit even if the data links are stopped.

Indicator	Color	Status	Operating status
NET ERR	---	Not lit	There are no Ethernet communications errors. • The power supply is OFF or the Controller is reset.
	Red	Flashing	A user-recoverable error is occurring. • An error is occurring in TCP/IP communications or CIP communications. • FTP Server Setting Error, NTP Server Setting Error, etc. • Tag Data Link Setting Error, Tag Data Link Verification Error, etc. • The same IP address is assigned to more than one node.
		Lit	A user-non-recoverable error is occurring. • A MAC address error or communications Controller error is occurring.
LINK/ACT	---	Not lit	The link is not established. • The cable is not connected. • The power supply is OFF or the Controller is reset.
	Yellow	Flashing	Data communications in progress after establishing the link.
		Lit	Link established.

Additional Information

When the built-in EtherNet/IP port is set to be disabled, all the indicators are turned OFF. Refer to *4-1* ***TCP/IP Settings*** *Display* on page 4-2 for details on the settings of a built-in EtherNet/IP port.

1-4 Introduction to Communications Services

1-4-1 CIP (Common Industrial Protocol) Communications Services

Tag Data Links (Cyclic Communications)

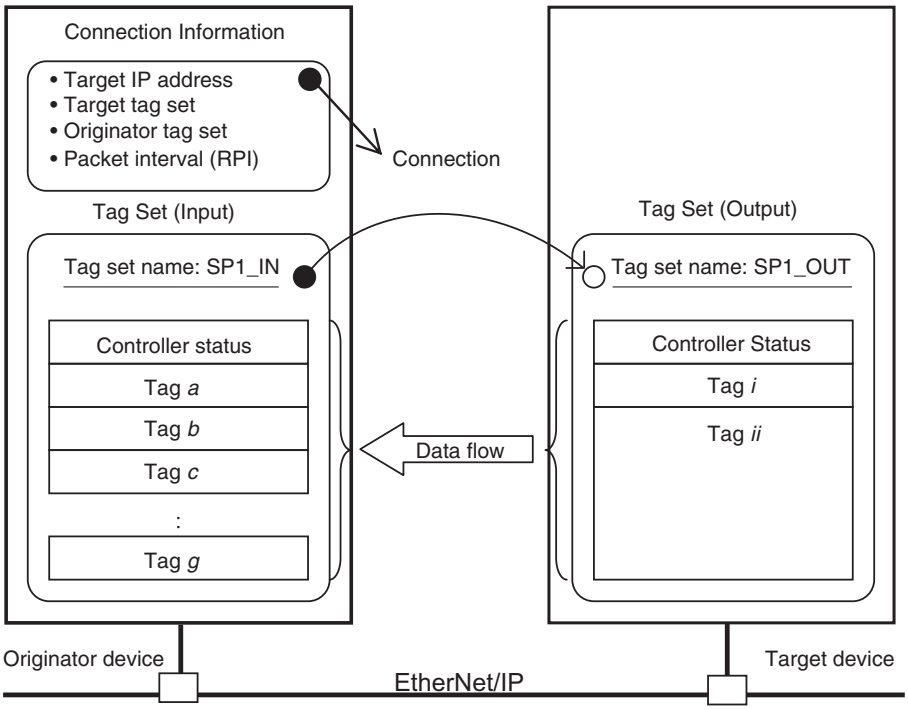
A program is not required to perform cyclic data exchanges with other devices on the EtherNet/IP network.

Normally, a connection is started with the target device for each tag set that was created with the Network Configurator to start communications for tag data links for a built-in EtherNet/IP port. One connection is used per tag set.

The maximum number of connections that can be registered is shown below.

- NX701 CPU Unit: 256 connections (total of 512 connections with two ports)
- NX102 CPU Unit: 32 connections (total of 64 connections with two ports)
- NX1P2 CPU Unit: 32 connections
- NJ-series CPU Unit: 32 connections

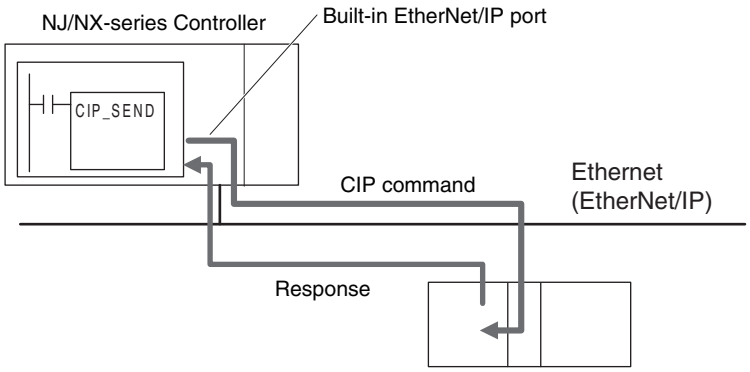
Refer to *1-3-1 Specifications* on page 1-9 for the built-in EtherNet/IP port tag and tag set specifications.



Note In this example, a connection is established with the originator's tag list with tags a to g (inputs), which are in a tag set called *SP1_IN*, and the target's tag list with tags i and ii (outputs), which are in a tag set called *SP1_OUT*.

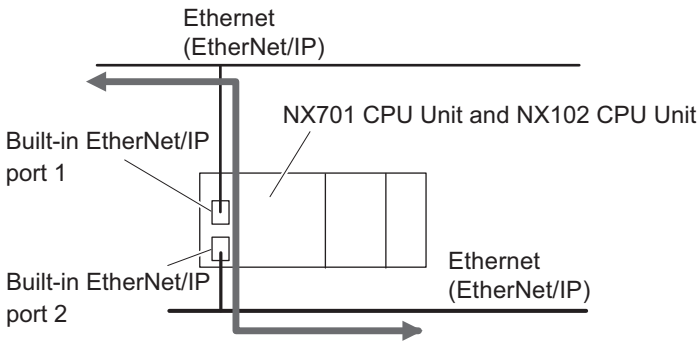
CIP Message Communications

User-specified CIP commands can be sent to devices on the EtherNet/IP network.
CIP commands, such as those for reading and writing data, can be sent and their responses received by executing the CIP communications instructions from the user program in the NJ/NX-series CPU Unit.

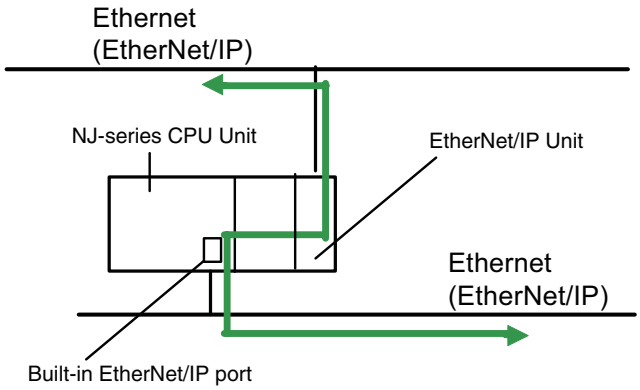


By specifying a route path, you can send CIP messages (CIP commands and responses) to a device on another CIP-based network segment via a built-in EtherNet/IP port or the EtherNet/IP Unit (CIP routing function for message communications).
The maximum number of levels of CIP routing via the ports is eight for any combination of CS, CJ, NJ, and NX-series CPU Units. Note that the number of levels of IP routing using an L3 Ethernet switch is not counted in the number of levels of CIP routing via the ports.

- NX701 CPU Unit and NX102 CPU Unit
  Because there are two built-in EtherNet/IP ports, CIP routing is possible by the CPU Unit alone.



- NJ-series CPU Unit
  By combining the built-in EtherNet/IP port and an EtherNet/IP Unit, CIP routing can be performed.



Additional Information

In CIP routing, a node (Unit) that routes information subtracts the equivalent of one hop from the timeout, deletes its own address from the route information, and relays the information to the next node (Unit).
When a timeout is specified, the timeout for the actual request service processing is set in the last hop.
In the case of relay hops, the timeout for the relay route must be added to the timeout for the request.
OMRON products that support CIP subtract 5 seconds per hop.

Version Information

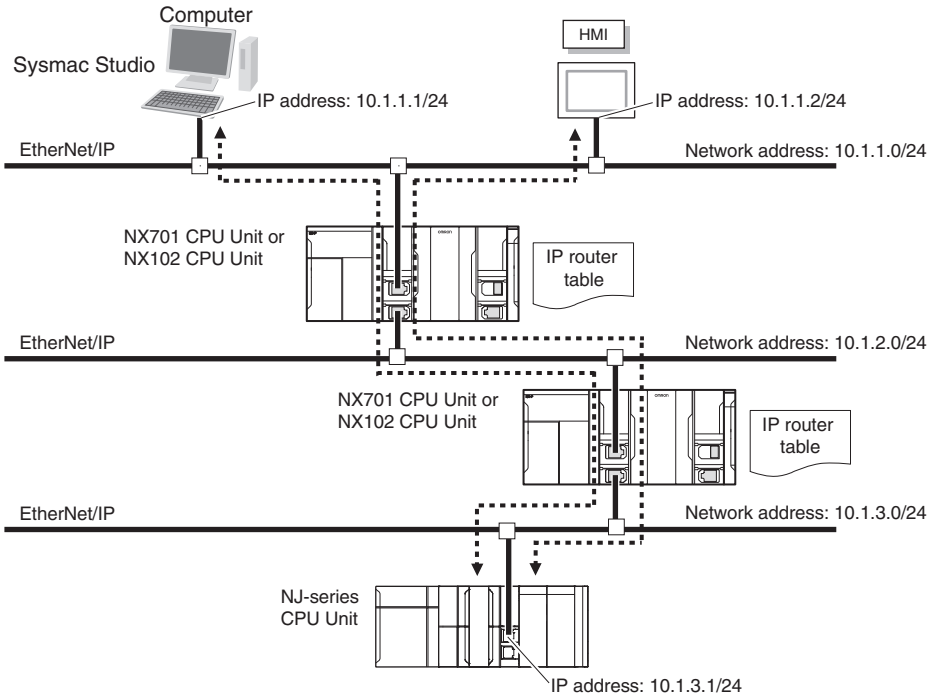
For NJ-series CPU Unit, you can use the EtherNet/IP Unit with the CPU unit version 1.01 or later and the Sysmac Studio version 1.02 or higher.

1-4-2 IP Routing

The built-in EtherNet/IP on the NX701 CPU Unit and NX102 CPU Unit have the IP routing function.
The IP routing function sends IP packets to other network segments based on the routing information set in the IP router table.
To communicate with devices on other network segments, you must set the IP router table and default gateway settings for the CPU Unit and each device on the network appropriately for your network configuration.

Precautions for Correct Use

- You cannot create tag data links between multiple CPU Units using IP routing on the NX701 CPU Unit and NX102 CPU Unit.
- The IP routing function can only be used with the NX701 CPU Units and NX102 CPU Units. IP routing cannot be used with a combination of a built-in EtherNet/IP port on an NJ-series CPU Unit and an EtherNet/IP Unit.



- **Port Forward - IP Forward**

  This function divides the network for the built-in EtherNet/IP ports 1 and 2. When you divide the network, set **IP Forward** to *Do not use*. When it is set to *Do not use*, any other IP packets than those addressed to the Controller are discarded. Refer to *4-1 **TCP/IP Settings** Display* on page 4-2 for details. This function can be used only for the NX102 CPU Unit.

Additional Information

CIP routing is not be affected by the **IP Forward** setting.

1-4-3 Packet Filter

This function filters IP packets in the receive processing at the built-in EtherNet/IP ports. While Packet Filter (Simple) is used to restrict Sysmac Studio connections, Packet Filter performs general-purpose packet filtering that does not restrict communication partner to Sysmac Studio. Specify IP addresses or TCP/UPD ports for packets that are allowed to be received.

Version Information

Packet Filer is available in the following CPU Units of stated versions.
- NJ-series, NX102, NX1P2 CPU Unit: Version 1.49 or later
- NX701 CPU Unit: Version 1.29 or later

1-4-4 Packet Filter (Simple)

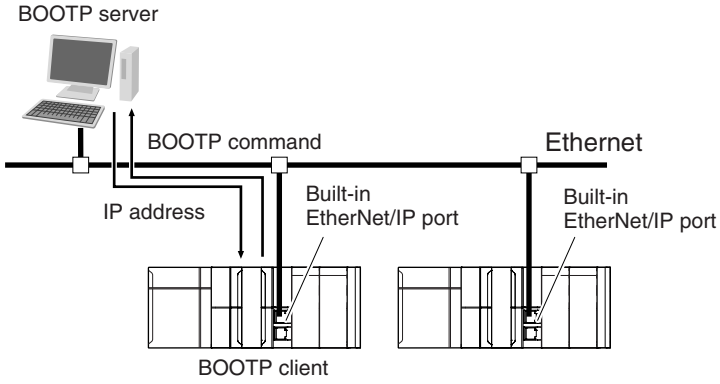
This function filters IP packets in the receive processing at the built-in EtherNet/IP ports. When Packet Filter (Simple) is enabled, it will allow you to connect the Sysmac Studio only from a computer with the preregistered IP address, and restrict any other connection from those with unregistered IP addresses. This function can be used only for NX102 CPU Unit.

Precautions for Correct Use

- Connections to NA-series and NS-series Programmable Terminals are restricted if this function is enabled. To make connections to these devices, register their IP addresses in the Packet Filter (Simple) settings. Refer to *Packet Filter (Simple)* on page 4-9 for details on the setting.
- If this function is enabled, you cannot connect the Sysmac Studio from a computer whose IP address is not registered. Before enabling this function, confirm in advance that the IP address of the computer is correctly registered.
- If this function is enabled, you cannot connect the Sysmac Studio to the Controller with the *Direct connection via Ethernet* Option selected for the connection type. Select **Controller - Communications Setup** to confirm that *Ethernet connection via a hub* is selected for connection type.
- You can disable this function tentatively by starting the Unit in Safe Mode in case you forget the registered IP address and cannot go online from the Sysmac Studio. Refer to *Troubleshooting When You Cannot Go Online from the Sysmac Studio* in the *NJ/NX-series Troubleshooting Manual (Cat. No. W503)* for details.
- You can use the Packet Filter (Simple) with Sysmac Studio version 1.49 or lower. Use the Packet Filter instead of the Packet Filter (Simple) when you use Sysmac Studio version 1.50 or higher.

1-4-5 BOOTP Client

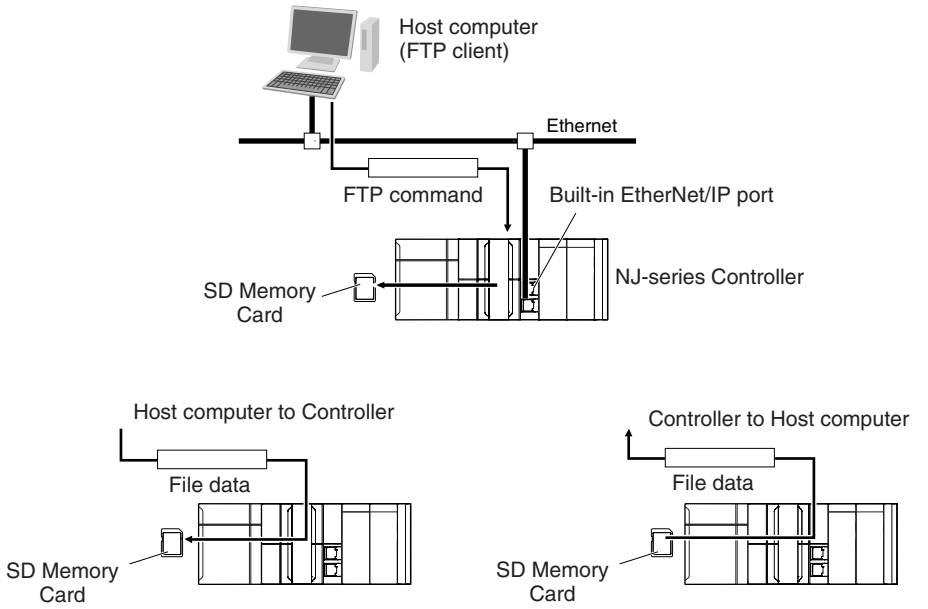
You set the built-in EtherNet/IP port in the BOOTP settings to use the BOOTP client to obtain settings, such as the built-in EtherNet/IP port IP address, from the BOOTP server.



The built-in EtherNet/IP port IP address is obtained from the BOOTP server when the power is turned ON.

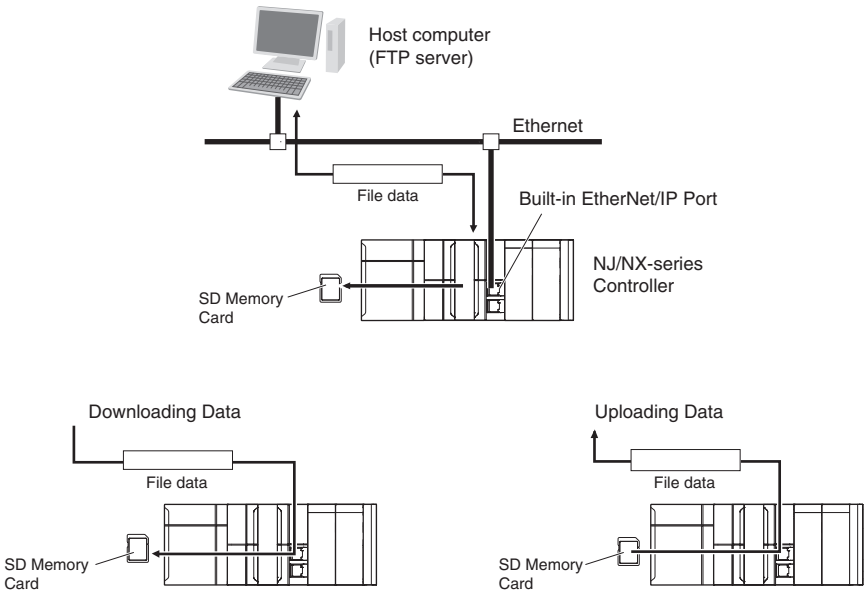
1-4-6 FTP Server

An FTP server is built into the built-in EtherNet/IP port so that files can be read from and written to the SD Memory Card in the CPU Unit of the Controller from computers at other Ethernet nodes.
This makes it possible to exchange data files between a host computer and the Controller with the host computer as the FTP client and the Controller as the FTP server.



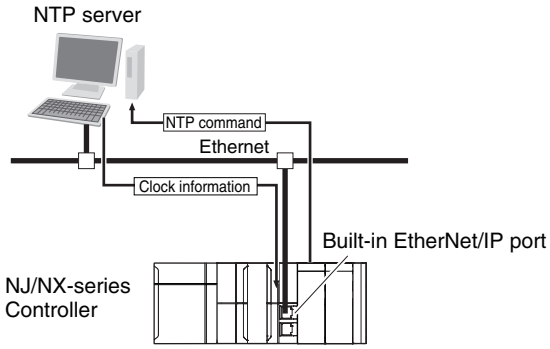
1-4-7 FTP Client

The built-in EtherNet/IP port contains an FTP client. With it, you can use FTP client communications instructions to transfer files between the CPU Unit and host computers on Ethernet.
This makes it possible to exchange data files between a host computer and the Controller with the Controller as the FTP client and the host computer as the FTP server.



1-4-8 Automatic Clock Adjustment

With the built-in EtherNet/IP port, clock information is read from the NTP server at the specified time or at a specified interval after the power supply to the CPU Unit is turned ON. The internal clock time in the CPU Unit is updated with the read time.

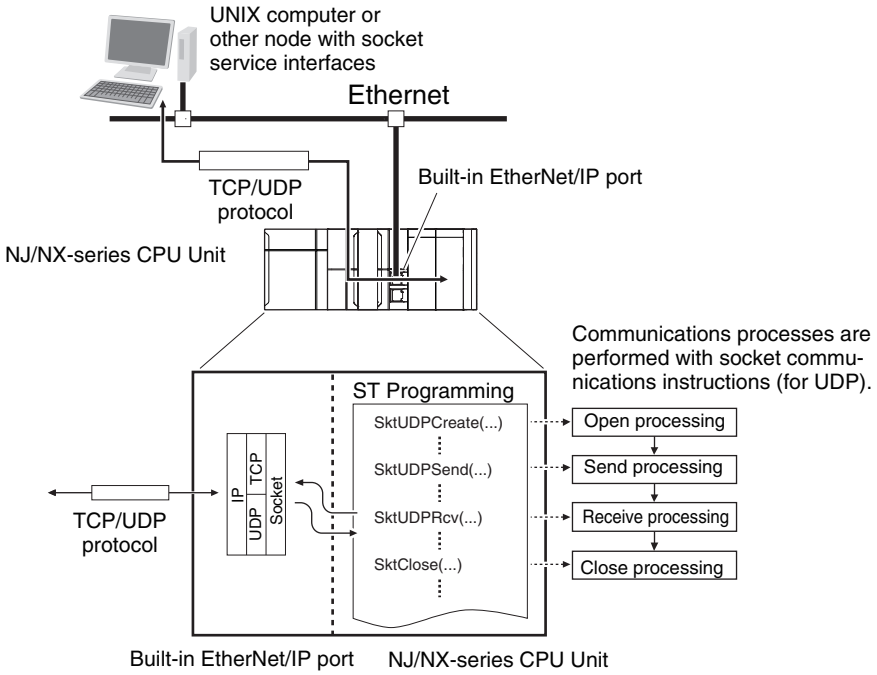


Precautions for Correct Use

An NTP server is required to use automatic clock adjustment.

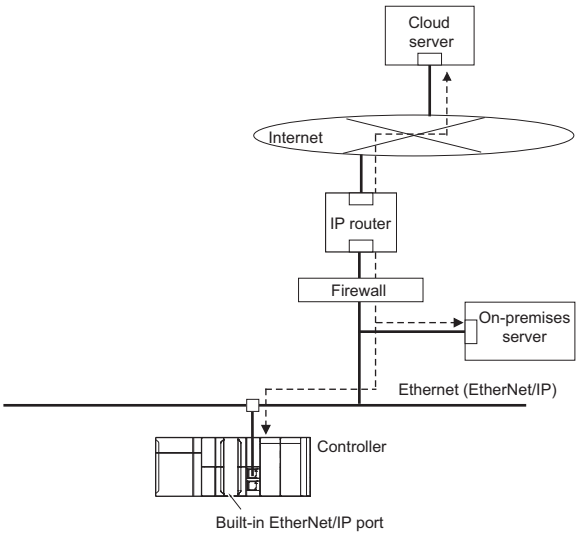
1-4-9 Socket Service

You can send data to and receive data from any node on Ethernet with the UDP or TCP protocol.
To send/receive data with a socket service, you execute multiple socket communications instructions in sequence in an ST program to execute the required communications processes.
After a connection with the other communications device is opened with an open instruction, the values of the variables that are specified for the send instruction are sent and the data that was received for a receive instruction is stored in the specified variables.
The connection is closed with a close instruction, and communications end.
For TCP, you can also read the socket status and received data.
You can use a total of 30 TCP ports and UDP ports. (A total of 60 ports for an NX102 CPU Unit)



1-4-10 Secure Socket Services

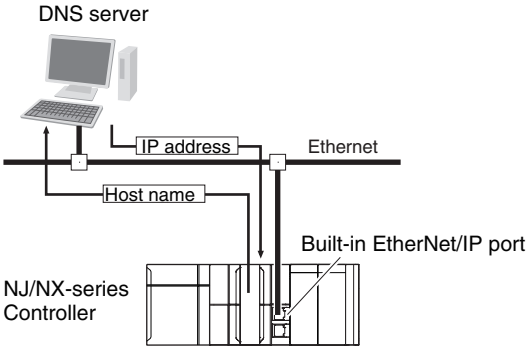
The secure socket services allow the built-in EtherNet/IP port on the CPU Unit to act as a client, enabling secure socket communications with the on-premises server on the private network or with the cloud server on the external network.
This function performs encrypted communications using TLS, which use client private keys and certificates, and enables safe communications.



1-4-11 Specifying Host Names

You can directly specify IP addresses, but you can also use the host names instead of the IP addresses for NTP servers, SNMP managers, or the destinations of socket instructions and CIP communications instructions (DNS client or hosts settings).

Example: Setting Host Names on the DNS Server



Precautions for Correct Use

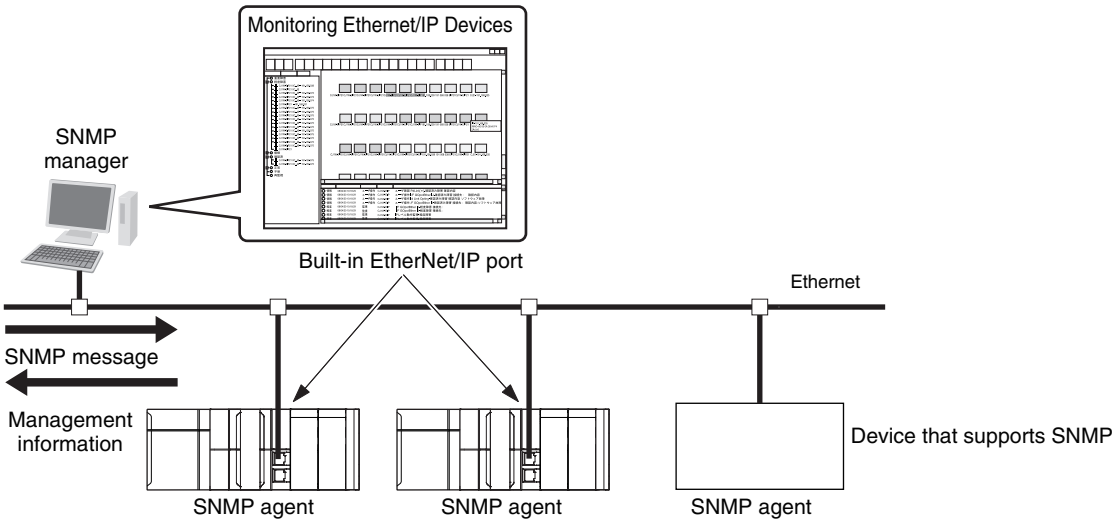
- A DNS server is required to use the server host names for the DNS client.

1-4-12 SNMP Agent

The SNMP agent has the following functions.

SNMP Agent

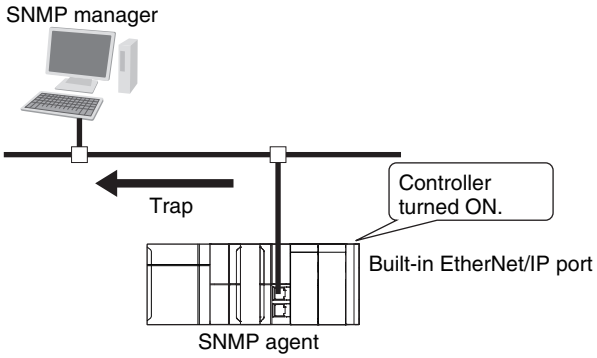
The SNMP agent passes internal status information from the built-in EtherNet/IP port to network management software that uses an SNMP manager.



SNMP Trap

When specific conditions occur, the built-in EtherNet/IP port that is set as the SNMP agent sends status notification reports to the SNMP manager.
The SNMP manager can learn about changes in status even without periodically monitoring the built-in EtherNet/IP port.
Status notification reports are sent under the following conditions.

- When the Controller is turned ON
- When links are established
- When an SNMP agent fails to be authorized



1-4-13 TCP/UDP Message Service

This function supports TCP/UDP socket communications, which allow simple access to CIP objects of the Controller from a system where EtherNet/IP is not supported. This will allow you to change settings and perform I/O control for NX Units connected to the Controller or the NX bus.
You can use the TCP/UDP message service only for the NX102 CPU Units.

1-5 EtherNet/IP Communications Procedures

● Basic Operation

1 Wire the Ethernet network with twisted-pair cable.

Section 2 Installing Ethernet Networks on page 2-1

↓

2 Set the built-in EtherNet/IP port IP address with the Sysmac Studio.

5-1 Determining IP Addresses on page 5-2

1. Use the Sysmac Studio to create a new project.
2. Set the local IP address in one of the following ways:

- Defaults:

NX701 CPU Unit		
	Built-in EtherNet/IP port 1	: 192.168.250.1 (subnet mask = 255.255.255.0)
	Built-in EtherNet/IP port 2	: 192.168.251.1 (subnet mask = 255.255.255.0)
NX102 CPU Unit		
	Built-in EtherNet/IP port 1	: 192.168.250.1 (subnet mask = 255.255.255.0)
	Built-in EtherNet/IP port 2	: 192.168.251.1 (subnet mask = 255.255.255.0)
NX1P2 CPU Unit Built-in EtherNet/IP port		: 192.168.250.1 (subnet mask = 255.255.255.0)
NJ-series CPU Unit Built-in EtherNet/IP port		

- Set any IP address.
- Obtain from the BOOTP server.

↓

3 Perform a communications test with a PING command from a computer.

5-3 Testing Communications on page 5-20

↓

4 Use the Sysmac Studio to set the initial settings of the EtherNet/IP Function Module.

Section 4 Sysmac Studio Settings for the Built-in EtherNet/IP Port on page 4-1

- Set the TCP/IP settings and Ethernet settings as required.

● Using Tag Data Links

1 Import the variable settings for the tags that were created on the Sysmac Studio to the Network Configurator.

6-2-4 Creating Tags and Tag Sets on page 6-23

↓

2 Use the Network Configurator to create the tag data link table.

Section 6 Tag Data Link Functions on page 6-1

- Create the network configuration.
- Set the tags, tag sets, and connections.

↓

3 Connect the Network Configurator online.

↓

4 Download the tag data link setting.

↓

5 **Start the tag data links (the links starts automatically when power is turned ON).**

↓

6 **Check operation.**

1-3-2 Part Names and Functions on page 1-12
Section 15 Troubleshooting on page 15-1

- Check the built-in EtherNet/IP port indicators.
- Use the Sysmac Studio to check the communications status with the All Tag Data Link Communications Status system-defined variable.
- Use the monitor function of the Network Configurator to confirm that the tag data links are in normal operation.

● Using the Message Communications Service

- CIP Communications Instructions

1 **Execute CIP communications instructions in the user program.**

Section 7 CIP Message Communications on page 7-1

↓

2 **Check operation.**

1-3-2 Part Names and Functions on page 1-12
Section 15 Troubleshooting on page 15-1

- Use the Sysmac Studio to check the communications status with the end codes of the instructions (Done, Err, and ErrID).

● Using the Socket Services

1 **Execute the socket service instructions in the user program.**

Section 8 Socket Service on page 8-1

↓

2 **Check operation.**

- Use the Sysmac Studio to check the communications status with the end codes of the instructions (Done, Err, and ErrorID).

● Using the FTP Server

1 **Use the Sysmac Studio to set the initial settings of the EtherNet/IP Function Module.**

Section 10 FTP Server on page 10-1

- Set the FTP settings (enabling FTP, login name, and password).

↓

2 **Connect to the FTP server in the NJ-series CPU Unit from an FTP client application.**

- Input the FTP login name and password to log onto the built-in EtherNet/IP port.
- Check the event log to see if the FTP server started.

● Using the Automatic Clock Adjustment

1 **Use the Sysmac Studio to set the initial settings of the EtherNet/IP Function Module.**

Section 12 Automatic Clock Adjustment on page 12-1

- Set the NTP settings (enabling NTP and execution conditions).

↓

2 Execute automatic clock adjustment.

- Execute automatic adjustment at specified times or specified intervals.
- Use the Sysmac Studio to check the NTP Last Operation Time and NTP Operation Result system-defined variables.
- Check the event log to see if the NTP client started.

● Using the SNMP Agent

1 Use the Sysmac Studio to set the initial settings of the EtherNet/IP Function Module.

Section 13 SNMP Agent on page 13-1

- Set the SNMP settings.
- Set the SNMP trap settings.

↓

2 Check operation.

- Check the event log to see if the SNMP agent started.

● Using BOOTP

1 Use the Sysmac Studio to set the initial settings of the EtherNet/IP Function Module.

Section 4 Sysmac Studio Settings for the Built-in EtherNet/IP Port on page 4-1

- Set the BOOTP settings.

↓

2 Check operation.

- Check the event log to see if BOOTP started.
- Check the Online system-defined variable.

2

Installing Ethernet Networks

2-1 Selecting the Network Devices

2-1-1 Recommended Network Devices

The following table shows the devices recommended for use with the EtherNet/IP.

● Ethernet Switches

Manufacturer	Model	Description
OMRON	W4S1-03B W4S1-05B W4S1-05C	Packet priority control (QoS): EtherNet/IP control data priority Failure detection: Broadcast storm, LSI error detection, 100Basae-TX/ 10Base-T, Auto negotiation Number of ports: three for the W4S1-03B, or five each for the W4S1-05B and W4S1-05C Failure detection output (W4S1-05C only)
Cisco Systems, Inc.	Consult the manufacturer. http://www.cisco.com/	
Contec USA, Inc.	Consult the manufacturer. http://www.contec.com/	
Phoenix Contact USA	Consult the manufacturer. https://www.phoenixcontact.com	

● Twisted-pair Cables and Connectors

Applicable EtherNet/IP communications cables and connectors vary depending on the used baud rate.

For 100Base-TX and 10Base-T, use an STP (shielded twisted-pair) cable of category 5 or higher. You can use either straight or cross cable.

For 1000Base-T, use an STP (shielded twisted-pair) cable (double shielding with aluminum tape and braiding) of category 5e or higher. You can use either straight or cross cable.

Cabling materials used for EtherNet/IP communication cables are shown in the table below. "100Base-TX" in the "Product" column of the table below indicates that either 100Base-TX or 10Base-T can be used.

Product			Manufacturer	Model
For 1000Base-T and 100Base-TX	Size and conductor pairs: AWG 24 × 4 pairs *1	Cable	Hitachi Metals, Ltd.	NETSTAR-C5E SAB 0.5 × 4P CP
			Kuramo Electric Co.	KETH-SB
			JMACS Japan Co., Ltd.	IETP-SB
		RJ45 Connectors	Panduit Corporation	MPS588
For 100Base-TX	Size and conductor pairs: AWG22 × 2P*1	Cable	Kuramo Electric Co., Ltd.	KETH-PSB-OMR
			JMACS Japan Co., Ltd.	PNET/B
		RJ45 Assembly Connectors 	OMRON	XS6G-T421-1

*1. We recommend that you use cables and connectors in above combinations.

2-1-2 Ethernet Switch Types

● Unmanaged Layer 2 (L2) Ethernet Switches

These Ethernet switches use the Ethernet MAC address to switch ports. Ordinary Ethernet switches have this function. Ethernet switch functions and settings cannot be changed.

● Managed Layer 2 (L2) Ethernet Switches

These Ethernet switches use the Ethernet MAC address to switch ports. Ethernet switch functions and settings can be changed with special software tools for Ethernet switches running on a network node. You can also collect analytical data. These Ethernet switches provide more-advanced functions than unmanaged layer 2 Ethernet switches.

2-1-3 Ethernet Switch Functions

This section describes the Ethernet switch functions that are important for an EtherNet/IP network. For a built-in EtherNet/IP port, consider whether the Ethernet switch supports these functions when you select the Ethernet switch.

- Multicast filtering
- QoS (Quality of Service) for TCP/UDP port numbers (L4)

● Multicast Filtering

Multicast filtering transfers multicast packets to the specific nodes only. This function is implemented in the Ethernet switch as IGMP snooping or GMRP.

“Specific nodes” are nodes equipped with an IGMP client, and have made transfer requests to the Ethernet switch. (OMRON built-in EtherNet/IP ports are equipped with an IGMP client.) When the Ethernet switch does not use multicast filtering, multicast packets are sent to all nodes, just like broadcast packets, which increases the traffic in the network.

Settings must be made in the Ethernet switch to enable this function. There must be enough multicast filters for the network.

● QoS (Quality of Service) Function for TCP/UDP Port Numbers (L4)

This function controls the priority of packet transmissions so that packets can be sent with higher priority to a specific IP address or TCP (UDP) port. The TCP and UDP protocols are called transport layer protocols, leading to the name L4 (layer 4) QoS function.

When tag data links and message communications are executed on the same network, tag data links can be sent at higher priority to prevent problems such as transmission delays due to message communications traffic and packet losses due to buffer overflow.

Settings must be made in the Ethernet switch to enable QoS function and give higher priority to tag data link packets.

These functions are supported by Ethernet switches as described in the table below.

Ethernet switch type	Multicast filtering	L4 QoS	Remarks
Unmanaged L2 Ethernet switch	Not supported	Not supported	---
Managed L2 Ethernet switch	Supported	Supported	Both functions must be set with a special software tool.

Ethernet switch type	Multicast filtering	L4 QoS	Remarks
OMRON Ethernet switch (W4S1-series Ethernet switches)	Not supported	Supported	L4 QoS is set with a switch. No software tool is necessary. *QoS (Quality of Service) Function for TCP/UDP Port Numbers (L4)* on page 2-3

Additional Information

If the Network Configurator is used to set the connection type in the connection settings to a **Multicast Connection**, multicast packets are used. If the connection type is set to a **Point to Point Connection**, multicast packets are not used.

2-1-4 Precautions for Ethernet Switch Selection

The functions supported by the Ethernet switch may affect tag data link transmission delays and the settings in the Controller configurations and setup.
In addition, if the Ethernet switch supports advanced functions, special settings are required for the functions.
When you select an Ethernet switch, it is necessary to consider what kind of data transmission and how much traffic you use over the the network.
Refer to the following precautions when you select an Ethernet switch.
Refer to *14-2 Adjusting the Communications Load* on page 14-7 to estimate the communications load for tag data links.

Selecting the Ethernet Switch Based on the Type of Network Communications

● Executing Tag Data Links Only

We recommend that you use an L2 Ethernet switch without multicast filtering or an L2 Ethernet switch with multicast filtering.
An L2 Ethernet switch with multicast filtering prevents increased traffic due to unnecessary multicast packets, so the tag data links can operate at higher speed.
If either of the following conditions exists, there is no difference in the traffic condition whether multicast filtering is supported or not.

- The tag data links are set to share the same data with all nodes in the network. (Multicast packets are transferred to all nodes in the network, just like broadcast transmission.)
- The tag data link settings are all one-to-one (unicast) and multicast packets cannot be used.

When multicast filtering is used, settings must be made accordingly on the Ethernet switch. There must be enough multicast filters for the network.

● Executing Tag Data Links and Message Communications

We recommend an L2 Ethernet switch with multicast filtering and L4 QoS.
If you set tag data links for higher-priority transmission, it is possible to prevent problems such as transmission delays due to message communications traffic and packet losses due to buffer overflow.
When multicast filtering and L4 QoS are used, settings must be made accordingly on the Ethernet switch.

Selecting the Ethernet Switch Based on the Ethernet Switch's Supported Functions

● L2 Ethernet Switch without Multicast Filtering

We recommend this kind of Ethernet switch when only tag data links are executed and any of the following conditions is met.

- The tag data links are set to share the same data with all nodes in the network. (Multicast packets are transferred to all nodes in the network, just like broadcast transmission.)
- The tag data link settings are all one-to-one (unicast) and multicast packets cannot be used.
- There is little traffic in the tag data links.

No special settings are required for an L2 Ethernet switch without multicast filtering.

● L2 Ethernet Switch with Multicast Filtering

We recommend this kind of Ethernet switch when only tag data links are executed and the following condition is met.

- There are many 1:N links (where N represents some number of nodes in the network) in the tag data link settings, i.e., there are many multicast packets used, or there is heavy traffic in the tag data links.

Specific settings are required for an L2 Ethernet switch with multicast filtering. There must be enough multicast filters for the network.

● L3 Ethernet Switch with Multicast Filtering and L4 QoS Functions

We recommend this kind of Ethernet switch when both tag data links and message communications are executed.

If you set tag data links for higher-priority transmission, you can prevent problems such as transmission delays due to message communications traffic and packet losses due to buffer overflow.

When multicast filtering and L4 QoS are used, settings must be made accordingly on the Ethernet switch. There must be enough multicast filters for the network.

Selecting the Ethernet Switch Based on the Network Communication Speed

● Executing Tag Data Links at a Baud Rate Over 100 Mbps

If you use data tag links with the following conditions, use an Ethernet switch with multicast filtering or an Ethernet switch that supports a baud rate of 1,000 Mbps.

- Multicast
- Baud rate over 100 Mbps

If there is an Ethernet device on the same network that communicates at a speed of 100 Mbps or less, the device may affect tag data link communications and cause tag data links to be broken, even if the device is not related to tag data link communications.

Precautions for Correct Use

- Ask the Ethernet switch manufacturer for setting procedures for the Ethernet switch.
- Install the Ethernet switch based on its environmental resistance specifications so that the environmental resistance specifications are fully met. Ask the Ethernet switch manufacturer for information on the environmental resistance of the Ethernet switch.

2-2 Network Installation

2-2-1 Basic Installation Precautions

- Take the greatest care when you install the Ethernet System. Be sure to follow ISO 8802-3 specifications. Be sure you understand them before attempting to install an Ethernet System.
- Unless you are already experienced in installation of communications systems, we strongly recommend that you employ a professional to install your system.
- Do not install Ethernet equipment near sources of noise.
  If a noisy environment is unavoidable, take adequate measures against noise interference, such as installation of network components in metal cases or the use of optical cable in the system.
- When using a shielded cable with the shields on both ends of the cable connected to connector hoods, ground loops induced by improper grounding methods may decrease noise immunity and cause device damage. To prevent ground loops caused by differences in potential between device grounding points, the reference potential between the devices must be stabilized. Design grounding appropriately so that noise current does not flow to ground lines between the devices.
  For grounding methods, refer to the *NJ-series CPU Unit Hardware User's Manual (Cat. No. W500)*, *NX-series CPU Unit Hardware User's Manual (Cat. No. W535)*, *NX-series NX102 CPU Unit Hardware User's Manual (Cat. No. W593)*, or *NX-series NX1P2 CPU Unit Hardware User's Manual (Cat. No. W578)*.
- To obtain information on installing EtherNet/IP cable, contact ODVA.
  ODVA web site: http://www.odva.org
- When you install an EtherNet/IP network that combines an information network with the control system, and the communications load may be heavy due to tag data links, we recommend that you set up a network where the load does not affect communications. For example, install the tag data links in a segment that is separate from the information network.

2-2-2 Recommended Network Devices

Refer to *2-1 Selecting the Network Devices* on page 2-2 for the devices recommended for use with the built-in EtherNet/IP port.

2-2-3 Precautions When Laying Twisted-pair Cable

Connecting the Shield to Connector Hoods

● Between an EtherNet/IP Port and an Ethernet Switch

Connect the shield to connector hoods as described below.

NJ-series CPU Unit		NX-series CPU Unit		
10Base-T	100Base-TX	10Base-T	100Base-TX	1000Base-T *1
• Connect the shield at both ends or • Connect the shield only at the Ethernet switch side		• Connect the shield at both ends or • Connect the shield only at the Ethernet switch side. A clamp core must be attached to the EtherNet/IP port side of the cable.		Connect the shield at both ends

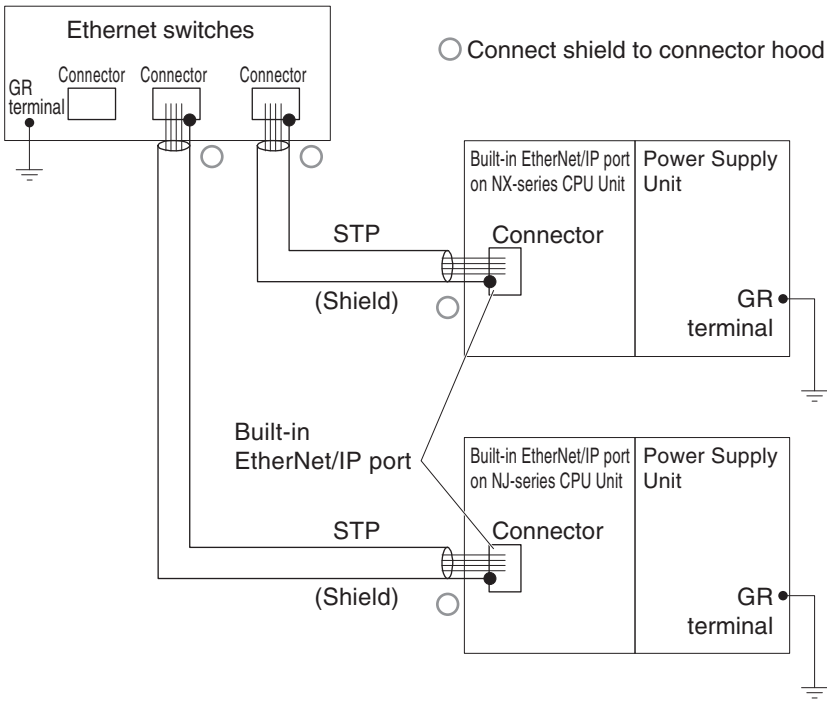
*1. For NX701 CPU Units only.

- 10Base-T or 100Base-TX

  Connect the cable shields to the connector hoods as described in either (1) or (2) below.

  1. Connecting the shields at both ends of the cable

     Connect the shields to the connector hoods at both ends of the cables.

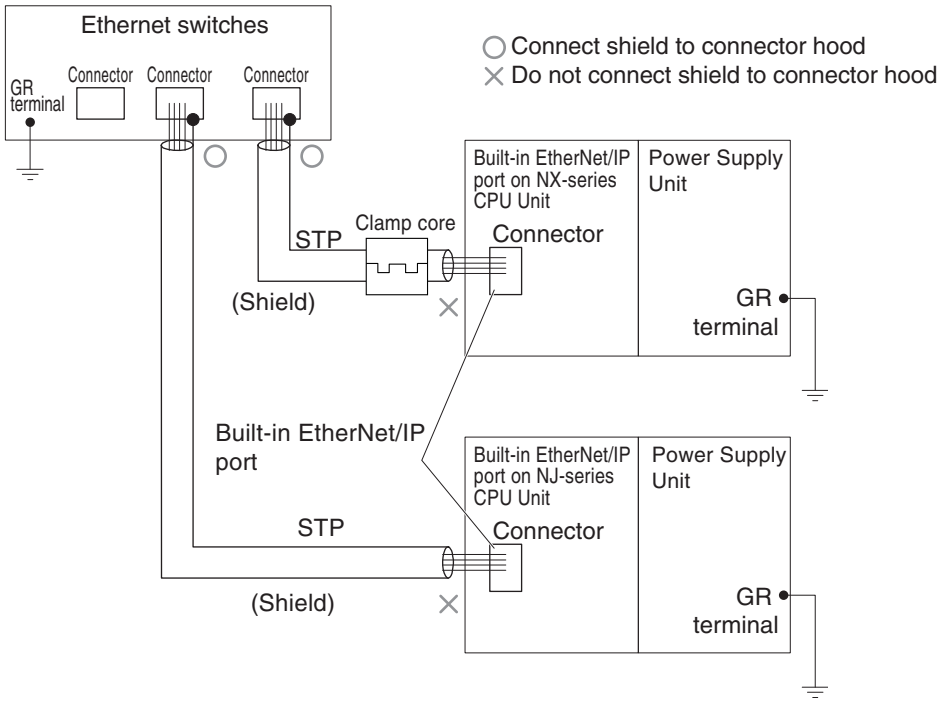


  2. Connecting the shields only at the Ethernet switch side

     Connect the shields to the connector hoods only at the Ethernet switch side.

     - For an NX-series CPU Unit, a clamp core must be attached to the end of the cable at the EtherNet/IP port side. For a recommended clamp core and attachment methods, refer to *Recommended Clamp Core and Attachment Method* on page 2-10.

       To comply with EMC standards, it is mandatory that a clamp core be attached when connecting the shield to the connector hood only at the Ethernet switch side.
     - For an NJ-series CPU Unit, it is not necessary to attach a clamp core.



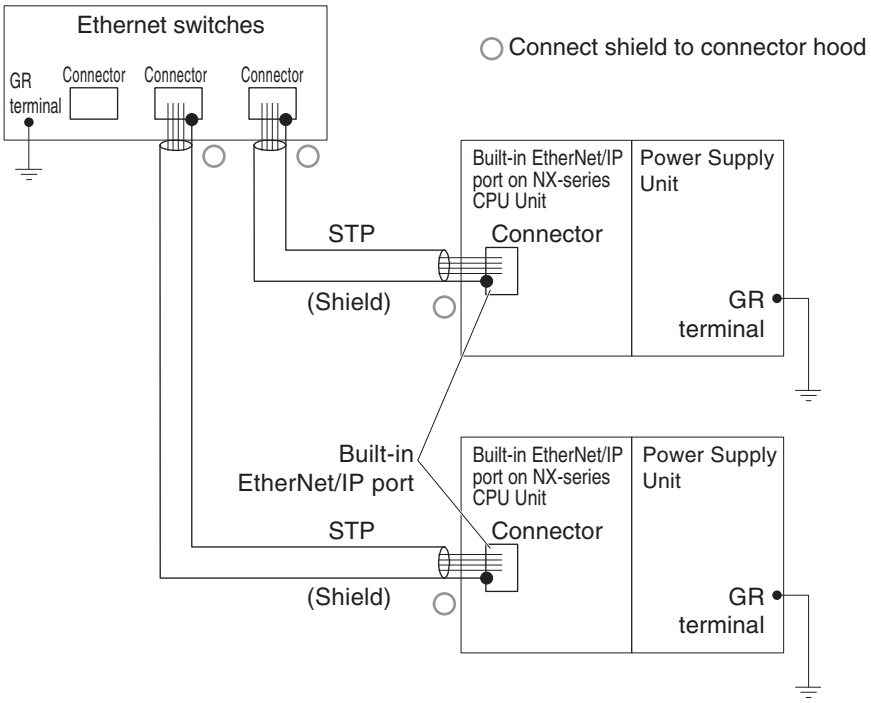
Additional Information

Noise immunity may be reduced and device damage may occur due to ground loops, which may be caused by improper shield connections and grounding methods.
When using a baud rate of 100 Mbps or less, it may be possible to alleviate this problem by connecting the shield only at the Ethernet switch side as described in (2), rather than connecting both ends as described in (1).

- 1000Base-T

  Connect the shields to respective connector hoods at both ends of the cables.

  This connection is required for 1000Base-T to ensure compliance with EMC standards.



● Between Two Ethernet Switches

Regardless of which baud rate is used, check with the Ethernet switch manufacturers for information about installing the network between Ethernet switches, and in particular whether or not it is necessary to connect the cable shields to the connector hoods.

Other Precautions When Laying the Twisted-pair Cable

- Firmly insert the connector until it locks into place when you connect the cable to the Ethernet switch and the built-in EtherNet/IP port.
- Do not install the twisted-pair cable together with high-voltage lines.
- Do not install the twisted-pair cable near devices that generate noise.
- Do not install the twisted-pair cable in locations subject to high temperatures or high humidity.
- Do not install the twisted-pair cable in locations subject to excessive dirt, dust, oil mist or other contaminants.

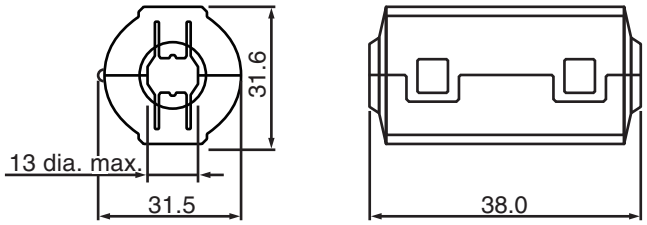
Recommended Clamp Core and Attachment Method

When you use an NX-series CPU Unit and connect the cable shield only with the connector hood of the Ethernet switch, you need to attach a clamp core to the EtherNet/IP port of the CPU Unit.
The recommended clamp core and attachment method are given below.

● Recommended Clamp Core

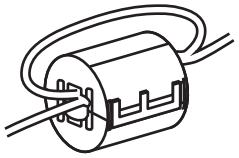
Manufacturer	Product	Model
NEC TOKIN	Clamp core	ESD-SR-250

ESD-SR-250 dimensions



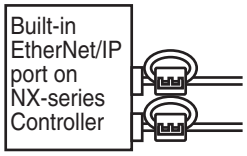
● Recommended Attachment Method

- Attach a clamp core to the communications cable as shown below.



Make two loops with the cable as shown.

- Connect the communications cable as shown below.



Attach close to the cable connection as shown.

2-2-4 Precautions When Installing and Connecting Ethernet Switches

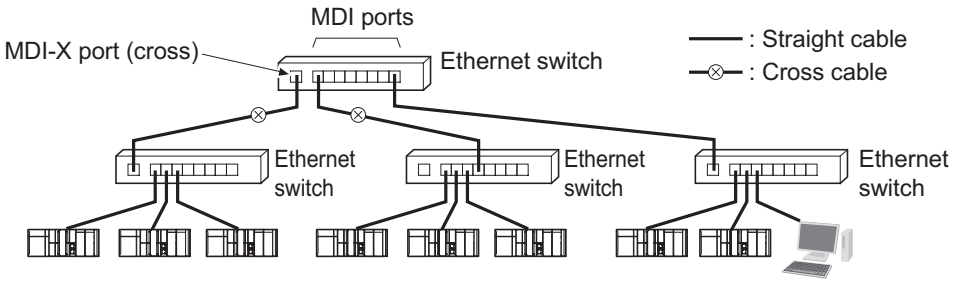
Precautions When Installing Ethernet Switches

- Do not ground the Ethernet switch in the same location as a drive-system component, such as an inverter.
- Always use a dedicated power supply for the Ethernet switch. Do not use the same power supply for other equipment, such as an I/O power supply, motor power supply, or control power supply.
- Before installation, check the Ethernet switch's environmental resistance specifications, and use an Ethernet switch that is appropriate for the ambient conditions. Contact the Ethernet switch manufacturer for details on Ethernet switch's environmental resistance specifications.

Ethernet Switch Connection Methods

- Connect Ethernet switches with twisted-pair cables, as follows: Connect an MDI port to an MDI-X port with a straight cable. Connect two MDI ports or two MDI-X ports with a cross cable.

  Note It is very difficult to distinguish cross cables and straight cables by appearance. Incorrect cables will cause communications to fail. We recommend cascade connections with straight cables wherever possible.



- Some Ethernet switches can automatically distinguish between MDI and MDI-X. When this kind of Ethernet switch is used, straight cable can be used between Ethernet switches.

Precautions for Correct Use

Adjust the built-in EtherNet/IP port's link settings to match the communications mode settings of the connected Ethernet switch. If the settings do not match, the link will be unstable and prevent normal communications. The following table shows the allowed settings for each Ethernet switch communications mode.
(Auto-Nego: Auto negotiation, Full: Full duplex, Half: Half duplex)

Ethernet switch		Built-in EtherNet/IP port					
		Auto-Nego	10 Mbps (fixed)		100 Mbps (fixed)		1,000 Mbps (fixed)
			Full	Half	Full	Half	Full
Auto-Nego		Best	---	OK	---	OK	---
10 Mbps (fixed)	Full	---	OK	---	---	---	---
	Half	OK	---	OK	---	---	---
100 Mbps (fixed)	Full	---	---	---	OK	---	---
	Half	OK	---	---	---	OK	---
1,000 Mbps (fixed)	Full	---	---	---	---	---	Best

Best = Recommended; OK = Allowed; --- = Not allowed.

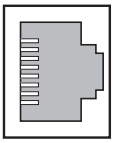
2-3 Connecting to the Network

2-3-1 Ethernet Connectors

The following standards and specifications apply to the connectors for the Ethernet twisted-pair cable.

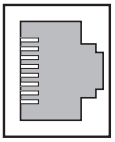
- Electrical specifications: Conforming to IEEE 802.3 standards.
- Connector structure: RJ45 8-pin Modular Connector (conforming to ISO 8877)
- For information on connecting shield wire to connector hoods, refer to *2-1-2 Ethernet Switch Types* on page 2-3.

10Base-T and 100Base-TX



Connector pin	Signal name	Abbr.	Signal direction
1	Transmission data +	TD+	Output
2	Transmission data −	TD−	Output
3	Reception data +	RD+	Input
4	Not used	---	---
5	Not used	---	---
6	Reception data −	RD−	Input
7	Not used	---	---
8	Not used	---	---

1000Base-T



Connector pin	Signal name	Abbr.	Signal direction
1	Communication data DA+	BI_DA+	Input/output
2	Communication data DA−	BI_DA-	Input/output
3	Communication data DB+	BI_DB+	Input/output
4	Communication data DC+	BI_DC+	Input/output
5	Communication data DC−	BI_DC−	Input/output
6	Communication data DB−	BI_DB−	Input/output
7	Communication data DD+	BI_DD+	Input/output
8	Communication data DD−	BI_DD−	Input/output

2-3-2 Connecting the Cable

Precautions for Correct Use

- Turn OFF the Controller's power supply before connecting or disconnecting Ethernet communications cable.
- Allow extra space for the bending radius of the communications cable.
  For the CPU Unit dimensions when the communications cable is connected to the Unit, refer to the *NJ-series CPU Unit Hardware User's Manual (Cat. No. W500)*, *NX-series CPU Unit Hardware User's Manual (Cat. No. W535)*, *NX-series NX102 CPU Unit Hardware User's Manual (Cat. No. W593)*, or *NX-series NX1P2 CPU Unit Hardware User's Manual (Cat. No. W578)*. The required space depends on the communications cable and connector that are used. Consult the manufacturer or sales agent.

1. Install the twisted-pair cable.
2. Connect the cable to the Ethernet switch.
3. Connect the twisted-pair cable to the connector on the built-in EtherNet/IP port.
   Be sure to press the connectors (both the Ethernet switch side and Ethernet side) until they lock into place.

3

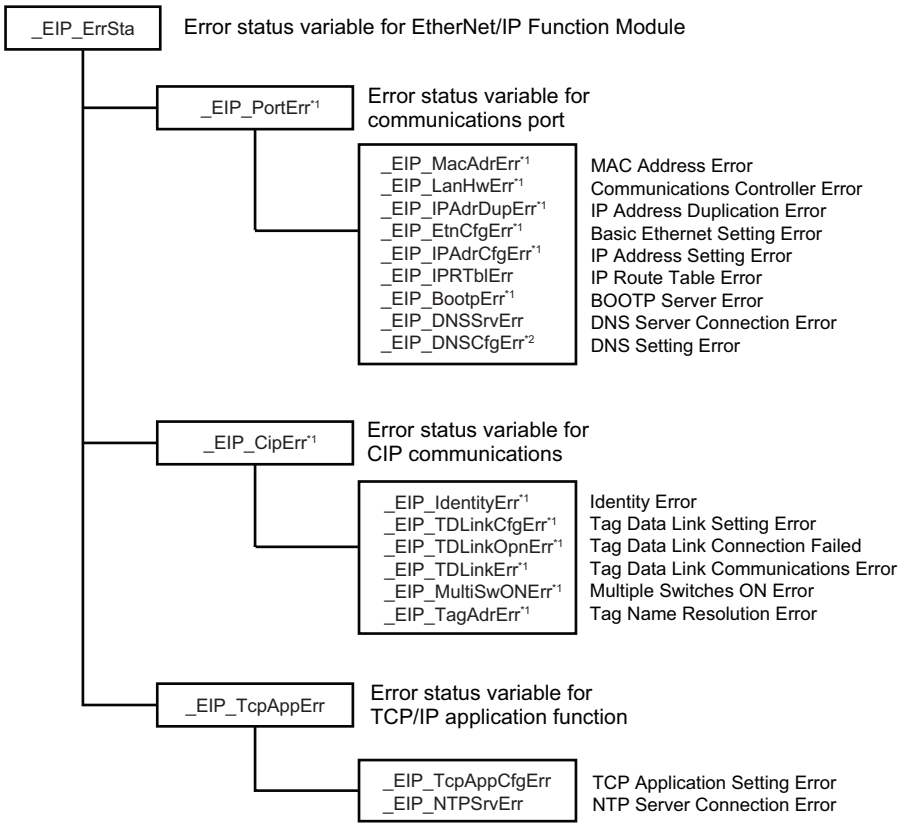
System-defined Variables Related to the Built-in EtherNet/IP Port

3-1 System-defined Variables Related to the Built-in EtherNet/IP Port

You can use the system-defined variables that are provided for the built-in EtherNet/IP port in programs to check the status of the built-in EtherNet/IP port.

● Checking for Errors in the Built-in EtherNet/IP Port

You can check for built-in EtherNet/IP port errors, Sysmac Studio setting errors, Network Configurator setting errors, TCP/IP application errors (e.g., FTP or NTP), etc.
The following hierarchy is used. The system gives the error status at each level by logically ORing the error status information in the next lower level.



*1. Error status variables for errors related to NX-series CPU Units are provided individually for communications port 1 and communications port 2. You can use error status variables for communications port 2 with the NX701 CPU Units and NX102 CPU Units only.
Refer to *Hierarchical Relationship of System-defined Variables Related to EtherNet/IP Errors in the NX-series CPU Unit page 3-18* for details.

*2. With the NJ-series CPU Unit, this variable can be used with the unit version 1.11 or later.

3-2 System-defined Variables

The variables are described in the tables as shown below.

Variable name	Meaning	Function	Data type	Range of values	Reference
This is the system-defined variable name. The prefix gives the category name.	This is the meaning of the variable.	The function of the variable is described.	The data type of the variable is given.	The range of values that the variable can take is given.	The page of the individual system-defined variable specifications table is given.

● Functional Classification: EtherNet/IP Communications Errors

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_ErrSta	Built-in EtherNet/IP Error	This is the error status variable for the built-in EtherNet/IP port. NX-series CPU Units: Represents the collective status of the following error flags. • _EIP1_PortErr (Communications Port1 Error) • _EIP2_PortErr (Communications Port2 Error) • _EIP1_CipErr (CIP Communications1 Error) • _EIP2_CipErr (CIP Communications2 Error) • _EIP_TcpAppErr (TCP Application Communications Error) NJ-series CPU Units: Represents the collective status of the following error flags. • _EIP_PortErr (Communications Port Error) • _EIP_CipErr (CIP Communications Error) • _EIP_TcpAppErr (TCP Application Communications Error) **Note** Refer to *Meanings of Error Status Bits* for the meanings of the error status bits.	WORD	16#0000 to 16#00F0	page 3-35

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_PortErr	Communications Port Error	This is the error status variable for the communications port. NX-series CPU Units: Represents the collective status of the following error flags. • _EIP1_MacAdrErr (Port1 MAC Address Error) • _EIP1_LanHwErr (Port1 Communications Controller Error) • _EIP1_EtnCfgErr (Port1 Basic Ethernet Setting Error) • _EIP1_IPAdrCfgErr (Port1 IP Address Setting Error) • _EIP1_IPAdrDupErr (Port1 IP Address Duplication Error) • _EIP1_BootpErr (Port1 BOOTP Server Error) • _EIP_DNSCfgErr (DNS Setting Error) • _EIP_DNSSrvErr (DNS Server Connection Error) • _EIP_IPRTblErr (IP Route Table Error) NJ-series CPU Units: Represents the collective status of the following error flags. • _EIP_MacAdrErr (MAC Address Error) • _EIP_LanHwErr (Communications Controller Error) • _EIP_EtnCfgErr (Basic Ethernet Setting Error) • _EIP_IPAdrCfgErr (IP Address Setting Error) • _EIP_IPAdrDupErr (IP Address Duplication Error) • _EIP_BootpErr (BOOTP Server Error) • _EIP_IPRTblErr (IP Route Table Error) **Note** If a *Link OFF Detected* or *Built-in EtherNet/IP Error* occurs, it is recorded in the event log and then the corresponding bit turns ON. Refer to *Meanings of Error Status Bits* for the meanings of the error status bits.	WORD	16#0000 to 16#00F0	page 3-36

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP1_PortErr	Communications Port1 Error	This is the error status variable for the communications port 1. Represents the collective status of the following error flags. • _EIP1_MacAdrErr (Port1 MAC Address Error) • _EIP1_LanHwErr (Port1 Communications Controller Error) • _EIP1_EtnCfgErr (Port1 Basic Ethernet Setting Error) • _EIP1_IPAdrCfgErr (Port1 IP Address Setting Error) • _EIP1_IPAdrDupErr (Port1 IP Address Duplication Error) • _EIP1_BootpErr (Port1 BOOTP Server Error) • _EIP_DNSCfgErr (DNS Setting Error) • _EIP_DNSSrvErr (DNS Server Connection Error) • _EIP_IPRTblErr (IP Route Table Error) **Note** If a *Link OFF Detected* or *Built-in EtherNet/IP Error* occurs, it is recorded in the event log and then the corresponding bit turns ON. Refer to *Meanings of Error Status Bits* for the meanings of the error status bits. **Note** You can use this system-defined variable only for NX-series CPU Units.	WORD	16#0000 to 16#00F0	page 3-36

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP2_PortErr	Communications Port2 Error	This is the error status variable for the communications port 2. Represents the collective status of the following error flags. • _EIP2_MacAdrErr (Port2 MAC Address Error) • _EIP2_LanHwErr (Port2 Communications Controller Error) • _EIP2_EtnCfgErr (Port2 Basic Ethernet Setting Error) • _EIP2_IPAdrCfgErr (Port2 IP Address Setting Error) • _EIP2_IPAdrDupErr (Port2 IP Address Duplication Error) • _EIP2_BootpErr (Port2 BOOTP Server Error) • _EIP_DNSCfgErr (DNS Setting Error) • _EIP_DNSSrvErr (DNS Server Connection Error) • _EIP_IPRTblErr (IP Route Table Error) **Note** If a *Link OFF Detected* or *Built-in EtherNet/IP Error* occurs, it is recorded in the event log and then the corresponding bit turns ON. Refer to *Meanings of Error Status Bits* for the meanings of the error status bits. **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	WORD	16#0000 to 16#00F0	page 3-37

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_CipErr	CIP Communications Error	This is the error status variable for CIP communications. NX-series CPU Units: Represents the collective status of the following error flags. • _EIP1_IdentityErr (CIP Communications1 Identity Error) • _EIP1_TDLinkCfgErr (CIP Communications1 Tag Data Link Setting Error) • _EIP1_TDLinkOpnErr (CIP Communications1 Tag Data Link Connection Failed) • _EIP1_TDLinkErr (CIP Communications1 Tag Data Link Communications Error) • _EIP1_TagAdrErr (CIP Communications1 Tag Name Resolution Error) • _EIP1_MultiSwONErr (CIP Communications1 Multiple Switches ON Error) NJ-series CPU Units: Represents the collective status of the following error flags. • _EIP_IdentityErr (Identity Error) • _EIP_TDLinkCfgErr (Tag Data Link Setting Error) • _EIP_TDLinkOpnErr (Tag Data Link Connection Failed) • _EIP_TDLinkErr (Tag Data Link Communications Error) • _EIP_TagAdrErr (Tag Name Resolution Error) • _EIP_MultiSwOnErr (Multiple Switches ON Error) **Note** If a *Tag Name Resolution Error* occurs, it is recorded in the event log and this variable changes to TRUE. Refer to *Meanings of Error Status Bits* for the meanings of the error status bits.	WORD	16#0000 to 16#00F0	page 3-37

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP1_CipErr	CIP Communications1 Error	This is the error status variable for CIP communications 1. Represents the collective status of the following error flags. • _EIP1_IdentityErr (CIP Communications1 Identity Error) • _EIP1_TDLinkCfgErr (CIP Communications1 Tag Data Link Setting Error) • _EIP1_TDLinkOpnErr (CIP Communications1 Tag Data Link Connection Failed) • _EIP1_TDLinkErr (CIP Communications1 Tag Data Link Communications Error) • _EIP1_TagAdrErr (CIP Communications1 Tag Name Resolution Error) • _EIP1_MultiSwONErr (CIP Communications1 Multiple Switches ON Error) **Note** If a *Tag Name Resolution Error* occurs, it is recorded in the event log and this variable changes to TRUE. Refer to *Meanings of Error Status Bits* for the meanings of the error status bits. **Note** You can use this system-defined variable only for NX-series CPU Units.	WORD	16#0000 to 16#00F0	page 3-38
_EIP2_CipErr	CIP Communications2 Error	This is the error status variable for CIP communications 2. Represents the collective status of the following error flags. • _EIP2_IdentityErr (CIP Communications2 Identity Error) • _EIP2_TDLinkCfgErr (CIP Communications2 Tag Data Link Setting Error) • _EIP2_TDLinkOpnErr (CIP Communications2 Tag Data Link Connection Failed) • _EIP2_TDLinkErr (CIP Communications2 Tag Data Link Communications Error) • _EIP2_TagAdrErr (CIP Communications2 Tag Name Resolution Error) • _EIP2_MultiSwONErr (CIP Communications2 Multiple Switches ON Error) **Note** If a *Tag Name Resolution Error* occurs, it is recorded in the event log and this variable changes to TRUE. Refer to *Meanings of Error Status Bits* for the meanings of the error status bits. **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	WORD	16#0000 to 16#00F0	page 3-38

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_TcpAppErr	TCP Application Communications Error	This is the error status variable for TCP application communications. Represents the collective status of the following error flags. • _EIP_TcpAppCfgErr (TCP Application Setting Error) • _EIP_NTPSrvErr (NTP Server Connection Error) **Note** Refer to *Meanings of Error Status Bits* for the meanings of the error status bits.	WORD	16#0000 to 16#00F0	page 3-38
_EIP_MacAdrErr	MAC Address Error	NX-series CPU Units: Indicates that an error occurred when the MAC address was read on the communications port 1 at startup. TRUE: Error FALSE: Normal NJ-series CPU Units: Indicates that an error occurred when the MAC address was read at startup. TRUE: Error FALSE: Normal	BOOL	TRUE or FALSE	page 3-39
_EIP1_MacAdrErr	Port1 MAC Address Error	Indicates that an error occurred when the MAC address was read on the communications port 1 at startup. TRUE: Error FALSE: Normal **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-39
_EIP2_MacAdrErr	Port2 MAC Address Error	Indicates that an error occurred when the MAC address was read on the communications port 2 at startup. TRUE: Error FALSE: Normal **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-39
_EIP_LanHwErr	Communications Controller Error	NX-series CPU Units: Indicates that a communications controller failure occurred on the communications port 1. TRUE: Failure FALSE: Normal NJ-series CPU Units: Indicates that a communications controller failure occurred. TRUE: Failure FALSE: Normal	BOOL	TRUE or FALSE	page 3-39
_EIP1_LanHwErr	Port1 Communications Controller Error	Indicates that a communications controller failure occurred on the communications port 1. TRUE: Failure FALSE: Normal **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-40

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP2_LanHwErr	Port2 Communications Controller Error	Indicates that a communications controller failure occurred on the communications port 2. TRUE: Failure FALSE: Normal **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-40
_EIP_EtnCfgErr	Basic Ethernet Setting Error	NX-series CPU Units: Indicates that the Ethernet communications speed setting (Speed/Duplex) for the communications port 1 is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal NJ-series CPU Units: Indicates that the Ethernet communications speed setting (Speed/Duplex) is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal	BOOL	TRUE or FALSE	page 3-40
_EIP1_EtnCfgErr	Port1 Basic Ethernet Setting Error	Indicates that the Ethernet communications speed setting (Speed/Duplex) for the communications port 1 is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-40
_EIP2_EtnCfgErr	Port2 Basic Ethernet Setting Error	Indicates that the Ethernet communications speed setting (Speed/Duplex) for the communications port 2 is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-41
_EIP_IPAdrCfgErr	IP Address Setting Error	NX-series CPU Units: Indicates the IP address setting errors for the communications port 1. TRUE: • There is an illegal IP address setting. • A read operation failed. • The IP address obtained from the BOOTP server is inconsistent. FALSE: Normal NJ-series CPU Units: Indicates the IP address setting errors. TRUE: • There is an illegal IP address setting. • A read operation failed. • The IP address obtained from the BOOTP server is inconsistent. • The default gateway settings are not correct. FALSE: Normal	BOOL	TRUE or FALSE	page 3-41

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP1_IPAdrCfgErr	Port1 IP Address Setting Error	Indicates the IP address setting errors for the communications port 1. TRUE: • There is an illegal IP address setting. • A read operation failed. • The IP address obtained from the BOOTP server is inconsistent. FALSE: Normal **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-41
_EIP2_IPAdrCfgErr	Port2 IP Address Setting Error	Indicates the IP address setting errors for the communications port 2. TRUE: • There is an illegal IP address setting. • A read operation failed. • The IP address obtained from the BOOTP server is inconsistent. FALSE: Normal **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-42
_EIP_IPAdrDupErr	IP Address Duplication Error	NX-series CPU Units: Indicates that the same IP address is assigned to more than one node for the communications port 1. TRUE: Duplication occurred. FALSE: Other than the above. NJ-series CPU Units: Indicates that the same IP address is assigned to more than one node. TRUE: Duplication occurred. FALSE: Other than the above.	BOOL	TRUE or FALSE	page 3-42
_EIP1_IPAdrDupErr	Port1 IP Address Duplication Error	Indicates that the same IP address is assigned to more than one node for the communications port 1. TRUE: Duplication occurred. FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-42
_EIP2_IPAdrDupErr	Port2 IP Address Duplication Error	Indicates that the same IP address is assigned to more than one node for the communications port 2. TRUE: Duplication occurred. FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-42
_EIP_DNSCfgErr*1	DNS Setting Error	Indicates that the DNS or hosts settings are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal	BOOL	TRUE or FALSE	page 3-43

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_BootpErr	BOOTP Server Error	NX-series CPU Units: Indicates that a BOOTP server connection failure occurred on the communications port 1. TRUE: There was a failure to connect to the BOOTP server (timeout). FALSE: The BOOTP is not enabled, or BOOTP is enabled and an IP address was normally obtained from the BOOTP server. NJ-series CPU Units: Indicates that a BOOTP server connection failure occurred. TRUE: There was a failure to connect to the BOOTP server (timeout). FALSE: The BOOTP is not enabled, or BOOTP is enabled and an IP address was normally obtained from the BOOTP server.	BOOL	TRUE or FALSE	page 3-43
_EIP1_BootpErr	Port1 BOOTP Server Error	Indicates that a BOOTP server connection failure occurred on the communications port 1. TRUE: There was a failure to connect to the BOOTP server (timeout). FALSE: The BOOTP is not enabled, or BOOTP is enabled and an IP address was normally obtained from the BOOTP server. **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-43
_EIP2_BootpErr	Port2 BOOTP Server Error	Indicates that a BOOTP server connection failure occurred on the communications port 2. TRUE: There was a failure to connect to the BOOTP server (timeout). FALSE: The BOOTP is not enabled, or BOOTP is enabled and an IP address was normally obtained from the BOOTP server. **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-43
_EIP_IPRTblErr	IP Route Table Error	NX-series CPU Units: Indicates that the default gateway settings or IP router table settings are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal NJ-series CPU Units: Indicates that the IP router table or hosts settings are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal	BOOL	TRUE or FALSE	page 3-44

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_IdentityErr	Identity Error	NX-series CPU Units: Indicates that the identity information for CIP communications 1 (which you cannot overwrite) is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal NJ-series CPU Units: Indicates that the identity information (which you cannot overwrite) is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal	BOOL	TRUE or FALSE	page 3-44
_EIP1_IdentityErr	CIP Communications1 Identity Error	Indicates that the identity information for CIP communications 1 (which you cannot overwrite) is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-44
_EIP2_IdentityErr	CIP Communications2 Identity Error	Indicates that the identity information for CIP communications 2 (which you cannot overwrite) is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-44
_EIP_TDLinkCfgErr	Tag Data Link Setting Error	NX-series CPU Units: Indicates that the tag data link settings for CIP communications 1 are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal NJ-series CPU Units: Indicates that the tag data link settings are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal	BOOL	TRUE or FALSE	page 3-45
_EIP1_TDLinkCfgErr	CIP Communications1 Tag Data Link Setting Error	Indicates that the tag data link settings for CIP communications 1 are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-45
_EIP2_TDLinkCfgErr	CIP Communications2 Tag Data Link Setting Error	Indicates that the tag data link settings for CIP communications 2 are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-45

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_TDLinkOpnErr	Tag Data Link Connection Failed	NX-series CPU Units: Indicates that establishing a tag data link connection for CIP communications 1 failed. TRUE: Establishing a tag data link connection failed due to one of the following causes. • The information registered for a target node in the tag data link parameters is different from the actual node information. • There was no response from the remote node. FALSE: Other than the above. NJ-series CPU Units: Indicates that establishing a tag data link connection failed. TRUE: Establishing a tag data link connection failed due to one of the following causes. • The information registered for a target node in the tag data link parameters is different from the actual node information. • There was no response from the remote node. FALSE: Other than the above.	BOOL	TRUE or FALSE	page 3-46
_EIP1_TDLinkOpnErr	CIP Communications1 Tag Data Link Connection Failed	Indicates that establishing a tag data link connection for CIP communications 1 failed. TRUE: Establishing a tag data link connection failed due to one of the following causes. • The information registered for a target node in the tag data link parameters is different from the actual node information. • There was no response from the remote node. FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-46
_EIP2_TDLinkOpnErr	CIP Communications2 Tag Data Link Connection Failed	Indicates that establishing a tag data link connection for CIP communications 2 failed. TRUE: Establishing a tag data link connection failed due to one of the following causes. • The information registered for a target node in the tag data link parameters is different from the actual node information. • There was no response from the remote node. FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-46

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_TDLinkErr	Tag Data Link Communications Error	NX-series CPU Units: Indicates that a timeout occurred in a tag data link connection for CIP communications 1. TRUE: A timeout occurred. FALSE: Other than the above. NJ-series CPU Units: Indicates that a timeout occurred in a tag data link connection. TRUE: A timeout occurred. FALSE: Other than the above.	BOOL	TRUE or FALSE	page 3-47
_EIP1_TDLinkErr	CIP Communications1 Tag Data Link Communications Error	Indicates that a timeout occurred in a tag data link connection for CIP communications 1. TRUE: A timeout occurred. FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-47
_EIP2_TDLinkErr	CIP Communications2 Tag Data Link Communications Error	Indicates that a timeout occurred in a tag data link connection for CIP communications 2. TRUE: A timeout occurred. FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-47

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_TagAdrErr	Tag Name Resolution Error	NX-series CPU Units: Indicates that the tag resolution for CIP communications 1 failed (i.e., the address could not be identified from the tag name). TRUE: Tag resolution failed (i.e., the address could not be identified from the tag name). The following causes are possible. • The size of the network variable is different from the tag settings. • The I/O direction that is set in the tag data link settings does not agree with the I/O direction of the variable in the CPU Unit. • There is no network variable in the CPU Unit that corresponds to the tag setting. FALSE: Other than the above. NJ-series CPU Units: Indicates that tag name resolution failed (i.e., the address could not be identified from the tag name). TRUE: Tag resolution failed (i.e., the address could not be identified from the tag name). The following causes are possible. • The size of the network variable is different from the tag settings. • The I/O direction that is set in the tag data link settings does not agree with the I/O direction of the variable in the CPU Unit. • There is no network variable in the CPU Unit that corresponds to the tag setting. FALSE: Other than the above.	BOOL	TRUE or FALSE	page 3-48
_EIP1_TagAdrErr	CIP Communications1 Tag Name Resolution Error	Indicates that the tag resolution for CIP communications 1 failed (i.e., the address could not be identified from the tag name). TRUE: Tag resolution failed (i.e., the address could not be identified from the tag name). The following causes are possible. • The size of the network variable is different from the tag settings. • The I/O direction that is set in the tag data link settings does not agree with the I/O direction of the variable in the CPU Unit. • There is no network variable in the CPU Unit that corresponds to the tag setting. FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-48

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP2_TagAdrErr	CIP Communications2 Tag Name Resolution Error	Indicates that the tag resolution for CIP communications 2 failed (i.e., the address could not be identified from the tag name). TRUE: Tag resolution failed (i.e., the address could not be identified from the tag name). The following causes are possible. • The size of the network variable is different from the tag settings. • The I/O direction that is set in the tag data link settings does not agree with the I/O direction of the variable in the CPU Unit. • There is no network variable in the CPU Unit that corresponds to the tag setting. FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-49
_EIP_MultiSwONErr	Multiple Switches ON Error	NX-series CPU Units: Indicates that more than one switch turned ON at the same time in CIP communications 1. TRUE: More than one data link start/stop switch changed to TRUE at the same time. FALSE: Other than the above. NJ-series CPU Units: Indicates that more than one switch turned ON at the same time. TRUE: More than one data link start/stop switch changed to TRUE at the same time. FALSE: Other than the above.	BOOL	TRUE or FALSE	page 3-49
_EIP1_MultiSwONErr	CIP Communications1 Multiple Switches ON Error	Indicates that more than one switch turned ON at the same time in CIP communications 1. TRUE: More than one data link start/stop switch changed to TRUE at the same time. FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-49
_EIP2_MultiSwONErr	CIP Communications2 Multiple Switches ON Error	Indicates that more than one switch turned ON at the same time in CIP communications 2. TRUE: More than one data link start/stop switch changed to TRUE at the same time. FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-49

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_TcpAppCfgErr	TCP Application Setting Error	TRUE: At least one of the set values for a TCP application (FTP, NTP, SNMP) is incorrect. Or, a read operation failed. FALSE: Normal	BOOL	TRUE or FALSE	page 3-50
_EIP_NTPSrvErr	NTP Server Connection Error	TRUE: The NTP client failed to connect to the server (timeout). FALSE: NTP is not set. Or, NTP is set and the connection was successful.	BOOL	TRUE or FALSE	page 3-50
_EIP_DNSSrvErr	DNS Server Connection Error	TRUE: The DNS client failed to connect to the server (timeout). FALSE: DNS is not enabled. Or, DNS is enabled and the connection was successful.	BOOL	TRUE or FALSE	page 3-50

*1. With the NJ-series CPU Unit, this variable can be used with the unit version 1.11 or later.

Hierarchical Relationship of System-defined Variables Related to EtherNet/IP Errors in the NJ-series CPU Unit

The system-defined variables that are related to EtherNet/IP errors have the following hierarchical relationship. For example, if the value of any of the _EIP_PortErr, _EIP_CipErr, or _EIP_TcpAppErr variables in the second level is TRUE, then the _EIP_ErrSta variable in the first level also changes to TRUE. Therefore, you can check the values of system-defined variables in a higher level to see if an error has occurred for a variable in a lower level.

Level 1		Level 2		Level 3	
Variable	Name	Variable	Name	Variable	Name
_EIP_ErrSta	Built-in EtherNet/IP Error	_EIP_PortErr	Communications Port Error	_EIP_MacAdrErr	MAC Address Error
				_EIP_LanHwErr	Communications Controller Error
				_EIP_EtnCfgErr	Basic Ethernet Setting Error
				_EIP_IPAdrCfgErr	IP Address Setting Error
				_EIP_IPAdrDupErr	IP Address Duplication Error
				_EIP_BootpErr	BOOTP Server Error
				_EIP_DNSSrvErr	DNS Server Connection Error
				_EIP_IPRTblErr	IP Route Table Error
		_EIP_CipErr	CIP Communications Error	_EIP_IdentityErr	Identity Error
				_EIP_TDLinkCfgErr	Tag Data Link Setting Error
				_EIP_TDLinkOpnErr	Tag Data Link Connection Failed
				_EIP_TDLinkErr	Tag Data Link Communications Error
				_EIP_TagAdrErr	Tag Name Resolution Error
				_EIP_MultiSwONErr	Multiple Switches ON Error
		_EIP_TcpAppErr	TCP Application Communications Error	_EIP_TcpAppCfgErr	TCP Application Setting Error
				_EIP_NTPSrvErr	NTP Server Connection Error

Hierarchical Relationship of System-defined Variables Related to EtherNet/IP Errors in the NX-series CPU Unit

The system-defined variables that are related to EtherNet/IP errors have the following hierarchical relationship. For example, if the value of any of the _EIP1_PortErr, _EIP2_PortErr, EIP1_CipErr,

_EIP2_CipErr, and _EIP_TcpAppErr variables in the second level is TRUE, then the _EIP_ErrSta variable in the first level also changes to TRUE. Therefore, you can check the values of system-defined variables in a higher level to see if an error has occurred for a variable in a lower level.

Level 1		Level 2		Level 3	
Variable	Name	Variable	Name	Variable	Name
_EIP_ErrSta	Built-in EtherNet/IP Error	_EIP1_PortErr	Communications Port1 Error	_EIP1_MacAdrErr	Port1 MAC Address Error
				_EIP1_LanHwErr	Port1 Communications Controller Error
				_EIP1_EtnCfgErr	Port1 Basic Ethernet Setting Error
				_EIP1_IPAdrCfgErr	Port1 IP Address Setting Error
				_EIP1_IPAdrDupErr	Port1 IP Address Duplication Error
				_EIP1_BootpErr	Port1 BOOTP Server Error
				_EIP_DNSCfgErr	DNS Setting Error
				_EIP_DNSSrvErr	DNS Server Connection Error
				_EIP_IPRTblErr	IP Route Table Error
		_EIP2_PortErr	Communications Port2 Error	_EIP2_MacAdrErr	Port2 MAC Address Error
				_EIP2_LanHwErr	Port2 Communications Controller Error
				_EIP2_EtnCfgErr	Port2 Basic Ethernet Setting Error
				_EIP2_IPAdrCfgErr	Port2 IP Address Setting Error
				_EIP2_IPAdrDupErr	Port2 IP Address Duplication Error
				_EIP2_BootpErr	Port2 BOOTP Server Error
				_EIP_DNSCfgErr	DNS Setting Error
				_EIP_DNSSrvErr	DNS Server Connection Error
				_EIP_IPRTblErr	IP Route Table Error
		_EIP1_CipErr	CIP Communications1 Error	_EIP1_IdentityErr	CIP Communications1 Identity Error
				_EIP1_TDLinkCfgErr	CIP Communications1 Tag Data Link Setting Error
				_EIP1_TDLinkOpnErr	CIP Communications1 Tag Data Link Connection Failed
				_EIP1_TDLinkErr	CIP Communications1 Tag Data Link Communications Error
				_EIP1_TagAdrErr	CIP Communications1 Tag Name Resolution Error
				_EIP1_MultiSwONErr	CIP Communications1 Multiple Switches ON Error
		_EIP2_CipErr	CIP Communications2 Error	_EIP2_IdentityErr	CIP Communications2 Identity Error
				_EIP2_TDLinkCfgErr	CIP Communications2 Tag Data Link Setting Error
				_EIP2_TDLinkOpnErr	CIP Communications2 Tag Data Link Connection Failed
				_EIP2_TDLinkErr	CIP Communications2 Tag Data Link Communications Error
				_EIP2_TagAdrErr	CIP Communications2 Tag Name Resolution Error
				_EIP2_MultiSwONErr	CIP Communications2 Multiple Switches ON Error

Level 1		Level 2		Level 3	
Variable	Name	Variable	Name	Variable	Name
		_EIP_TcpAppErr	TCP Application Communications Error	_EIP_TcpAppCfgErr _EIP_NTPSrvErr	TCP Application Setting Error NTP Server Connection Error

Note 1. You can access the same values of the system-defined variables whose variable names with *_EIP1* and the system-defined variables whose variable names with *_EIP*. For example, you can access the same values of _EIP1_PortErr (Communications Port1 Error) and _EIP_PortErr (Communications Port Error).

Note 2. You can use the system-defined variables whose variable names with *_EIP2* only for the NX701 CPU Units and NX102 CPU Units.

● Meanings of Error Status Bits

The meanings of the individual bits in the following error status are the same.

- *_ErrSta* (Controller Error Status)
- *_PLC_ErrSta* (PLC Function Module Error Status)
- *_CJB_ErrSta* (I/O Bus Error Status)
- *_CJB_MstrErrSta* (I/O Bus Master Error Status)
- *_CJB_UnitErrSta* (I/O Bus Unit Error Status)
- *_NXB_ErrSta* (NX Bus Function Module Error Status)
- *_NXB_MstrErrSta* (NX Bus Function Module Master Error Status)
- *_NXB_UnitErrStaTbl* (NX Bus Function Module Unit Error Status)
- *_MC_ErrSta* (MC Error Status)
- *_MC_ComErrSta* (MC Common Error Status)
- *_MC_AX_ErrSta* (Axis Error Status)
- *_MC_GRP_ErrSta* (Axes Group Error Status)
- *_EC_ErrSta* (Built-in EtherCAT Error)
- *_EC_PortErr* (Communications Port Error)
- *_EC_MstrErr* (Master Error)
- *_EC_SlavErr* (Slave Error)
- *_EC_SlavErrTbl* (Slave Error Table)
- *_EIP_ErrSta* (Built-in EtherNet/IP Error)
- *_EIP_PortErr* (Communications Port Error), *_EIP1_PortErr* (Communications Port1 Error), *_EIP2_PortErr* (Communications Port2 Error)
- *_EIP_CipErr* (CIP Communications Error), *_EIP1_CipErr* (CIP Communications1 Error), *_EIP2_CipErr* (CIP Communications2 Error)
- *_EIP_TcpAppErr* (TCP Application Communications Error)

The meanings of the bits are shown in the following table. However, do not use the following variables in the user program: *_ErrSta* (Controller Error Status), *_CJB_ErrSta* (I/O Bus Error Status), *_CJB_MstrErrSta* (I/O Bus Master Error Status), *_CJB_UnitErrSta* (I/O Bus Unit Error Status), *_NXB_ErrSta* (NX Bus Function Module Error Status), *_NXB_MstrErrSta* (NX Bus Function Module Master Error Status), and *_NXB_UnitErrStaTbl* (NX Bus Function Module Unit Error Status). There may be a delay in updating them and concurrency problems in relation to the error status of the function module. Use these variables only to access status through communications from an external device.

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WORD			-	-	-	-	-	-					-	-	-	-

Bit	Description
15	Master-detected error: This bit indicates whether the master detected a Controller error in the Unit/slave for the error status of the Controller error. TRUE: The master detected a Controller error. FALSE: The master has not detected a Controller error. (Valid for _CJB_UnitErrSta.)
14	Collective slave error status: This bit indicates if a Controller error is detected for levels (e.g., a Unit, slave, axis, or axes group) that are lower than the event source (i.e., a function module). TRUE: A Controller error has occurred at a lower level. FALSE: A Controller error has not occurred at a lower level. (Valid for _CJB_ErrSta, _MC_ErrSta, and _EC_ErrSta.)
13 to 8	Reserved.
7	This bit indicates whether a major fault level Controller error has occurred. TRUE: A major fault level Controller error has occurred. FALSE: A major fault level Controller error has not occurred.
6	This bit indicates whether a partial fault level Controller error has occurred. TRUE: A partial fault level Controller error has occurred. FALSE: A partial fault level Controller error has not occurred.
5	This bit indicates whether a minor fault level Controller error has occurred. TRUE: A minor fault level Controller error has occurred. FALSE: A minor fault level Controller error has not occurred.
4	This bit indicates whether an observation level Controller error has occurred. TRUE: An observation level Controller error has occurred. FALSE: An observation level Controller error has not occurred.
3 to 0	Reserved.

Note Bits 14 and 15 are never TRUE for the built-in EtherNet/IP port.

● Functional Classification: EtherNet/IP Communications Status

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_EtnOnlineSta	Online	NX-series CPU Units: Indicates that the built-in EtherNet/IP port's communications can be used via the communications port 1 (that is, the link is ON, IP address is defined, and there are no errors.) TRUE: The built-in EtherNet/IP port's communications can be used. FALSE: The built-in EtherNet/IP port's communications is disabled due to an error in initial processing, restart processing, or link OFF status. NJ-series CPU Units: Indicates that the built-in EtherNet/IP port's communications can be used via the communications port (that is, the link is ON and IP address is defined, and there are no errors.) TRUE: The built-in EtherNet/IP port's communications can be used. FALSE: The built-in EtherNet/IP port's communications is disabled due to an error in initial processing, restart processing, or link OFF status.	BOOL	TRUE or FALSE	page 3-50

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP1_EtnOnlineSta	Port1 Online	Indicates that the built-in EtherNet/IP port's communications can be used via the communications port 1 (that is, the link is ON, IP address is defined, and there are no errors.) TRUE: The built-in EtherNet/IP port's communications can be used. FALSE: The built-in EtherNet/IP port's communications is disabled due to an error in initial processing, restart processing, or link OFF status. **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-51
_EIP2_EtnOnlineSta	Port2 Online	Indicates that the built-in EtherNet/IP port's communications can be used via the communications port 2 (that is, the link is ON, IP address is defined, and there are no errors.) TRUE: The built-in EtherNet/IP port's communications can be used. FALSE: The built-in EtherNet/IP port's communications is disabled due to an error in initial processing, restart processing, or link OFF status. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-51
_EIP_TDLinkRunSta	Tag Data Link Communications Status	NX-series CPU Units: Indicates that at least one connection is in normal operation in CIP communications 1. TRUE: Normal operation FALSE: Other than the above. NJ-series CPU Units: Indicates that at least one connection is in normal operation. TRUE: Normal operation FALSE: Other than the above.	BOOL	TRUE or FALSE	page 3-51
_EIP1_TDLinkRunSta	CIP Communications1 Tag Data Link Communications Status	Indicates that at least one connection is in normal operation in CIP communications 1. TRUE: Normal operation FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-51
_EIP2_TDLinkRunSta	CIP Communications2 Tag Data Link Communications Status	Indicates that at least one connection is in normal operation in CIP communications 2. TRUE: Normal operation FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-52

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_TDLinkAllRunSta	All Tag Data Link Communications Status	NX-series CPU Units: Indicates that all tag data links are communicating in CIP communications 1. TRUE: Tag data links are communicating in all connections as the originator. FALSE: An error occurred in at least one connection. NJ-series CPU Units: Indicates that all tag data links are communicating. TRUE: Tag data links are communicating in all connections as the originator. FALSE: An error occurred in at least one connection.	BOOL	TRUE or FALSE	page 3-52
_EIP1_TDLinkAllRunSta	CIP Communications1 All Tag Data Link Communications Status	Indicates that all tag data links are communicating in CIP communications 1. TRUE: Tag data links are communicating in all connections as the originator. FALSE: An error occurred in at least one connection. **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-52
_EIP2_TDLinkAllRunSta	CIP Communications2 All Tag Data Link Communications Status	Indicates that all tag data links are communicating in CIP communications 2. TRUE: Tag data links are communicating in all connections as the originator. FALSE: An error occurred in at least one connection. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-52
_EIP_RegTargetSta[255]	Registered Target Node Information	NX-series CPU Units: Gives a list of nodes for which built-in EtherNet/IP connections are registered for CIP communications 1. This variable is valid only when the built-in EtherNet/IP port is the originator. Array[x] is TRUE: The connection to the node with a target node ID of x is registered. Array[x] is FALSE: The connection to the node with a target node ID of x is not registered. NJ-series CPU Units: Gives a list of nodes for which built-in EtherNet/IP connections are registered. This variable is valid only when the built-in EtherNet/IP port is the originator. Array[x] is TRUE: The connection to the node with a target node ID of x is registered. Array[x] is FALSE: The connection to the node with a target node ID of x is not registered.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-53

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP1_RegTarget-Sta[255]	CIP Communications1 Registered Target Node Information	Gives a list of nodes for which built-in EtherNet/IP connections are registered for CIP communications 1. This variable is valid only when the built-in EtherNet/IP port is the originator. Array[x] is TRUE: The connection to the node with a target node ID of x is registered. Array[x] is FALSE: The connection to the node with a target node ID of x is not registered. **Note** You can use this system-defined variable only for NX-series CPU Units.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-53
_EIP2_RegTarget-Sta[255]	CIP Communications2 Registered Target Node Information	Gives a list of nodes for which built-in EtherNet/IP connections are registered for CIP communications 2. This variable is valid only when the built-in EtherNet/IP port is the originator. Array[x] is TRUE: The connection to the node with a target node ID of x is registered. Array[x] is FALSE: The connection to the node with a target node ID of x is not registered. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-53
_EIP_EstbTarget-Sta[255]	Normal Target Node Information	NX-series CPU Units: Gives a list of nodes that have normally established built-in EtherNet/IP connections for CIP communications 1. Array[x] is TRUE: The connection to the node with a target node ID of x was established normally. Array[x] is FALSE: The connection to the node with a target node ID of x was not established, or an error occurred. NJ-series CPU Units: Gives a list of nodes that have normally established built-in EtherNet/IP connections. Array[x] is TRUE: The connection to the node with a target node ID of x was established normally. Array[x] is FALSE: The connection to the node with a target node ID of x was not established, or an error occurred.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-54

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP1_EstbTarget-Sta[255]	CIP Communica-tions1 Normal Target Node Information	Gives a list of nodes that have normally established built-in EtherNet/IP connec-tions for CIP communications 1. Array[x] is TRUE: The connection to the node with a target node ID of x was established normally. Array[x] is FALSE: The connection to the node with a target node ID of x was not established, or an error occurred. **Note** You can use this system-de-fined variable only for NX-ser-ies CPU Units.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-54
_EIP2_EstbTarget-Sta[255]	CIP Communica-tions2 Normal Target Node Information	Gives a list of nodes that have normally established built-in EtherNet/IP connec-tions for CIP communications 2. Gives a list of nodes that have normally established EtherNet/IP connections for CIP communications 2. Array[x] is TRUE: The connection to the node with a target node ID of x was established normally. Array[x] is FALSE: The connection to the node with a target node ID of x was not established, or an error occurred. **Note** You can use this system-de-fined variable only for the NX701 CPU Units and NX102 CPU Units.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-54

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_TargetPLCModeSta[255]	Target PLC Operating Mode	NX-series CPU Units: Shows the operating status of the target node Controllers that are connected for CIP communications 1, with the built-in EtherNet/IP port as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. If the corresponding Normal Target Node Information is FALSE, it indicates the previous operating status. Array[x] is TRUE: This is the operating state of the target Controller with a node address of x. Array[x] is FALSE: Other than the above. NJ-series CPU Units: Shows the operating status of the target node Controllers that are connected with the built-in EtherNet/IP port as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. If the corresponding Normal Target Node Information is FALSE, it indicates the previous operating status. Array[x] is TRUE: This is the operating state of the target Controller with a node address of x. Array[x] is FALSE: Other than the above.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-55
_EIP1_TargetPLCModeSta[255]	CIP Communications1 Target PLC Operating Mode	Shows the operating status of the target node Controllers that are connected for CIP communications 1, with the built-in EtherNet/IP port as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. If the corresponding Normal Target Node Information is FALSE, it indicates the previous operating status. Array[x] is TRUE: This is the operating state of the target Controller with a node address of x. Array[x] is FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-55

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP2_TargetPLC-ModeSta[255]	CIP Communications2 Target PLC Operating Mode	Shows the operating status of the target node Controllers that are connected for CIP communications 2, with the built-in EtherNet/IP port as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. If the corresponding Normal Target Node Information is FALSE, it indicates the previous operating status. Array[x] is TRUE: This is the operating state of the target Controller with a node address of x. Array[x] is FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-55
_EIP_TargetPLC-CErr[255]	Target PLC Error Information	NX-series CPU Units: Shows the error status (logical OR of fatal and non-fatal errors) of the target node Controllers that are connected for CIP communications 1, with the built-in EtherNet/IP ports as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. The immediately preceding value is retained if this variable is FALSE. Array[x] is TRUE: A fatal or non-fatal error occurred in the target Controller with a target node ID of x. Array[x] is FALSE: Other than the above. NJ-series CPU Units: Shows the error status (logical OR of fatal and non-fatal errors) of the target node Controllers that are connected with the built-in EtherNet/IP ports as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. The immediately preceding value is retained if this variable is FALSE. Array[x] is TRUE: A fatal or non-fatal error occurred in the target Controller with a target node ID of x. Array[x] is FALSE: Other than the above.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-56

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP1_TargetPLCErr[255]	CIP Communications1 Target PLC Error Information	Shows the error status (logical OR of fatal and non-fatal errors) of the target node Controllers that are connected for CIP communications 1, with the built-in EtherNet/IP ports as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. The immediately preceding value is retained if this variable is FALSE. Array[x] is TRUE: A fatal or non-fatal error occurred in the target Controller with a target node ID of x. Array[x] is FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-56
_EIP2_TargetPLCErr[255]	CIP Communications2 Target PLC Error Information	Shows the error status (logical OR of fatal and non-fatal errors) of the target node Controllers that are connected for CIP communications 2, with the built-in EtherNet/IP ports as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. The immediately preceding value is retained if this variable is FALSE. Array[x] is TRUE: A fatal or non-fatal error occurred in the target Controller with a target node ID of x. Array[x] is FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-56

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_TargetNodeErr[255]	Target Node Error Information	NX-series CPU Units: Indicates that the connection for the Registered Target Node Information for CIP communications 1 was not established or that an error occurred in the target Controller. The array elements are valid only when the Registered Target Node Information is TRUE. Array[x] is TRUE: A connection was not normally established with the target node for a target node ID of x (the Registered Target Node Information is TRUE and the Normal Target Node Information is FALSE), or a connection was established with the target node but an error occurred in the target Controller. Array[x] is FALSE: The target node is not registered for a target node ID of x (the Registered Target Node Information is FALSE), or a connection was normally established with the target node (the Registered Target Node Information is TRUE and the Normal Target Node Information is TRUE). An error occurred in the target Controller (the Target PLC Error Information is TRUE). NJ-series CPU Units: Indicates that the connection for the Registered Target Node Information was not established or that an error occurred in the target Controller. The array elements are valid only when the Registered Target Node Information is TRUE. Array[x] is TRUE: A connection was not normally established with the target node for a target node ID of x (the Registered Target Node Information is TRUE and the Normal Target Node Information is FALSE), or a connection was established with the target node but an error occurred in the target Controller. Array[x] is FALSE: The target node is not registered for a target node ID of x (the Registered Target Node Information is FALSE), or a connection was normally established with the target node (the Registered Target Node Information is TRUE and the Normal Target Node Information is TRUE). An error occurred in the target Controller (the Target PLC Error Information is TRUE).	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-57

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP1_TargetNodeErr[255]	CIP Communications1 Target Node Error Information	Indicates that the connection for the Registered Target Node Information for CIP communications 1 was not established or that an error occurred in the target Controller. The array elements are valid only when the Registered Target Node Information is TRUE. Array[x] is TRUE: A connection was not normally established with the target node for a target node ID of x (the Registered Target Node Information is TRUE and the Normal Target Node Information is FALSE), or a connection was established with the target node but an error occurred in the target Controller. Array[x] is FALSE: The target node is not registered for a target node ID of x (the Registered Target Node Information is FALSE), or a connection was normally established with the target node (the Registered Target Node Information is TRUE and the Normal Target Node Information is TRUE). An error occurred in the target Controller (the Target PLC Error Information is TRUE). **Note** You can use this system-defined variable only for NX-series CPU Units.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-57

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP2_TargetNodeErr[255]	CIP Communications2 Target Node Error Information	Indicates that the connection for the Registered Target Node Information for CIP communications 2 was not established or that an error occurred in the target Controller. The array elements are valid only when the Registered Target Node Information is TRUE. Array[x] is TRUE: A connection was not normally established with the target node for a target node ID of x (the Registered Target Node Information is TRUE and the Normal Target Node Information is FALSE), or a connection was established with the target node but an error occurred in the target Controller. Array[x] is FALSE: The target node is not registered for a target node ID of x (the Registered Target Node Information is FALSE), or a connection was normally established with the target node (the Registered Target Node Information is TRUE and the Normal Target Node Information is TRUE). An error occurred in the target Controller (the Target PLC Error Information is TRUE). **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.	ARRAY [0..255] OF BOOL	TRUE or FALSE	page 3-58
_EIP_NTPResult	NTP Operation Information	Use the GetNTPStatus instruction to read the NTP operation information from the user program. Direct access is not possible.	_sNTP_RESULT		page 3-58
.ExecTime	NTP Last Operation Time	Gives the last time that NTP processing ended normally. The time that was obtained from the NTP server is stored when the time is obtained normally. The time is not stored if it is not obtained from the NTP server normally. **Note** Do not use this variable in the user program. There may be a delay in updating it. Use this variable only to access status through communications from an external device.	DATE_AND_TIME	Depends on data type.	page 3-58
.ExecNormal	NTP Operation Result	TRUE: Indicates an NTP normal end. FALSE: Indicates that NTP operation ended in an error or has not been executed even once. **Note** Do not use this variable in the user program. There may be a delay in updating it. Use this variable only to access status through communications from an external device.	BOOL	TRUE or FALSE	page 3-58

Precautions for Correct Use

Communications Status with Target Node

The communications status with the target node of an NJ/NX-series Controller is shown by the combination of the values of four system-defined variables.

- _EIP_RegTargetSta (Registered Target Node Information)
- _EIP_EstbTargetSta (Normal Target Node Information)
- _EIP_TargetPLCErr (Target PLC Error Information)
- _EIP_TargetNodeErr (Target Node Error Information)

Value of *_EIP_RegTargetSta*	Value of *_EIP_EstbTargetSta*	Value of *_EIP_TargetPLCErr*	Value of *_EIP_TargetNodeErr*	Communications status with target node
TRUE	TRUE	FALSE	FALSE	A connection with the target node was established normally and there is no error in the target PLC.
		TRUE	TRUE	A connection with the target node was established but there is an error in the target PLC.
	FALSE	---	TRUE	A connection with the target node was not established normally.
FALSE	---	---	---	The information is not valid because the target node is not registered.

For the NX-series Controller, the communications status of CIP communications 1 and CIP communications 2 is shown by the combination of the values of four system-defined variables in the same way as shown in the above table.

- CIP Communications 1
  - _EIP1_RegTargetSta (CIP Communications1 Registered Target Node Information)
  - _EIP1_EstbTargetSta (CIP Communications1 Normal Target Node Information)
  - _EIP1_TargetPLCErr (CIP Communications1 Target PLC Error Information)
  - _EIP1_TargetNodeErr (CIP Communications1 Target Node Error Information)
- CIP Communications 2
  - _EIP2_RegTargetSta (CIP Communications2 Registered Target Node Information)
  - _EIP2_EstbTargetSta (CIP Communications2 Normal Target Node Information)
  - _EIP2_TargetPLCErr (CIP Communications2 Target PLC Error Information)
  - _EIP2_TargetNodeErr (CIP Communications2 Target Node Error Information)

● Functional Classification: EtherNet/IP Communications Switches

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP_TDLink-StartCmd	Tag Data Link Communications Start Switch	NX-series CPU Units: Change this variable to TRUE to start tag data links for CIP communications 1. It automatically changes back to FALSE after tag data link operation starts. NJ-series CPU Units: Change this variable to TRUE to start tag data links. It automatically changes back to FALSE after tag data link operation starts. **Note** Do not force this switch to change to FALSE from the user program or from the Sysmac Studio. It changes to FALSE automatically.	BOOL	TRUE or FALSE	page 3-59
_EIP1_TDLink-StartCmd	CIP Communications1 Tag Data Link Communications Start Switch	Change this variable to TRUE to start tag data links for CIP communications 1. It automatically changes back to FALSE after tag data link operation starts. **Note** Do not force this switch to change to FALSE from the user program or from the Sysmac Studio. It changes to FALSE automatically. **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-59
_EIP2_TDLink-StartCmd	CIP Communications2 Tag Data Link Communications Start Switch	Change this variable to TRUE to start tag data links for CIP communications 2. It automatically changes back to FALSE after tag data link operation starts. **Note** Do not force this switch to change to FALSE from the user program or from the Sysmac Studio. It changes to FALSE automatically. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-59
_EIP_TDLink-StopCmd	Tag Data Link Communications Stop Switch	NX-series CPU Units: Change this variable to TRUE to stop tag data links for CIP communications 1. It automatically changes back to FALSE after tag data link operation stops. NJ-series CPU Units: Change this variable to TRUE to stop tag data links. It automatically changes back to FALSE after tag data link operation stops. **Note** Do not force this switch to change to FALSE from the user program or from the Sysmac Studio. It changes to FALSE automatically.	BOOL	TRUE or FALSE	page 3-59

Variable name	Meaning	Function	Data type	Range of values	Reference
_EIP1_TDLink-StopCmd	CIP Communications1 Tag Data Link Communications Stop Switch	Change this variable to TRUE to stop tag data links for CIP communications 1. It automatically changes back to FALSE after tag data link operation stops. **Note** Do not force this switch to change to FALSE from the user program or from the Sysmac Studio. It changes to FALSE automatically. **Note** You can use this system-defined variable only for NX-series CPU Units.	BOOL	TRUE or FALSE	page 3-60
_EIP2_TDLink-StopCmd	CIP Communications2 Tag Data Link Communications Stop Switch	Change this variable to TRUE to stop tag data links for CIP communications 2. It automatically changes back to FALSE after tag data link operation stops. **Note** Do not force this switch to change to FALSE from the user program or from the Sysmac Studio. It changes to FALSE automatically. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.	BOOL	TRUE or FALSE	page 3-60

3-3 Specifications for Individual System-defined Variables

The specifications for each system-defined variable are given as described below.

Variable name	This is the system-defined variable name. The prefix gives the category name.			**Members**	The member names are given for structure variables.
Meaning	This is the meaning of the variable.			**Global/local**	Global: Global variable, Local: Local variable
Function	The function of the variable is described.				
Data type	The data type of the variable is given.			**Range of values**	The range of values that the variable can take is given.
R/W access	R: Read only, RW: Read/write	**Retained**	The Retain attribute of the variable is given.	**Network Publish**	The Network Publish attribute of the variable is given.
Usage in user program	Whether you can use the variable directly in the user program is specified.	**Related instructions**	The instructions that are related to the variable are given. If you cannot use the variable directly in the user program, the instructions that access the variable are given.		

● Functional Classification: EtherNet/IP Communications Errors

Variable name	_EIP_ErrSta				
Meaning	Built-in EtherNet/IP Error			**Global/local**	Global
Function	This is the error status variable for the built-in EtherNet/IP port. NX-series CPU Units: Represents the collective status of the following error flags. • _EIP1_PortErr (Communications Port1 Error) • _EIP2_PortErr (Communications Port2 Error) • _EIP1_CipErr (CIP Communications1 Error) • _EIP2_CipErr (CIP Communications2 Error) • _EIP_TcpAppErr (TCP Application Communications Error) NJ-series CPU Units: Represents the collective status of the following error flags. • _EIP_PortErr (Communications Port Error) • _EIP_CipErr (CIP Communications Error) • _EIP_TcpAppErr (TCP Application Communications Error) **Note** Refer to *Meanings of Error Status Bits* on page 3-20 for the meanings of the error status bits.				
Data type	WORD			**Range of values**	16#0000 to 16#00F0
R/W access	R	**Retained**	Not retained.	**Network Publish**	Published.
Usage in user program	Possible.	**Related instructions**	You can access this variable from the user program with the following instruction. • GetEIPError		

Variable name	_EIP_PortErr				
Meaning	Communications Port Error			Global/local	Global
Function	This is the error status variable for the communications port. NX-series CPU Units: Represents the collective status of the following error flags. • _EIP1_MacAdrErr (Port1 MAC Address Error) • _EIP1_LanHwErr (Port1 Communications Controller Error) • _EIP1_EtnCfgErr (Port1 Basic Ethernet Setting Error) • _EIP1_IPAdrCfgErr (Port1 IP Address Setting Error) • _EIP1_IPAdrDupErr (Port1 IP Address Duplication Error) • _EIP1_BootpErr (Port1 BOOTP Server Error) • _EIP_DNSCfgErr (DNS Setting Error) • _EIP_DNSSrvErr (DNS Server Connection Error) • _EIP_IPRTblErr (IP Route Table Error) NJ-series CPU Units: Represents the collective status of the following error flags. • _EIP_MacAdrErr (MAC Address Error) • _EIP_LanHwErr (Communications Controller Error) • _EIP_EtnCfgErr (Basic Ethernet Setting Error) • _EIP_IPAdrCfgErr (IP Address Setting Error) • _EIP_IPAdrDupErr (IP Address Duplication Error) • _EIP_BootpErr (BOOTP Server Error) • _EIP_DNSSrvErr (DNS Server Connection Error) • _EIP_IPRTblErr (IP Route Table Error) **Note** If a Link OFF Detected or Built-in EtherNet/IP Error occurs, it is recorded in the event log and then the corresponding bit turns ON. Refer to *Meanings of Error Status Bits* on page 3-20 for the meanings of the error status bits.				
Data type	WORD			Range of values	16#0000 to 16#00F0
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	You can access this variable from the user program with the following instruction. • GetEIPError		

Variable name	_EIP1_PortErr				
Meaning	Communications Port1 Error			Global/local	Global
Function	This is the error status variable for the communications port 1. It represents the collective status of the following error flags. • _EIP1_MacAdrErr (Port1 MAC Address Error) • _EIP1_LanHwErr (Port1 Communications Controller Error) • _EIP1_EtnCfgErr (Port1 Basic Ethernet Setting Error) • _EIP1_IPAdrCfgErr (Port1 IP Address Setting Error) • _EIP1_IPAdrDupErr (Port1 IP Address Duplication Error) • _EIP1_BootpErr (Port1 BOOTP Server Error) • _EIP_DNSCfgErr (DNS Setting Error) • _EIP_DNSSrvErr (DNS Server Connection Error) • _EIP_IPRTblErr (IP Route Table Error) **Note** If a Link OFF Detected or Built-in EtherNet/IP Error occurs, it is recorded in the event log and then the corresponding bit turns ON. Refer to *Meanings of Error Status Bits* on page 3-20 for the meanings of the error status bits. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	WORD			Range of values	16#0000 to 16#00F0
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	You can access this variable from the user program with the following instruction. • GetEIPError		

Variable name	_EIP2_PortErr				
Meaning	Communications Port2 Error			Global/local	Global
Function	This is the error status variable for the communications port 2. It represents the collective status of the following error flags. • _EIP2_MacAdrErr (Port2 MAC Address Error) • _EIP2_LanHwErr (Port2 Communications Controller Error) • _EIP2_EtnCfgErr (Port2 Basic Ethernet Setting Error) • _EIP2_IPAdrCfgErr (Port2 IP Address Setting Error) • _EIP2_IPAdrDupErr (Port2 IP Address Duplication Error) • _EIP2_BootpErr (Port2 BOOTP Server Error) • _EIP_DNSCfgErr (DNS Setting Error) • _EIP_DNSSrvErr (DNS Server Connection Error) • _EIP_IPRTblErr (IP Route Table Error) **Note** If a Link OFF Detected or Built-in EtherNet/IP Error occurs, it is recorded in the event log and then the corresponding bit turns ON. Refer to *Meanings of Error Status Bits* on page 3-20 for the meanings of the error status bits. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	WORD			Range of values	16#0000 to 16#00F0
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	You can access this variable from the user program with the following instruction. • GetEIPError		

Variable name	_EIP_CipErr				
Meaning	CIP Communications Error			Global/local	Global
Function	This is the error status variable for CIP communications. NX-series CPU Units: Represents the collective status of the following error flags. • _EIP1_IdentityErr (CIP Communications1 Identity Error) • _EIP1_TDLinkCfgErr (CIP Communications1 Tag Data Link Setting Error) • _EIP1_TDLinkOpnErr (CIP Communications1 Tag Data Link Connection Failed) • _EIP1_TDLinkErr (CIP Communications1 Tag Data Link Communications Error) • _EIP1_TagAdrErr (CIP Communications1 Tag Name Resolution Error) • _EIP1_MultiSwONErr (CIP Communications1 Multiple Switches ON Error) NJ-series CPU Units: Represents the collective status of the following error flags. • _EIP_IdentityErr (Identity Error) • _EIP_TDLinkCfgErr (Tag Data Link Setting Error) • _EIP_TDLinkOpnErr (Tag Data Link Connection Failed) • _EIP_TDLinkErr (Tag Data Link Communications Error) • _EIP_TagAdrErr (Tag Name Resolution Error) • _EIP_MultiSwOnErr (Multiple Switches ON Error) **Note** If a Tag Name Resolution Error occurs, it is recorded in the event log and this variable changes to TRUE. Refer to *Meanings of Error Status Bits* on page 3-20 for the meanings of the error status bits.				
Data type	WORD			Range of values	16#0000 to 16#00F0
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	You can access this variable from the user program with the following instruction. • GetEIPError		

Variable name	_EIP1_CipErr				
Meaning	CIP Communications1 Error			Global/local	Global
Function	This is the error status variable for CIP communications 1. It represents the collective status of the following error flags. • _EIP1_IdentityErr (CIP Communications1 Identity Error) • _EIP1_TDLinkCfgErr (CIP Communications1 Tag Data Link Setting Error) • _EIP1_TDLinkOpnErr (CIP Communications1 Tag Data Link Connection Failed) • _EIP1_TDLinkErr (CIP Communications1 Tag Data Link Communications Error) • _EIP1_TagAdrErr (CIP Communications1 Tag Name Resolution Error) • _EIP1_MultiSwONErr (CIP Communications1 Multiple Switches ON Error) **Note** If a Tag Name Resolution Error occurs, it is recorded in the event log and this variable changes to TRUE. Refer to *Meanings of Error Status Bits* on page 3-20 for the meanings of the error status bits. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	WORD			Range of values	16#0000 to 16#00F0
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	You can access this variable from the user program with the following instruction. • GetEIPError		

Variable name	_EIP2_CipErr				
Meaning	CIP Communications2 Error			Global/local	Global
Function	This is the error status variable for CIP communications 2. It represents the collective status of the following error flags. • _EIP2_IdentityErr (CIP Communications2 Identity Error) • _EIP2_TDLinkCfgErr (CIP Communications2 Tag Data Link Setting Error) • _EIP2_TDLinkOpnErr (CIP Communications2 Tag Data Link Connection Failed) • _EIP2_TDLinkErr (CIP Communications2 Tag Data Link Communications Error) • _EIP2_TagAdrErr (CIP Communications2 Tag Name Resolution Error) • _EIP2_MultiSwONErr (CIP Communications2 Multiple Switches ON Error) **Note** If a Tag Name Resolution Error occurs, it is recorded in the event log and this variable changes to TRUE. Refer to *Meanings of Error Status Bits* on page 3-20 for the meanings of the error status bits. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	WORD			Range of values	16#0000 to 16#00F0
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	You can access this variable from the user program with the following instruction. • GetEIPError		

Variable name	_EIP_TcpAppErr				
Meaning	TCP Application Communications Error			Global/local	Global
Function	This is the error status variable for TCP application communications. It represents the collective status of the following error flags. • _EIP_TcpAppCfgErr (TCP Application Setting Error) • _EIP_NTPSrvErr (NTP Server Connection Error) **Note** Refer to *Meanings of Error Status Bits* on page 3-20 for the meanings of the error status bits.				
Data type	WORD			Range of values	16#0000 to 16#00F0
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	You can access this variable from the user program with the following instruction. • GetEIPError		

Variable name	_EIP_MacAdrErr				
Meaning	MAC Address Error			Global/local	Global
Function	NX-series CPU Units: Indicates that an error occurred when the MAC address was read on the communications port 1 at startup. TRUE: Error FALSE: Normal NJ-series CPU Units: Indicates that an error occurred when the MAC address was read at startup. TRUE: Error FALSE: Normal				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_MacAdrErr				
Meaning	Port1 MAC Address Error			Global/local	Global
Function	Indicates that an error occurred when the MAC address was read on the communications port 1 at startup. TRUE: Error FALSE: Normal **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_MacAdrErr				
Meaning	Port2 MAC Address Error			Global/local	Global
Function	Indicates that an error occurred when the MAC address was read on the communications port 2 at startup. TRUE: Error FALSE: Normal **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_LanHwErr				
Meaning	Communications Controller Error			Global/local	Global
Function	NX-series CPU Units: Indicates that a communications controller failure occurred on the communications port 1. TRUE: Failure FALSE: Normal NJ-series CPU Units: Indicates that a communications controller failure occurred. TRUE: Failure FALSE: Normal				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_LanHwErr				
Meaning	Port1 Communications Controller Error			Global/local	Global
Function	Indicates that a communications controller failure occurred on the communications port 1. TRUE: Failure FALSE: Normal **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TTRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_LanHwErr				
Meaning	Port2 Communications Controller Error			Global/local	Global
Function	Indicates that a communications controller failure occurred on the communications port 2. TRUE: Failure FALSE: Normal **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_EtnCfgErr				
Meaning	Basic Ethernet Setting Error			Global/local	Global
Function	NX-series CPU Units: Indicates that the Ethernet communications speed setting (Speed/Duplex) for the communications port 1 is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal NJ-series CPU Units: Indicates that the Ethernet communications speed setting (Speed/Duplex) is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_EtnCfgErr				
Meaning	Port1 Basic Ethernet Setting Error			Global/local	Global
Function	Indicates that the Ethernet communications speed setting (Speed/Duplex) for the communications port 1 is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_EtnCfgErr				
Meaning	Port2 Basic Ethernet Setting Error			Global/local	Global
Function	Indicates that the Ethernet communications speed setting (Speed/Duplex) for the communications port 2 is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_IPAdrCfgErr				
Meaning	IP Address Setting Error			Global/local	Global
Function	NX-series CPU Units: Indicates the IP address setting errors for the communications port 1. TRUE: • There is an illegal IP address setting. • A read operation failed. • The IP address obtained from the BOOTP server is inconsistent. FALSE: Normal NJ-series CPU Units: Indicates the IP address setting errors. TRUE: • There is an illegal IP address setting. • A read operation failed. • The IP address obtained from the BOOTP server is inconsistent. • The default gateway settings are not correct. FALSE: Normal				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_IPAdrCfgErr				
Meaning	Port1 IP Address Setting Error			Global/local	Global
Function	Indicates the IP address setting errors for the communications port 1. TRUE: • There is an illegal IP address setting. • A read operation failed. • The IP address obtained from the BOOTP server is inconsistent. FALSE: Normal **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Related instructions	Related instructions	---		

Variable name	_EIP2_IPAdrCfgErr				
Meaning	Port2 IP Address Setting Error			Global/local	Global
Function	Indicates the IP address setting errors for the communications port 2. TRUE: • There is an illegal IP address setting. • A read operation failed. • The IP address obtained from the BOOTP server is inconsistent. FALSE: Normal **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_IPAdrDupErr				
Meaning	IP Address Duplication Error			Global/local	Global
Function	NX-series CPU Units: Indicates that the same IP address is assigned to more than one node for the communications port 1. TRUE: Duplication occurred. FALSE: Other than the above. NJ-series CPU Units: Indicates that the same IP address is assigned to more than one node. TRUE: Duplication occurred. FALSE: Other than the above.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_IPAdrDupErr				
Meaning	Port1 IP Address Duplication Error			Global/local	Global
Function	Indicates that the same IP address is assigned to more than one node for the communications port 1. TRUE: Duplication occurred. FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_IPAdrDupErr				
Meaning	Port2 IP Address Duplication Error			Global/local	Global
Function	Indicates that the same IP address is assigned to more than one node for the communications port 2. TRUE: Duplication occurred. FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_DNSCfgErr*1				
Meaning	DNS Setting Error			Global/local	Global
Function	Indicates that the DNS or hosts settings are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

*1. With the NJ-series CPU Unit, this variable can be used with the unit version 1.11 or later.

Variable name	_EIP_BootpErr				
Meaning	BOOTP Server Error			Global/local	Global
Function	NX-series CPU Units: Indicates that a BOOTP server connection failure occurred on the communications port 1. TRUE: There was a failure to connect to the BOOTP server (timeout). FALSE: The BOOTP is not enabled, or BOOTP is enabled and an IP address was normally obtained from the BOOTP server. NJ-series CPU Units: Indicates that a BOOTP server connection failure occurred. TRUE: There was a failure to connect to the BOOTP server (timeout). FALSE: The BOOTP is not enabled, or BOOTP is enabled and an IP address was normally obtained from the BOOTP server.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_BootpErr				
Meaning	Port1 BOOTP Server Error			Global/local	Global
Function	Indicates that a BOOTP server connection failure occurred on the communications port 1. TRUE: There was a failure to connect to the BOOTP server (timeout). FALSE: The BOOTP is not enabled, or BOOTP is enabled and an IP address was normally obtained from the BOOTP server. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_BootpErr				
Meaning	Port2 BOOTP Server Error			Global/local	Global
Function	Indicates that a BOOTP server connection failure occurred on the communications port 2. TRUE: There was a failure to connect to the BOOTP server (timeout). FALSE: The BOOTP is not enabled, or BOOTP is enabled and an IP address was normally obtained from the BOOTP server. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_IPRTblErr				
Meaning	IP Route Table Error			Global/local	Global
Function	NX-series CPU Units: Indicates that the default gateway settings or IP router table settings are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal NJ-series CPU Units: Indicates that the IP router table or hosts settings are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_IdentityErr				
Meaning	Identity Error			Global/local	Global
Function	NX-series CPU Units: Indicates that the identity information for CIP communications 1 (which you cannot overwrite) is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal NJ-series CPU Units: Indicates that the identity information (which you cannot overwrite) is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_IdentityErr				
Meaning	CIP Communications1 Identity Error			Global/local	Global
Function	Indicates that the identity information for CIP communications 1 (which you cannot overwrite) is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_IdentityErr				
Meaning	CIP Communications2 Identity Error			Global/local	Global
Function	Indicates that the identity information for CIP communications 2 (which you cannot overwrite) is incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_TDLinkCfgErr				
Meaning	Tag Data Link Setting Error			Global/local	Global
Function	NX-series CPU Units: Indicates that the tag data link settings for CIP communications 1 are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal NJ-series CPU Units: Indicates that the tag data link settings are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_TDLinkCfgErr				
Meaning	CIP Communications1 Tag Data Link Setting Error			Global/local	Global
Function	Indicates that the tag data link settings for CIP communications 1 are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_TDLinkCfgErr				
Meaning	CIP Communications2 Tag Data Link Setting Error			Global/local	Global
Function	Indicates that the tag data link setting for CIP communications 2 are incorrect. Or, a read operation failed. TRUE: Setting incorrect or read failed. FALSE: Normal **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_TDLinkOpnErr				
Meaning	Tag Data Link Connection Failed			Global/local	Global
Function	NX-series CPU Units: Indicates that establishing a tag data link connection for CIP communications 1 failed. TRUE: Establishing a tag data link connection failed due to one of the following causes. • The information registered for a target node in the tag data link parameters is different from the actual node information. • There was no response from the remote node. FALSE: Other than the above. NJ-series CPU Units: Indicates that establishing a tag data link connection failed. TRUE: Establishing a tag data link connection failed due to one of the following causes. • The information registered for a target node in the tag data link parameters is different from the actual node information. • There was no response from the remote node. FALSE: Other than the above.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_TDLinkOpnErr				
Meaning	CIP Communications1 Tag Data Link Connection Failed			Global/local	Global
Function	Indicates that establishing a tag data link connection for CIP communications 1 failed. TRUE: Establishing a tag data link connection failed due to one of the following causes. • The information registered for a target node in the tag data link parameters is different from the actual node information. • There was no response from the remote node. FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_TDLinkOpnErr				
Meaning	CIP Communications2 Tag Data Link Connection Failed			Global/local	Global
Function	Indicates that establishing a tag data link connection for CIP communications 2 failed. TRUE: Establishing a tag data link connection failed due to one of the following causes. • The information registered for a target node in the tag data link parameters is different from the actual node information. • There was no response from the remote node. FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_TDLinkErr				
Meaning	Tag Data Link Communications Error			Global/local	Global
Function	NX-series CPU Units: Indicates that a timeout occurred in a tag data link connection for CIP communications 1. TRUE: A timeout occurred. FALSE: Other than the above. NJ-series CPU Units: Indicates that a timeout occurred in a tag data link connection. TRUE: A timeout occurred. FALSE: Other than the above.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_TDLinkErr				
Meaning	CIP Communications1 Tag Data Link Communications Error			Global/local	Global
Function	Indicates that a timeout occurred in a tag data link connection for CIP communications 1. TRUE: A timeout occurred. FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_TDLinkErr				
Meaning	CIP Communications2 Tag Data Link Communications Error			Global/local	Global
Function	Indicates that a timeout occurred in a tag data link connection for CIP communications 2. TRUE: A timeout occurred. FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_TagAdrErr				
Meaning	Tag Name Resolution Error			Global/local	Global
Function	NX-series CPU Units: Indicates that the tag resolution for CIP communications 1 failed (i.e., the address could not be identified from the tag name). TRUE: Tag resolution failed (i.e., the address could not be identified from the tag name). The following causes are possible. • The size of the network variable is different from the tag settings. • The I/O direction that is set in the tag data link settings does not agree with the I/O direction of the variable in the CPU Unit. • There is no network variable in the CPU Unit that corresponds to the tag setting. FALSE: Other than the above. NJ-series CPU Units: Indicates that tag name resolution failed (i.e., the address could not be identified from the tag name). TRUE: Tag resolution failed (i.e., the address could not be identified from the tag name). The following causes are possible. • The size of the network variable is different from the tag settings. • The I/O direction that is set in the tag data link settings does not agree with the I/O direction of the variable in the CPU Unit. • There is no network variable in the CPU Unit that corresponds to the tag setting. FALSE: Other than the above.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_TagAdrErr				
Meaning	CIP Communications1 Tag Name Resolution Error			Global/local	Global
Function	Indicates that the tag resolution for CIP communications 1 failed (i.e., the address could not be identified from the tag name). TRUE: Tag resolution failed (i.e., the address could not be identified from the tag name). The following causes are possible. • The size of the network variable is different from the tag settings. • The I/O direction that is set in the tag data link settings does not agree with the I/O direction of the variable in the CPU Unit. • There is no network variable in the CPU Unit that corresponds to the tag setting. FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_TagAdrErr				
Meaning	CIP Communications2 Tag Name Resolution Error			Global/local	Global
Function	Indicates that the tag resolution for CIP communications 2 failed (i.e., the address could not be identified from the tag name). TRUE: Tag resolution failed (i.e., the address could not be identified from the tag name). The following causes are possible. • The size of the network variable is different from the tag settings. • The I/O direction that is set in the tag data link settings does not agree with the I/O direction of the variable in the CPU Unit. • There is no network variable in the CPU Unit that corresponds to the tag setting. FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_MultiSwONErr				
Meaning	Multiple Switches ON Error			Global/local	Global
Function	NX-series CPU Units: Indicates that more than one switch turned ON at the same time in CIP communications 1. TRUE: More than one data link start/stop switch changed to TRUE at the same time. FALSE: Other than the above. NJ-series CPU Units: Indicates that more than one switch turned ON at the same time TRUE: More than one data link start/stop switch changed to TRUE at the same time. FALSE: Other than the above.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_MultiSwONErr				
Meaning	CIP Communications1 Multiple Switches ON Error			Global/local	Global
Function	Indicates that more than one switch turned ON at the same time in CIP communications 1. TRUE: More than one data link start/stop switch changed to TRUE at the same time. FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_MultiSwONErr				
Meaning	CIP Communications2 Multiple Switches ON Error			Global/local	Global
Function	Indicates that more than one switch turned ON at the same time in CIP communications 2. TRUE: More than one data link start/stop switch changed to TRUE at the same time. FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_TcpAppCfgErr				
Meaning	TCP Application Setting Error			Global/local	Global
Function	TRUE: At least one of the set values for a TCP application (FTP, NTP, SNMP) is incorrect. Or, a read operation failed. FALSE: Normal				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_NTPSrvErr				
Meaning	NTP Server Connection Error			Global/local	Global
Function	TRUE: The NTP client failed to connect to the server (timeout). FALSE: NTP is not set. Or, NTP is set and the connection was successful.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_DNSSrvErr				
Meaning	DNS Server Connection Error			Global/local	Global
Function	TRUE: The DNS client failed to connect to the server (timeout). FALSE: DNS is not enabled. Or, DNS is enabled and the connection was successful.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Global
Usage in user program	Possible.	Related instructions	---		

● Functional Classification: EtherNet/IP Communications Status

Variable name	_EIP_EtnOnlineSta				
Meaning	Online			Global/local	Global
Function	NX-series CPU Units: Indicates that the built-in EtherNet/IP port's communications can be used via the communications port 1 (that is, the link is ON, IP address is defined, and there are no errors.) TRUE: The built-in EtherNet/IP port’s communications can be used. FALSE: The built-in EtherNet/IP port’s communications is disabled due to an error in initial processing, restart processing, or link OFF status. NJ-series CPU Units: Indicates that the built-in EtherNet/IP port's communications can be used via the communications port (that is, the link is ON and IP address is defined, and there are no errors.) TRUE: The built-in EtherNet/IP port’s communications can be used. FALSE: The built-in EtherNet/IP port’s communications is disabled due to an error in initial processing, restart processing, or link OFF status.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_EtnOnlineSta				
Meaning	Port1 Online			Global/local	Global
Function	Indicates that the built-in EtherNet/IP port's communications can be used via the communications port 1 (that is, the link is ON, IP address is defined, and there are no errors.) TRUE: The built-in EtherNet/IP port's communications can be used. FALSE: The built-in EtherNet/IP port's communications is disabled due to an error in initial processing, restart processing, or link OFF status. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_EtnOnlineSta				
Meaning	Port2 Online				Global
Function	Indicates that the built-in EtherNet/IP port's communications can be used via the communications port 2 (that is, the link is ON, IP address is defined, and there are no errors.) TRUE: The built-in EtherNet/IP port's communications can be used. FALSE: The built-in EtherNet/IP port's communications is disabled due to an error in initial processing, restart processing, or link OFF status. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_TDLinkRunSta				
Meaning	Tag Data Link Communications Status			Global/local	Global
Function	NX-series CPU Units: Indicates that at least one connection is in normal operation in CIP communications 1. TRUE: Normal operation FALSE: Other than the above. NJ-series CPU Units: Indicates that at least one connection is in normal operation. TRUE: Normal operation FALSE: Other than the above.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_TDLinkRunSta				
Meaning	CIP Communications1 Tag Data Link Communications Status			Global/local	Global
Function	Indicates that at least one connection is in normal operation in CIP communications 1. TRUE: Normal operation FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_TDLinkRunSta				
Meaning	CIP Communications2 Tag Data Link Communications Status			Global/local	Global
Function	Indicates that at least one connection is in normal operation in CIP communications 2. TRUE: Normal operation FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_TDLinkAllRunSta				
Meaning	All Tag Data Link Communications Status			Global/local	Global
Function	NX-series CPU Units: Indicates that all tag data links are communicating in CIP communications 1. TRUE: Tag data links are communicating in all connections as the originator. FALSE: An error occurred in at least one connection. NJ-series CPU Units: Indicates that all tag data links are communicating. TRUE: Tag data links are communicating in all connections as the originator. FALSE: An error occurred in at least one connection.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_TDLinkAllRunSta				
Meaning	CIP Communications1 All Tag Data Link Communications Status			Global/local	Global
Function	Indicates that all tag data links are communicating in CIP communications 1. TRUE: Tag data links are communicating in all connections as the originator. FALSE: An error occurred in at least one connection. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_TDLinkAllRunSta				
Meaning	CIP Communications2 All Tag Data Link Communications Status			Global/local	Global
Function	Indicates that all tag data links are communicating in CIP communications 2. TRUE: Tag data links are communicating in all connections as the originator. FALSE: An error occurred in at least one connection. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_RegTargetSta[255]				
Meaning	Registered Target Node Information			Global/local	Global
Function	NX-series CPU Units: Gives a list of nodes for which built-in EtherNet/IP connections are registered for CIP communications 1. This variable is valid only when the built-in EtherNet/IP port is the originator. Array[x] is TRUE: The connection to the node with a target node ID of x is registered. Array[x] is FALSE: The connection to the node with a target node ID of x is not registered. NJ-series CPU Units: Gives a list of nodes for which built-in EtherNet/IP connections are registered. This variable is valid only when the built-in EtherNet/IP port is the originator. Array[x] is TRUE: The connection to the node with a target node ID of x is registered. Array[x] is FALSE: The connection to the node with a target node ID of x is not registered.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_RegTargetSta[255]				
Meaning	CIP Communications1 Registered Target Node Information			Global/local	Global
Function	Gives a list of nodes for which built-in EtherNet/IP connections are registered for CIP communications 1. This variable is valid only when the built-in EtherNet/IP port is the originator. Array[x] is TRUE: The connection to the node with a target node ID of x is registered. Array[x] is FALSE: The connection to the node with a target node ID of x is not registered. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_RegTargetSta[255]				
Meaning	CIP Communications2 Registered Target Node Information			Global/local	Global
Function	Gives a list of nodes for which built-in EtherNet/IP connections are registered for CIP communications 2. This variable is valid only when the built-in EtherNet/IP port is the originator. Array[x] is TRUE: The connection to the node with a target node ID of x is registered. Array[x] is FALSE: The connection to the node with a target node ID of x is not registered. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_EstbTargetSta[255]				
Meaning	Normal Target Node Information			Global/local	Global
Function	NX-series CPU Units: Gives a list of nodes that have normally established built-in EtherNet/IP connections for CIP communications 1. Array[x] is TRUE: The connection to the node with a target node ID of x was established normally. Array[x] is FALSE: The connection to the node with a target node ID of x was not established, or an error occurred. NJ-series CPU Units: Gives a list of nodes that have normally established built-in EtherNet/IP connections. Array[x] is TRUE: The connection to the node with a target node ID of x was established normally. Array[x] is FALSE: The connection to the node with a target node ID of x was not established, or an error occurred.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_EstbTargetSta[255]				
Meaning	CIP Communications1 Normal Target Node Information			Global/local	Global
Function	Gives a list of nodes that have normally established built-in EtherNet/IP connections for CIP communications 1. Array[x] is TRUE: The connection to the node with a target node ID of x was established normally. Array[x] is FALSE: The connection to the node with a target node ID of x was not established, or an error occurred. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_EstbTargetSta[255]				
Meaning	CIP Communications2 Normal Target Node Information			Global/local	Global
Function	Gives a list of nodes that have normally established built-in EtherNet/IP connections for CIP communications 2. Array[x] is TRUE: The connection to the node with a target node ID of x was established normally. Array[x] is FALSE: The connection to the node with a target node ID of x was not established, or an error occurred. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_TargetPLCModeSta[255]				
Meaning	Target PLC Operating Mode			Global/local	Global
Function	NX-series CPU Units: Shows the operating status of the target node Controllers that are connected for CIP communications 1, with the built-in EtherNet/IP port as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. If the corresponding Normal Target Node Information is FALSE, it indicates the previous operating status. Array[x] is TRUE: This is the operating state of the target Controller with a node address of x. Array[x] is FALSE: Other than the above. NJ-series CPU Units: Shows the operating status of the target node Controllers that are connected with the built-in EtherNet/IP port as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. If the corresponding Normal Target Node Information is FALSE, it indicates the previous operating status. Array[x] is TRUE: This is the operating state of the target Controller with a node address of x. Array[x] is FALSE: Other than the above.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_TargetPLCModeSta[255]				
Meaning	CIP Communications1 Target PLC Operating Mode			Global/local	Global
Function	Shows the operating status of the target node Controllers that are connected for CIP communications 1, with the built-in EtherNet/IP port as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. If the corresponding Normal Target Node Information is FALSE, it indicates the previous operating status. Array[x] is TRUE: This is the operating state of the target Controller with a node address of x. Array[x] is FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_TargetPLCModeSta[255]				
Meaning	CIP Communications2 Target PLC Operating Mode			Global/local	Global
Function	Shows the operating status of the target node Controllers that are connected for CIP communications 2, with the built-in EtherNet/IP port as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. If the corresponding Normal Target Node Information is FALSE, it indicates the previous operating status. Array[x] is TRUE: This is the operating state of the target Controller with a node address of x. Array[x] is FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_TargetPLCErr[255]				
Meaning	Target PLC Error Information			Global/local	Global
Function	NX-series CPU Units: Shows the error status (logical OR of fatal and non-fatal errors) of the target node Controllers that are connected for CIP communications 1, with the built-in EtherNet/IP ports as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. The immediately preceding value is retained if this variable is FALSE. Array[x] is TRUE: A fatal or non-fatal error occurred in the target Controller with a target node ID of x. Array[x] is FALSE: Other than the above. NJ-series CPU Units: Shows the error status (logical OR of fatal and non-fatal errors) of the target node Controllers that are connected with the built-in EtherNet/IP ports as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. The immediately preceding value is retained if this variable is FALSE. Array[x] is TRUE: A fatal or non-fatal error occurred in the target Controller with a target node ID of x. Array[x] is FALSE: Other than the above.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_TargetPLCErr[255]				
Meaning	CIP Communications1 Target PLC Error Information			Global/local	Global
Function	Shows the error status (logical OR of fatal and non-fatal errors) of the target node Controllers that are connected for CIP communications 1, with the built-in EtherNet/IP ports as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. The immediately preceding value is retained if this variable is FALSE. Array[x] is TRUE: A fatal or non-fatal error occurred in the target Controller with a target node ID of x. Array[x] is FALSE: Other than the above. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_TargetPLCErr[255]				
Meaning	CIP Communications2 Target PLC Error Information			Global/local	Global
Function	Shows the error status (logical OR of fatal and non-fatal errors) of the target node Controllers that are connected for CIP communications 2, with the built-in EtherNet/IP ports as the originator. The array elements are valid only when the corresponding Normal Target Node Information is TRUE. The immediately preceding value is retained if this variable is FALSE. Array[x] is TRUE: A fatal or non-fatal error occurred in the target Controller with a target node ID of x. Array[x] is FALSE: Other than the above. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_TargetNodeErr[255]				
Meaning	Target Node Error Information			Global/local	Global
Function	NX-series CPU Units: Indicates that the connection for the Registered Target Node Information for CIP communications 1 was not established or that an error occurred in the target Controller. The array elements are valid only when the Registered Target Node Information is TRUE. Array[x] is TRUE: A connection was not normally established with the target node for a target node ID of x (the Registered Target Node Information is TRUE and the Normal Target Node Information is FALSE), or a connection was established with the target node but an error occurred in the target Controller. Array[x] is FALSE: The target node is not registered for a target node ID of x (the Registered Target Node Information is FALSE), or a connection was normally established with the target node (the Registered Target Node Information is TRUE and the Normal Target Node Information is TRUE). An error occurred in the target Controller (the Target PLC Error Information is TRUE). NJ-series CPU Units: Indicates that the connection for the Registered Target Node Information was not established or that an error occurred in the target Controller. The array elements are valid only when the Registered Target Node Information is TRUE. Array[x] is TRUE: A connection was not normally established with the target node for a target node ID of x (the Registered Target Node Information is TRUE and the Normal Target Node Information is FALSE), or a connection was established with the target node but an error occurred in the target Controller. Array[x] is FALSE: The target node is not registered for a target node ID of x (the Registered Target Node Information is FALSE), or a connection was normally established with the target node (the Registered Target Node Information is TRUE and the Normal Target Node Information is TRUE). An error occurred in the target Controller (the Target PLC Error Information is TRUE).				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_TargetNodeErr[255]				
Meaning	CIP Communications1 Target Node Error Information			Global/local	Global
Function	Indicates that the connection for the Registered Target Node Information for CIP communications 1 was not established or that an error occurred in the target Controller. The array elements are valid only when the Registered Target Node Information is TRUE. Array[x] is TRUE: A connection was not normally established with the target node for a target node ID of x (the Registered Target Node Information is TRUE and the Normal Target Node Information is FALSE), or a connection was established with the target node but an error occurred in the target Controller. Array[x] is FALSE: The target node is not registered for a target node ID of x (the Registered Target Node Information is FALSE), or a connection was normally established with the target node (the Registered Target Node Information is TRUE and the Normal Target Node Information is TRUE). An error occurred in the target Controller (the Target PLC Error Information is TRUE). **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_TargetNodeErr[255]				
Meaning	CIP Communications2 Target Node Error Information			Global/local	Global
Function	Indicates that the connection for the Registered Target Node Information for CIP communications 2 was not established or that an error occurred in the target Controller. The array elements are valid only when the Registered Target Node Information is TRUE. Array[x] is TRUE: A connection was not normally established with the target node for a target node ID of x (the Registered Target Node Information is TRUE and the Normal Target Node Information is FALSE), or a connection was established with the target node but an error occurred in the target Controller. Array[x] is FALSE: The target node is not registered for a target node ID of x (the Registered Target Node Information is FALSE), or a connection was normally established with the target node (the Registered Target Node Information is TRUE and the Normal Target Node Information is TRUE). An error occurred in the target Controller (the Target PLC Error Information is TRUE). **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	ARRAY [0..255] OF BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_NTPResult			Member name	.ExecTime
Meaning	NTP Last Operation Time			Global/local	Global
Function	Gives the last time that NTP processing ended normally. The time that was obtained from the NTP server is stored when the time is obtained normally. The time is not stored if it is not obtained from the NTP server normally. **Note** Do not use this variable in the user program. There may be a delay in updating it. Use this variable only to access status through communications from an external device.				
Data type	Structure: _sNTP_RESULT Members: DATE_AND_TIME			Range of values	Depends on data type.
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Not possible.	Related instructions	You can read the contents of this variable with the GetNTPStatus instruction.		

Variable name	_EIP_NTPResult			Member name	.ExecNormal
Meaning	NTP Operation Result			Global/local	Global
Function	This variable shows if the NTP operation ended normally. TRUE: Indicates an NTP normal end. FALSE:Indicates that NTP operation ended in an error or has not been executed even once. **Note** Do not use this variable in the user program. There may be a delay in updating it. Use this variable only to access status through communications from an external device.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	R	Retained	Not retained.	Network Publish	Published.
Usage in user program	Not possible.	Related instructions	You can read the contents of this variable with the GetNTPStatus instruction.		

● Functional Classification: EtherNet/IP Communications Switches

Variable name	_EIP_TDLinkStartCmd				
Meaning	Tag Data Link Communications Start Switch			Global/local	Global
Function	NX-series CPU Units: Change this variable to TRUE to start tag data links for CIP communications 1. It automatically changes back to FALSE after tag data link operation starts. NJ-series CPU Units: Change this variable to TRUE to start tag data links. It automatically changes back to FALSE after tag data link operation starts. **Note** Do not force this switch to change to FALSE from the user program or from the Sysmac Studio. It changes to FALSE automatically.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	RW	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_TDLinkStartCmd				
Meaning	CIP Communications1 Tag Data Link Communications Start Switch			Global/local	Global
Function	Change this variable to TRUE to start tag data links for CIP communications 1. It automatically changes back to FALSE after tag data link operation starts. **Note** Do not force this switch to change to FALSE from the user program or from the Sysmac Studio. It changes to FALSE automatically. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	RW	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP2_TDLinkStartCmd				
Meaning	CIP Communications2 Tag Data Link Communications Start Switch			Global/local	Global
Function	Change this variable to TRUE to start tag data links for CIP communications 2. It automatically changes back to FALSE after tag data link operation starts. **Note** Do not force this switch to change to FALSE from the user program or from the Sysmac Studio. It changes to FALSE automatically. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	RW	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP_TDLinkStopCmd				
Meaning	Tag Data Link Communications Stop Switch			Global/local	Global
Function	NX-series CPU Units: Change this variable to TRUE to stop tag data links for CIP communications 1. It automatically changes back to FALSE after tag data link operation stops. NJ-series CPU Units: Change this variable to TRUE to stop tag data links. It automatically changes back to FALSE after tag data link operation stops. **Note** Do not force this switch to change to FALSE from the user program or from the Sysmac Studio. It changes to FALSE automatically.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	RW	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

Variable name	_EIP1_TDLinkStopCmd				
Meaning	CIP Communications1 Tag Data Link Communications Stop Switch			Global/local	Global
Function	Change this variable to TRUE to stop tag data links for CIP communications 1. It automatically changes back to FALSE after tag data link operation stops. **Note** Do not force this switch to change to FALSE from the user program or from the Sysmac Studio. It changes to FALSE automatically. **Note** You can use this system-defined variable only for NX-series CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	RW	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

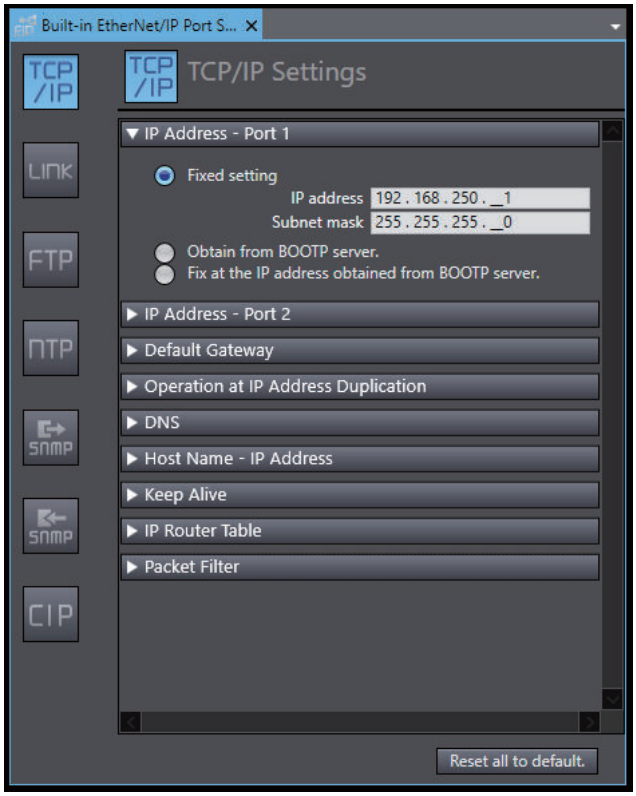
Variable name	_EIP2_TDLinkStopCmd				
Meaning	CIP Communications2 Tag Data Link Communications Stop Switch			Global/local	Global
Function	Change this variable to TRUE to stop tag data links for CIP communications 2. It automatically changes back to FALSE after tag data link operation stops. **Note** Do not force this switch to change to FALSE from the user program or from the Sysmac Studio. It changes to FALSE automatically. **Note** You can use this system-defined variable only for the NX701 CPU Units and NX102 CPU Units.				
Data type	BOOL			Range of values	TRUE or FALSE
R/W access	RW	Retained	Not retained.	Network Publish	Published.
Usage in user program	Possible.	Related instructions	---		

4

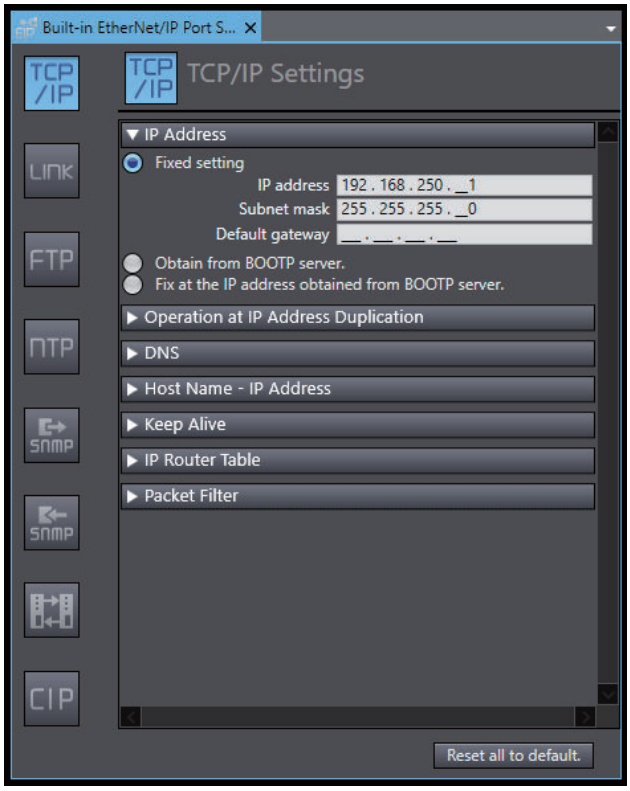
Sysmac Studio Settings for the Built-in EtherNet/IP Port

4-1 TCP/IP Settings Display

For NX701 CPU Units

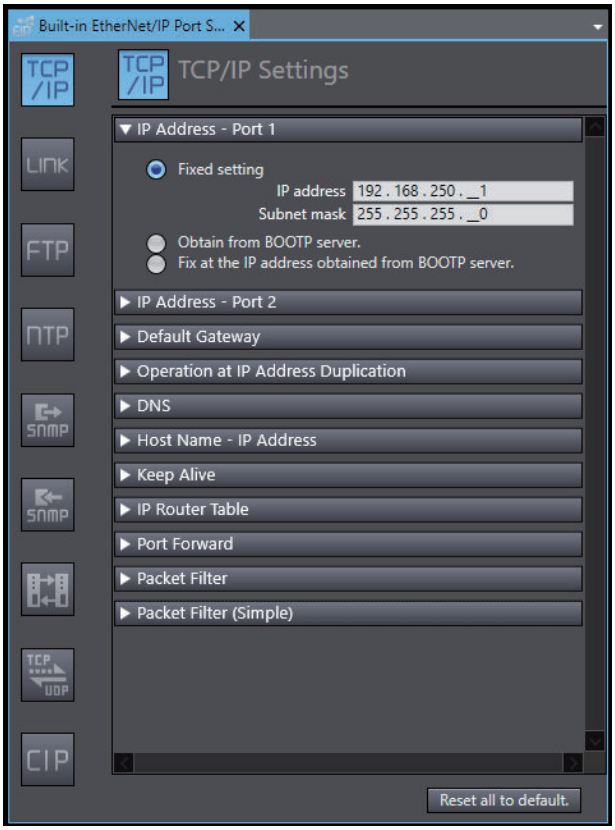


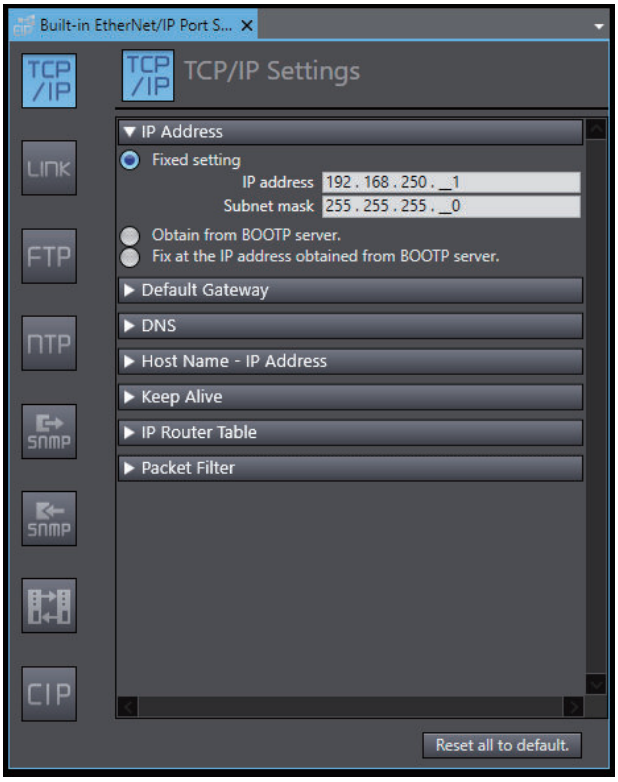
For NJ-series CPU Units



For NX102 CPU units

For NX1P2 CPU Units





● IP Address - Port 1 (NX-series CPU Unit)

Set an IP address for the built-in EtherNet/IP port 1.

Setting	Description	Default
IP address setting method	Select one of the following IP address setting methods for the built-in EtherNet/IP port 1. • Fixed setting • Obtain from BOOTP server. • Fix at the IP address obtained from BOOTP server.	Fixed setting
IP address*1	Set the IP address for the built-in EtherNet/IP port 1. *2	192.168.250.1
Subnet mask*2	Set the subnet mask for the built-in EtherNet/IP port 1.	255.255.255.0

*1. These settings are required if you set IP address setting method to **Fixed setting**.

*2. Refer to *5-1-2 Built-in EtherNet/IP Port IP Address Settings* on page 5-4 for details on setting IP addresses.

● IP Address - Port 2 (NX701 and NX102 CPU Units)

Set an IP address for the built-in EtherNet/IP port 2.

Setting	Description	Default
Use Port 2	Select the check box to use the built-in EtherNet/IP port 2.	Selected (use)
IP address setting method	Select one of the following IP address setting methods for the built-in EtherNet/IP port 2. • Fixed setting • Obtain from BOOTP server. • Fix at the IP address obtained from BOOTP server.	Fixed setting
IP address*1	Set the IP address for the built-in EtherNet/IP port 2. *2	192.168.251.1

Setting	Description	Default
Subnet mask*2	Set the subnet mask for the built-in EtherNet/IP port 2.	255.255.255.0

*1. These settings are required if you select **Fixed setting** for the IP address setting method.
*2. Refer to *5-1-2 Built-in EtherNet/IP Port IP Address Settings* on page 5-4 for details on setting IP addresses.

Precautions for Correct Use

For NX701 CPU Units and NX102 CPU Units, you cannot set IP addresses that make two built-in EtherNet/IP ports belong to the same network.

● IP Address (NJ-series CPU Unit)

Setting	Description	Default
IP address setting method	Select one of the following IP address setting methods for the built-in EtherNet/IP port. *1 • Fixed setting • Obtain from BOOTP server. • Fix at the IP address obtained from BOOTP server.	Fixed setting
IP address*2	Set the IP address for the built-in EtherNet/IP port.	192.168.250.1
Subnet mask*2	Set the subnet mask for the built-in EtherNet/IP port.	255.255.255.0
Default gateway*3	Set the IP address of the default gateway for the built-in EtherNet/IP port. This setting is not required when the default gateway is not used.	None

*1. Refer to *5-1-2 Built-in EtherNet/IP Port IP Address Settings* on page 5-4 for details on setting IP addresses.
*2. These settings are required if you select **Fixed setting** for the IP address setting method.
*3. This setting is valid if you select **Fixed setting** for the IP address setting method.

● Default Gateway (NX-series CPU Unit)

Setting	Description	Default
Default gateway*1	Set the IP address of the default gateway for the built-in EtherNet/IP port. *2 This setting is not required when the default gateway is not used.	None

*1. If you select **Obtain from BOOTP server** or **Fix at the IP address obtained from BOOTP server** for the IP address setting method, the default gateway obtained from a BOOTP server is enabled.
*2. For NX701 and NX102 CPU Units, even if you are using both of port 1 and port 2, you can set the default gateway for only one of the ports.

Operation at IP Address Duplication

Setting	Description	Default
Use of duplicated IP address	When you set an IP address for the built-in EtherNet/IP port and find an IP address conflict with another node, select whether to stop the use of the IP address. • Stop If the IP address conflict is not resolved for a certain length of time, the use of the IP address is stopped, and an IP Address Duplication Error will occur. • Do not stop*1 You continue to use the IP address and wait until the duplicate IP address of the other node is changed.	Stop

*1. For systems that use OPC UA to connect to an information system network, it is recommended that you set this to **Do not stop**.

Version Information

The setting for the **Use of duplicated IP address** can be used with the CPU Units that support OPC UA, and the Sysmac Studio. Refer to the *NJ/NX-series CPU Unit OPC UA User's Manual (Cat. No. W588)* for information on the models and unit versions of the CPU Units that support OPC UA, and the Sysmac Studio version.

DNS

Setting	Description	Default
Use/ Do not use DNS	When you specify a host name for CIP communications instructions, socket instructions or NTP server settings, select the **Use** Option if you use DNS for resolving host name. A DNS server is required to use DNS.	Do Not Use
Priority DNS server*1	Set the IP address of the DNS server. You can set priority and secondary IP addresses.	None
Secondary DNS server		None
Domain name*1	Set the domain name of the domain to which the built-in EtherNet/IP port belongs. (Single-byte alphanumeric characters, dots, and hyphens: 48 characters max.)	None

*1. These settings are required if you select the **Use** Option for **DNS**.

Host Name - IP Address

Setting	Description	Default
Host name	Addresses are converted according to this setting when a host name is used to specify remote communications nodes. Host names can be set whether DNS is used or not. You can set up to six host names. (You can use up to 200 single-byte alphanumeric characters, dots, and hyphens, including up to 63 single-byte alphanumeric characters between dots.)	None
IP address	Set the IP address of the registered host name.	None

● Keep Alive

Setting	Description	Default
Keep Alive	Set whether to use the remote node Keep Alive function of connected servers and clients (such as socket service, FTP server, Sysmac Studio, and FINS/TCP) for each connection number. If the **Use** Option is selected for **Keep Alive** and no communications are performed with the remote node for the **Keep Alive monitoring time**, transmission of Keep Alive packets is started. The connection will be disconnected if the remote node does not respond for longer than five times the total time of Keep Alive packet transmission + five seconds for resending.*1 The connection to the remote node is left open if the power supply to the remote node is turned OFF without warning. Select the **Use** Option for Keep Alive wherever possible. • Use • Do not use	Use
Keep Alive monitoring time	This is a set period of time before the transmission of Keep Alive packets is started. Setting range: 1 to 65,535 (seconds)	300
Linger option	Set whether to specify the Linger Option for connections to FINS/TCP or socket services. If the Linger Option is specified, the port number is immediately opened even before the port number is released after the socket closes (approx. 1 minute). • Specify • Do not specify	Do not specify

*1. If the remote node does not respond, the connection is disconnected after the Keep Alive monitoring time + 30 seconds.

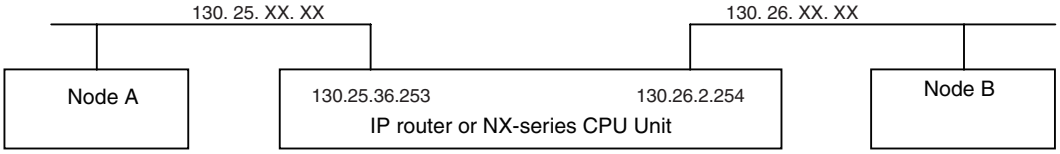
● IP Router Table

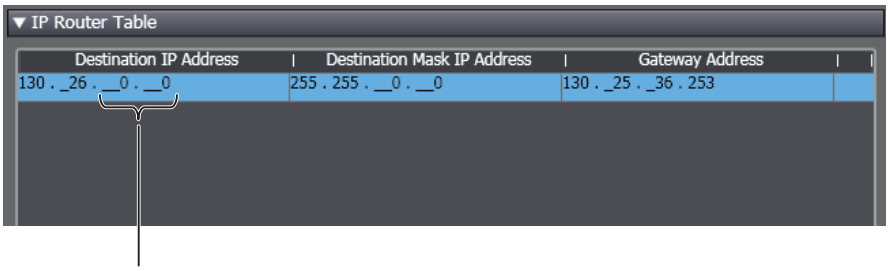
Setting	Description	Default
Destination IP Address	Set these settings when the built-in EtherNet/IP port is used for tag data links or CIP message communications with nodes on other IP network segments via an IP router. Accordingly, set these settings when you use an NX-series CPU Unit as an IP router using the IP routing function for the built-in EtherNet/IP port. You can set up to 128 combinations of an IP address and a gateway address for an NX701 CPU Unit, up to 64 combinations for an NX102 CPU Unit, and up to eight combinations for an NJ-series CPU Unit or an NX1P2 CPU Unit. Specify 0 for the host portions of the IP addresses.	None
Destination Mask IP Address		
Gateway Address		None

Additional Information

IP Router Table Setting Example

Set the following IP router table in node A to use tag data links or CIP message communications between node A and node B through the IP router.
If you set the IP router table and execute a communications instruction from node A to node B, node A sends packets addressed to the gateway IP address (130.25.36.253).





The host fields are set to 0 in the destination IP address.

● Port Forward (NX102 CPU Units)

Setting	Description	Default
IP Forward	Select whether to transfer IP packets between communications ports.	Use

Precautions for Correct Use

For CPU Units other than the NX102 CPU Unit, there is no setting for port forward. To disable port forward, specify the IP address of the built-in EtherNet/IP port in the destination IP address of the Packet Filter.

● Packet Filter

For information on usage and restrictions of Packet Filter, refer to *5-4 Packet Filter* on page 5-22.

Setting	Description	Default
Packet Filter	Select whether to use Packet Filter or not. Use Do not use	Do not use
Source	Set the conditions for the source	---
IP Address Specification Method	Select the method for specifying the IP address of the source. any*1 IP address specification	any
IP Address	If the IP address specification method is **IP address specification**, set the source IP address.*2	None
Mask	If the IP address specification method is **IP address specification**, set the mask of source IP address.*3	None
Destination	Set the conditions for the destination.	---

Setting		Description	Default
	IP Address Specification Method	Same as those for the source	
	IP Address		
	Mask		
Protocol		Set the communications protocol. any*4 tcp udp igmp*5 icmp*6	any
Source Port		If **tcp** or **udp** is selected for Protocol, set the source port conditions.	---
	Specification Method	Select the method for specifying the IP packets of the source port. any*7 Port specification	any
	Range Specification	Specify whether or not to set the port range if the specification method selected is **Port specification**. If it is selected, reception from the source ports from the **Start Number** to the **End number** is allowed. If it is not selected, reception from the source port specified by the **Start Number** is allowed. No check. Checked.	No check.
	Start Number	Set the start number when **Port specification** is selected for the specification method. 1 to 65535	None
	End Number	Set the end number when the specification method is **Port specification** and the range specification is selected. 1 to 65535	None
Destination Port		Set the conditions for the destination port if **tcp** or **udp** is selected for Protocol. Same as the settings for the source port.	
	Specification Method		
	Range Specification		
	Start Number		
	End Number		

*1. If you select any, packets from any IP addresses will be allowed.

*2. The allowed IP address is calculated by the logical AND of the **IP address** and the **Mask**. If you want to allow more than one IP address, mask a part of the IP address by setting the **Mask**. In this case, set 0 to the bits to be masked in the **IP address** and **Mask**.
The following is an example of how to calculate the allowed IP addresses.

Example 1. Allowing IP address 192.168.250.1
If you want to allow one IP address, set 255.255.255.255 to the mask.

Setting	Decimal notation	Binary notation
IP address	192.168.250.1	11000000.10101000.11111010.00000001
Mask	255.255.255.255	11111111.11111111.11111111.11111111

Example 2. Allowing IP address 192.168.250.*
Set 255.255.255.0 to the mask to mask the lower 8 bits of the IP address.

Setting	Decimal notation	Binary notation
IP address	192.168.250.0	11000000.10101000.11111010.00000000
Mask	255.255.255.0	11111111.11111111.11111111.00000000

Example 3. Allowing IP address 192.168.250.1 to 192.168.250.31
Set 255.255.255.224 to the mask to mask the lower 5 bits if the IP address.

Setting	Decimal notation	Binary notation
IP address	192.168.250.0	11000000.10101000.11111010.00000000
Mask	255.255.255.224	11111111.11111111.11111111.11100000

*3. Set 0 to the bits to be masked in **Mask**. Multiple bits can be masked, but only bits from the least significant can be masked. It is not possible to mask the higher bits, such as 0.255.255.255, or the middle bits, such as 255.0.255.255.
The following are examples of setting a mask.

Example 1. Masking the lower 8 bits
Set 0 to the lower 8 bits.

Setting	Decimal notation	Binary notation
Mask	255.255.255.0	11111111.11111111.11111111.00000000

Example 2. Masking the lower 24 bits
Set 0 to the lower 24 bits.

Setting	Decimal notation	Binary notation
Mask	255.0.0.0	11111111.00000000.00000000.00000000

*4. If you select any, packets from tcp, udp, igmp, and icmp will be allowed.
*5. Select igmp when EtherNet/IP tag data links are used for multicast and the built-in EtherNet/IP is specified as the originator.
*6. Select icmp for receiving Ping requests.
*7. If you select any, packets from any TCP/UDP port are allowed.

Version Information

Packet Filer is available in the following CPU Units of the stated versions.
- NJ-series, NX102, NX1P2 CPU Unit: Version 1.49 or later
- NX701 CPU Unit: Version 1.29 or later

● Packet Filter (Simple)

You can select **Use Packet Filter (Simple)** on NX102 CPU Units only.

Setting	Description	Default
Packet Filter (Simple)	Select whether or not to set conditions of IP packets to be received at the communications port.	Do not use
Pass Frame	Set the following items as the conditions of IP packets to be received at the communications port. You can set the conditions under which up to 32 packets are allowed to be received. This setting is valid only when the **Use** Option is selected for **Packet Filter (Simple)**.	---

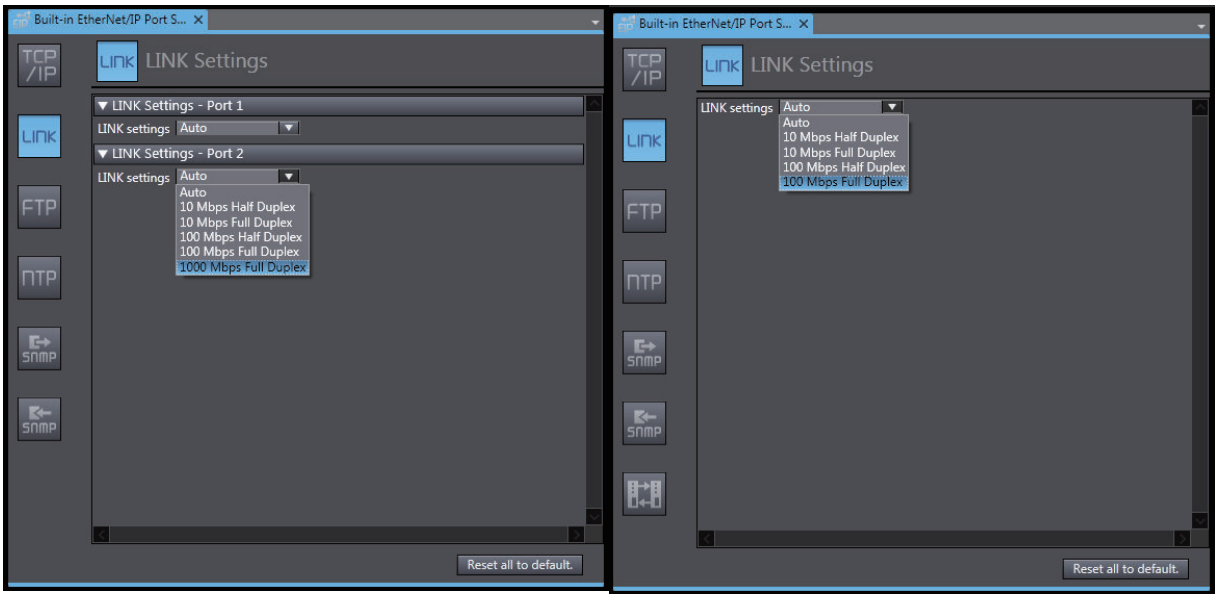
Setting		Description	Default
	Port	Select the communications port to use Packet Filter (Simple).	No.1: Port 1 No.2: Port 2
	Specification Method	Select the method for specifying IP packets to be received. IP address specification any*1	No.1: any No.2: any
	IP Address	Specify an IP address that is allowed to be received.	None
	Mask	Set the mask for the IP address allowed to be received. If you select **IP address specification** for **Specification Method**, *255.255.255.255* is automatically set.	None

*1. If you select any, packets from any IP addresses will be received.

Precautions for Correct Use

- Connections to NA-series and NS-series Programmable Terminals are restricted if this function is enabled. To make connections to these devices, register their IP addresses in the Packet Filter (Simple) settings.
- If this function is enabled, you cannot connect the Sysmac Studio from a computer whose IP address is not registered. Before enabling this function, confirm in advance that the IP address of the computer is correctly registered.
- If this function is enabled, you cannot connect the Sysmac Studio to the Controller with the **Direct connection via Ethernet** Option selected for the connection type. Select **Controller - Communications Setup** to confirm that the connection type is **Ethernet connection via a hub**.
- You can disable this function tentatively by starting the Unit in Safe Mode in case you forget the registered IP address and cannot go online from the Sysmac Studio. Refer to *Troubleshooting When You Cannot Go Online from the Sysmac Studio* in the *NJ/NX-series Troubleshooting Manual (Cat. No. W503)* for details.
- You can use the Packet Filter (Simple) with Sysmac Studio version 1.49 or lower. Use the Packet Filter instead of the Packet Filter (Simple) when you use Sysmac Studio version 1.50 or higher.

4-2 LINK Settings Display



NX701 CPU Unit
NX102 CPU Unit

NJ-series CPU Unit
NX1P2 CPU Unit

● LINK Settings - Port 1 and Port 2 (NX701 and NX102 CPU Units)

Set for each built-in EtherNet/IP port.

Setting	Description	Default
LINK settings	Set the baud rate for the built-in EtherNet/IP ports.*1 • Auto • 10 Mbps Half Duplex • 10 Mbps Full Duplex • 100 Mbps Half Duplex • 100 Mbps Full Duplex • 1000 Mbps Full Duplex (NX701 CPU Unit)	Auto

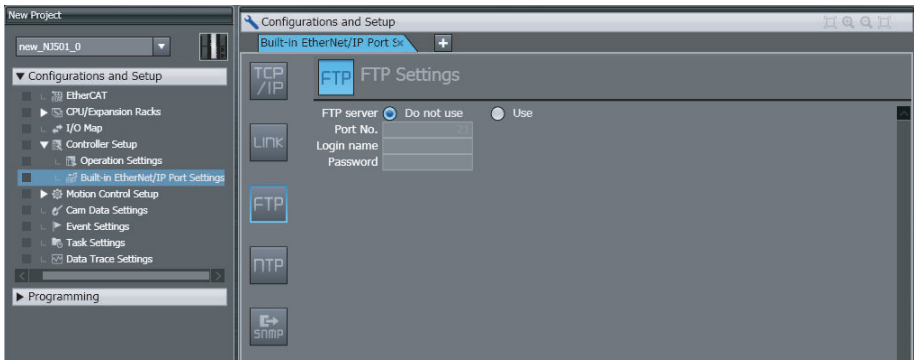
*1. For an NX701 CPU Unit with the hardware revision *B* or later, Auto will be set regardless of the setting of the Sysmac Studio. If an item other than **Auto** is selected and the setting is transferred from the Sysmac Studio, *Link Setting Not Supported* (342B0000 hex) event will occur.

● LINK Settings (NJ-series CPU Unit and NX1P2 CPU Unit)

Setting	Description	Default
LINK settings	Set the baud rate for the built-in EtherNet/IP ports. *1 • Auto • 10 Mbps Half Duplex • 10 Mbps Full Duplex • 100 Mbps Half Duplex • 100 Mbps Full Duplex	Auto

*1. For an NJ-series CPU Unit with the hardware revision *D* or later, Auto will be set regardless of the setting from the Sysmac Studio. If an item other than **Auto** is selected and the setting is transferred from the Sysmac Studio, *Link Setting Not Supported* (342B0000 hex) event will occur.

4-3 FTP Settings Display



Setting	Description	Default
FTP server	Specify whether to use the FTP server or not. FTP connections from external devices will not be possible if the **Do not use** Option is selected.	Do not use
Port No.*1,*2	Set the FTP port number of the built-in EtherNet/IP port. This setting does not normally need to be changed. The FTP control port is set here. The FTP data transfer port is always port 20.	21
Login name *1	Set the login name to externally connect to the built-in EtherNet/IP port via FTP. (You can use up to 12 alphanumeric characters.)*3	None
Password*1	Set the password to externally connect to the built-in EtherNet/IP port via FTP. (You can use 8 to 32 alphanumeric characters.)*3	None

*1. These settings are required when the **Use** Option is selected for the **FTP server**.

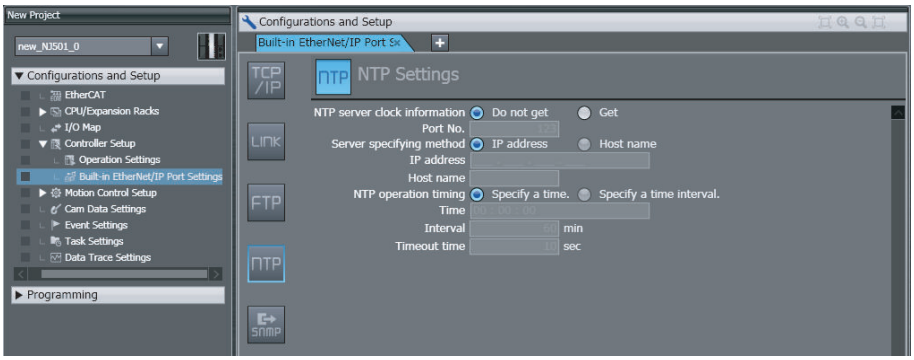
*2. The following ports are used by the system and cannot be set by the user: 20, 23, 25, 80, 110, 9610, and 44818.

*3. The login name and password are case sensitive.

Additional Information

Refer to *Section 10 FTP Server* on page 10-1 for details on the FTP server.

4-4 NTP Settings Display



Setting	Description	Default
NTP server clock information	Set whether to obtain clock information from the NTP server to update the clock in the CPU Unit.	Do not get
Port No. *1 *2	Set the port number to use to connect to the NTP server to obtain clock information. It is normally not necessary to change this setting.	123
Server specifying method*1	Set the method to use to specify the NTP server to obtain clock information. • IP address • Host name	IP address
IP address	Set the IP address of the NTP server. Specify this setting if the **server specifying method** is set to the **IP address** Option.	None
Host name	Set the host name of the NTP server (i.e., the domain name of the host). Specify this setting if the **server specifying method** is set to the **Host name** Option. (You can use up to 200 single-byte alphanumeric characters, dots, and hyphens, including up to 63 single-byte alphanumeric characters between dots.)	None
NTP operation timing*1	Set the time at which the NTP server is accessed to synchronize the clocks. • Specify a time • Specify a time interval	Specify a time
Time [hours:minutes:seconds]	The NTP server is accessed at the specified time. (Setting range: 00:00:00 to 23:59:59) Specify this setting if the **NTP operation timing** is set to the **Specify a time** Option.	00:00:00
Interval [minutes]	The NTP server is accessed when the specified period of time has passed. (Setting range: 1 to 1,440 minutes) Specify this setting if the **NTP operation timing** is set to the **Specify a time interval** Option.	60 minutes

Setting		Description	Default
	Timeout time (seconds)*1	Set the timeout detection time. (Setting range: 1 to 255 seconds) If the remote host does not respond, retry processing is performed four times within the time interval that is set here. If the **Specify a time interval** Option is selected for the **NTP operation timing**, timing for the next execution of the NTP operation starts when the fourth retry processing times out.	10 seconds

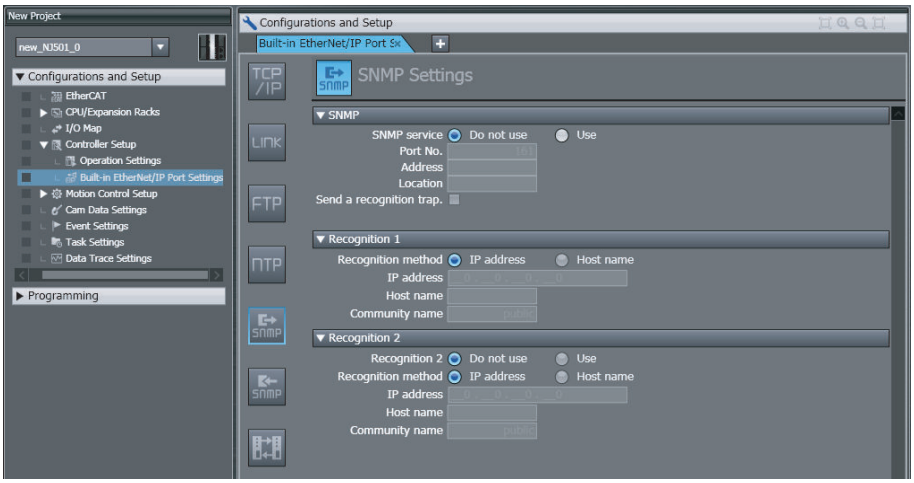
*1. This setting is required when the **Get** Option is selected for the **NTP server clock information**.

*2. The following ports are used by the system and cannot be set by the user: 25, 53, 68, 110, 2222, 2223, 2224, 9600, and 44818.

Additional Information

Refer to *Section 12 Automatic Clock Adjustment* on page 12-1 for details on obtaining clock information from the NTP server.

4-5 SNMP Settings Display



● SNMP

Setting		Description	Default
SNMP service		Specify whether to use the SNMP monitor service.*1 If the **Do not use** Option is selected, an SNMP manager cannot connect from an external device.	Do not use
	Port No.*2	Set the port number to use to connect to the SNMP server that is used to connect from an SNMP manager. This setting does not normally need to be changed.	161
	Address	Set the communications device administrator's name and installation location as text information. You do not necessarily have to input all items. This information is read by the SNMP manager. (You can input up to 255 single-byte alphanumeric characters for each item.)	None
	Location		None
	Send a recognition trap	Set whether to send an authentication trap. If you select **Send a recognition trap** and there is access from an SNMP manager that is not set in Recognition 1 or Recognition 2, an authentication trap is sent to the SNMP manager. If you select **Send a recognition trap**, specify the SNMP trap settings on the **SNMP Trap** Tab.	Not selected

*1. If you select the Use Option for the SNMP service, you also have to set Recognition 1 and 2 as described below.

*2. The following ports are used by the system and cannot be set by the user: 25, 53, 68, 110, 2222, 2223, 2224, 9600, and 44818.

Additional Information

Refer to *Section 13 SNMP Agent* on page 13-1 for details on the SNMP service.

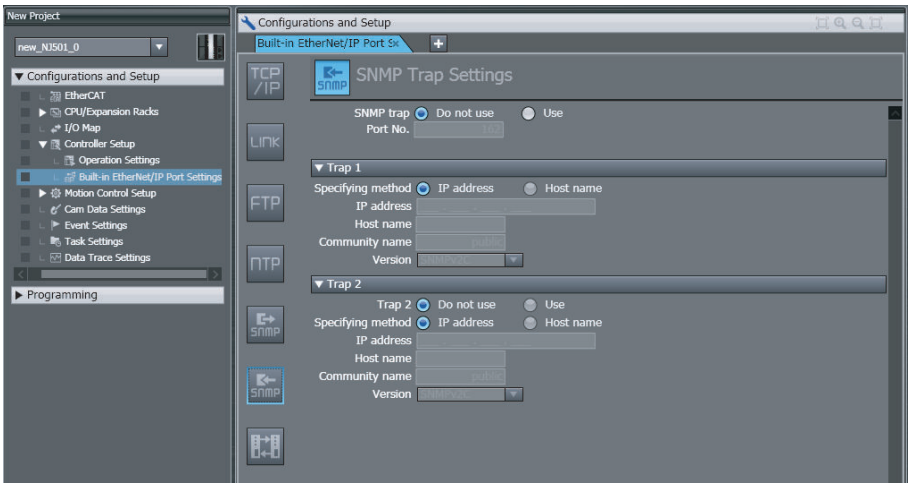
● Recognition 1

Setting	Description	Default
Recognition method	Set the method to use to specify SNMP managers for which access is permitted. • IP address • Host name Make these settings to permit access by only certain SNMP managers. Access is not allowed unless an IP address or host name is set.	IP address
IP address	Set the IP address of the SNMP manager. If the default setting of 0.0.0.0 is used, access by all SNMP managers is permitted. (Set this setting if **Recognition method** in Recognition 1 is set to the **IP address** Option.)	None
Host name	Set the host name of the SNMP manager. (Set this setting if **Recognition method** in Recognition 1 is set to the **Host name** Option.) (You can use up to 200 single-byte alphanumeric characters, dots, and hyphens with up to 63 single-byte alphanumeric characters between dots.)	None
Community name	Set the community name to enable the SNMP manager to access information from the built-in EtherNet/IP port. (Single-byte alphanumeric characters, dots, and hyphens: 255 characters max.)	public

● Recognition 2

Setting		Description	Default
Recognition 2		Specify whether to use the recognition 2 settings. • Use • Do not use	Do not use
	Recognition method	Set the method to use to specify SNMP managers for which access is permitted. • IP address • Host name Make these settings to permit access by only certain SNMP managers. Access is not allowed unless an IP address or host name is set.	IP address
	IP address	Set the IP address of the SNMP manager. If the default setting of 0.0.0.0 is used, access by all SNMP managers is permitted. (Set this setting if **Recognition method** in Recognition 2 is set to the **IP address** Option.)	None
	Host name	Set the host name of the SNMP manager. (Set this setting if **Recognition method** in Recognition 2 is set to the **Host name** Option.) (You can use up to 200 single-byte alphanumeric characters, dots, and hyphens for a host name, including up to 63 single-byte alphanumeric characters between two dots.)	None
	Community name	Set the community name to enable the SNMP manager to access information from the built-in EtherNet/IP port. (Single-byte alphanumeric characters, dots, and hyphens: 255 characters max.)	public

4-6 SNMP Trap Settings Display



● SNMP Trap

Setting	Description	Default
SNMP trap	Specify whether to use the SNMP trap (network error detection).*1 If the **Do not use** Option is selected for SNMP trap, SNMP traps are not sent to the SNMP manager	Do not use
Port No.*2	Set the port number to use to connect to the SNMP server. It is normally not necessary to change this setting.	162

*1. If you specify to use the SNMP trap, you also have to set Trap 1 and Trap 2 as described below.

*2. The following ports are used by the system and cannot be set by the user: 25, 53, 68, 110, 2222, 2223, 2224, 9600, and 44818.

Additional Information

Refer to *13-1-1 Overview* on page 13-2 for details on the SNMP trap.

● Trap 1

If the **Use** Option is selected for **SNMP trap**, you need to make the following settings.

Setting	Description	Default
Specifying method	Set the specifying method for the SNMP manager destination for SNMP traps. • IP address • Host name	IP address
IP address	Set the IP address of the SNMP manager. (Set this setting if the **Specifying method** in the Trap 1 settings is set to the **IP address** Option.)	None
Host name	Set the host name of the SNMP manager. (Set this setting if the **Specifying method** in the Trap 1 settings is set to the **Host name** Option.) (Single-byte alphanumeric characters, dots, and hyphens: 200 characters max. with up to 63 single-byte alphanumeric characters between dots.)	None

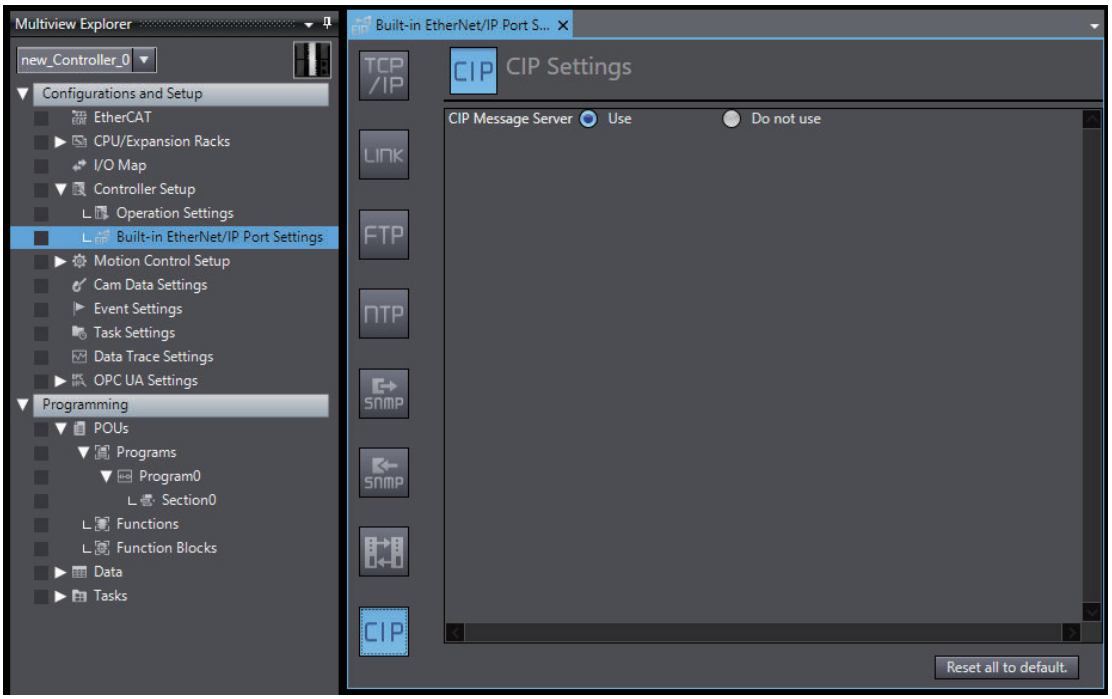
Setting	Description	Default
Community name	Set the community name. (You can use up to 255 single-byte alphanumeric characters.)	public
Version	Set the version of the SNMP manager. • SNMPv1 • SNMPv2C	SNMPv1

Trap 2

If the **Use** Option is selected for **SNMP trap**, you need to make the following settings.

Setting		Description	Default
Trap 2		Specify whether to use the Tap 2 settings. • Use • Do not use	Do not use
	Specifying method	Set the specifying method for the SNMP manager destination for SNMP traps. • IP address • Host name	IP address
	IP address	Set the IP address of the SNMP manager. (Set this setting if the **Specifying method** in the Trap 2 settings is set to the **IP address** Option.)	None
	Host name	Set the host name of the SNMP manager. (Set this setting if the **Specifying method** in the Trap 2 settings is set to the **Host name** Option.) (Single-byte alphanumeric characters, dots, and hyphens: 200 characters max. with up to 63 single-byte alphanumeric characters between dots.)	None
	Community name	Set the community name. (You can use up to 255 single-byte alphanumeric characters.)	public
	Version	Set the version of the SNMP manager. • SNMPv1 • SNMPv2C	SNMPv1

4-7 CIP Settings Display



● CIP Message Server

Setting	Description	Default
CIP Message Server	Specify whether to use the CIP message server or not. If the **Use** Option is selected, the following ports will be opened. • UDP 2222 • UDP 44818 • TCP 44818	Use

Refer to *7-3 CIP Communication Server Function* on page 7-39 for restrictions when the **Do not use** Option is selected for CIP message server.

5

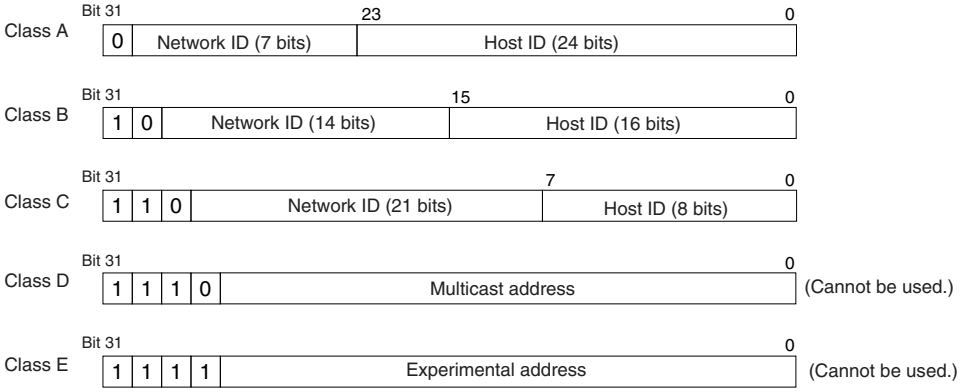
TCP/IP function

5-1 Determining IP Addresses

5-1-1 IP Addresses

IP Address Configuration

IP addresses are made up of 32 bits of binary data that specify the network number (net ID) and host number (host ID). The net ID is an address used for identifying a network. The host ID is an address used for identifying a host (node).
IP addresses are divided into three classes, A, B, and C, so that the address system can be selected according to the scale of the network. (Classes D and E are not used.)



The number of networks in each class and the number of hosts possible on the network differ according to the class.

Class	Number of networks	Number of hosts
Class A	Small	224−2 max. (16,777,214 max.)
Class B	Medium	216−2 max. (65,534 max.)
Class C	Large	28−2 max. (254 max.)

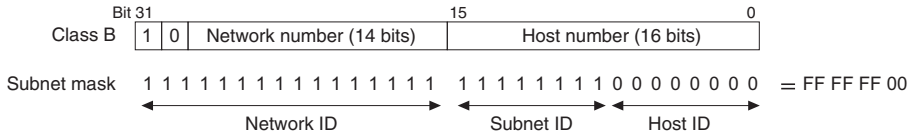
The 32 bits of binary data in an IP address are divided into four sections of eight bits each. IP addresses are represented by the decimal equivalent of each of the four octets in the 32-bit address, each separated by a period.
For example, the binary address 10000010 00111010 00010001 00100000 would be represented as 130.58.17.32.

Allocating IP Addresses

You must assign IP addresses nodes so that each IP address is assigned only once in the network or between several networks.

Subnet Mask

Operation and management of a network can become very difficult if too many nodes are connected on a single network. In such a case it can be helpful to configure the system so that a single network is divided up into several subnetworks. Internally the network can be treated as a number of subnetworks, but from the outside it acts as a single network and uses only a single network ID.
To establish subnetworks, the host ID in the IP address is divided into a subnet ID and a host ID by using a setting called the subnet mask.
The subnet mask indicates which part of the host ID is to be used as the subnet ID. All bits in the subnet mask that correspond to the bits in the IP address used either as the network ID or subnet ID are set to "1", and the remaining bits, which correspond to the bits in the IP address actually used for the host ID, are set to "0".
The following example shows the subnet mask for an 8-bit subnet ID used in the class-B IP address.



Set the same subnet mask for all of the nodes on the subnetwork. The built-in EtherNet/IP port supports CIDR (Classless Inter-Domain Routing). The subnet mask can be set to 192.0.0.0 to 255.255.255.252.
If subnetworks are not used, set the following subnet mask values for IP address classes A to C.

Class	Subnet mask
Class A	255.0.0.0
Class B	255.255.0.0
Class C	255.255.255.0

A network address is information derived from a subnet mask and used to identify each network. A network address enables users to determine whether multiple nodes belong to the same network. A network address is calculated by performing a logical AND operation on the IP address and subnet mask of a node.
The following are examples of network address calculation.
In this example, the IP address of node 1 is set to *192.168.250.20*, the IP address of node 2 is set to *192.168.245.30*, and the subnet mask is set to *255.255.240.0*. The network addresses of the two nodes are calculated as follows.

- Calculating network address of node 1

Item	Decimal notation	Binary notation
IP address	192.168.250.20	11000000.10101000.11111010.00010100
Subnet Mask	255.255.240.0	11111111.11111111.11110000.00000000
Network address	192.168.240.0	11000000.10101000.11110000.00000000

- Calculating network address of node 2

Item	Decimal notation	Binary notation
IP address	192.168.245.30	11000000.10101000.11111010.00010100
Subnet Mask	255.255.240.0	11111111.11111111.11110000.00000000
Network address	192.168.240.0	11000000.10101000.11110000.00000000

As shown in the above tables, node 1 and node 2 have the same network address, which means these nodes belong to the same network.

CIDR

CIDR, or classless interdomain routing, is used to assign IP addresses that do not use classes.
IP addresses that use classes are separated into blocks according to network IDs and host IDs, resulting in inefficient usage of IP address space.
CIDR does not use classes, so IP address space can be divided as required to more efficiently use IP address space.
For example, using a subnet mask setting with CIDR enables building a horizontally distributed network exceeding 254 nodes even if a class C address block (e.g., 192, 168...) is used.

Subnet Mask Range
192.0.0.0 to 255.255.255.252

5-1-2 Built-in EtherNet/IP Port IP Address Settings

Determining IP Addresses

Use one of the following methods to set an IP address of a built-in EtherNet/IP port.

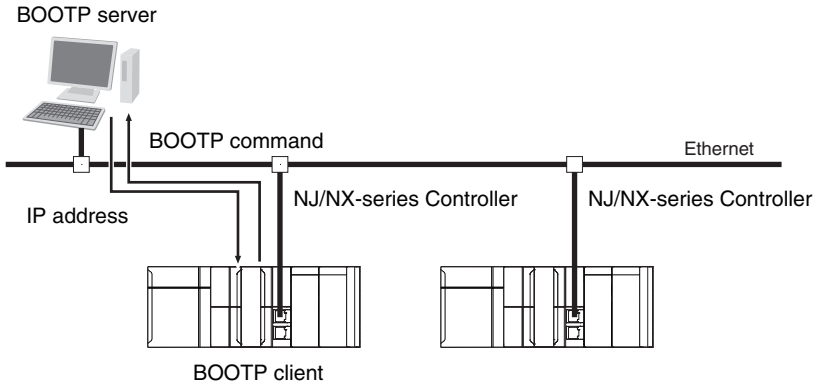
● Setting a User-specified IP Address

If you need to change the default IP address of the built-in EtherNet/IP port or if you need to use the built-in EtherNet/IP port with another EtherNet/IP node, set the IP address to a required value.
For NX701 and NX102 CPU Units, you cannot set IP addresses that make two built-in EtherNet/IP ports belong to the same network.

● Automatically Obtaining an IP Address from the BOOTP Server

There are two methods to automatically obtain an IP address.

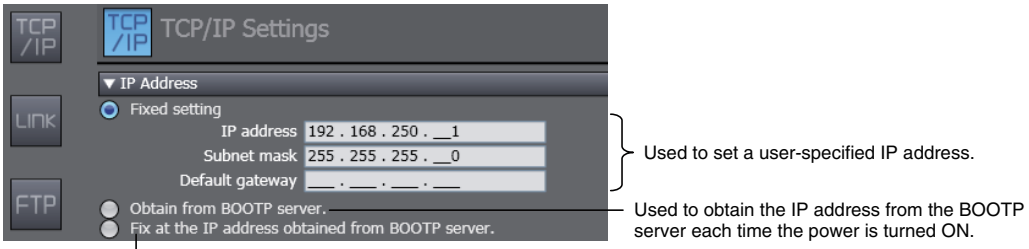
- Obtain an IP address from the BOOTP server each time the power is turned ON.
- Obtain an IP address from the BOOTP server at initial power on and set the address as a fixed IP address.



Setting IP Addresses

Use the Sysmac Studio to set an IP address of the built-in EtherNet/IP port.

1 Select a method for setting the IP address.
Make the following settings on the **TCP/IP Settings** Display of the Built-in EtherNet/IP Port Settings Tab Page in the Controller Setup to set the local IP address.



Display when using the NJ-series CPU Unit

For an NX701 CPU Unit and an NX102 CPU Unit, the IP addresses must be set separately for built-in EtherNet/IP ports 1 and 2.

Precautions for Correct Use

For an NX701 CPU Unit and an NX102 CPU Unit, you cannot set IP addresses that make two built-in EtherNet/IP ports belong to the same network.

2 Connect the Sysmac Studio to the NJ/NX-series CPU Unit via a USB connection or the Ethernet network.

Precautions for Correct Use

The NX102 CPU Unit and NX1P2 CPU Unit can be connected only via Ethernet.

3 Connect the Sysmac Studio online to the NJ/NX-series CPU Unit.
Refer to *Online Connection* on page 5-7 for the procedure to connect online.

4 Use one of the following methods to download the IP address that was set on the Sysmac Studio to the NJ/NX-series CPU Unit.

1) Go online with the Controller, and then select **Synchronization** from the **Controller** Menu. The data on the computer and the data in the physical Controller are compared to each other automatically.
2) Click the **Transfer to Controller** Button.

Note Use the "synchronization" of the Sysmac Studio to upload and download data.

5 After the IP address settings are downloaded, the IP address is reflected in the CPU Unit as follows:

- **Setting a User-specified IP Address**
  After the IP address settings are downloaded, the set IP address is automatically reflected in the CPU Unit.
- **Obtaining the IP Address from the BOOTP Server Each Time the Power Is Turned ON**

After the IP address settings are downloaded, the IP address from the BOOTP server is automatically reflected in the CPU Unit.
Each time the power supply is turned ON, the IP address from the BOOTP server is automatically reflected in the CPU Unit.

Additional Information

If you cannot obtain the IP address from the BOOTP server or the obtained IP address is not correct, select the **Fixed setting** Option in the **IP Address** Area and manually set the IP address, subnet mask, and default gateway.
Requests to the BOOTP server for an IP address will continue if connecting to the BOOTP server fails.

- **Obtaining the IP Address from the BOOTP Server Once When the Power Is Turned ON and Then Not Allow It to Change**

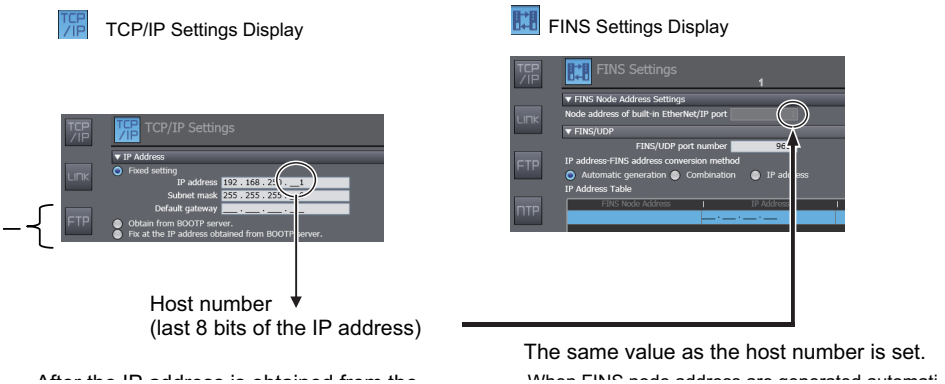
  After the IP address settings are downloaded, the IP address from the BOOTP server is automatically reflected in the Controller and set for **Fixed setting**.

Additional Information

- The **TCP/IP Settings** Display is not updated even if the IP address is obtained normally from the BOOTP server.
  To check the IP address that was obtained from the BOOTP server, upload the project from the NJ/NX-series Controller and check the Controller Status Pane.
- If you cannot obtain the IP address from the BOOTP server, the **Fix at the IP address obtained from BOOTP server** Option is selected on the **TCP/IP Settings** Display.
  To stop obtaining the IP address from the BOOTP server, select **Fixed setting** in the **IP Address** Area and manually set the IP address, subnet mask, and default gateway.
- If the Controller power supply is turned OFF and then ON after the IP address was not normally obtained from the BOOTP server, the setting remains at **Fix at the IP address obtained from BOOTP server**.
- After you select **Fix at the IP address obtained from BOOTP server** and download the IP address from the BOOTP server, the built-in EtherNet/IP port IP address setting is automatically set to **Fixed setting**. Therefore, the IP address will not match when the program is verified on the Sysmac Studio.
- To use the Packet Filter, you must allow packets (UDP:68) used for BOOTP. Refer to *5-4-5 Settings for Devices That Access the Controller* on page 5-35 for details on the setting.

Additional Information

For an NX701-□□20 CPU Unit and an NX102 CPU Unit, when the local IP address of the built-in EtherNet/IP port is set, the FINS node address is automatically set as shown below. You can set the FINS node address only with the NX701-□□20 CPU Unit and NX102 CPU Unit.
Example: Pairing an IP Address and an FINS Node Address with the Automatic generation Method



The FINS node address is required for FINS communications (e.g., to connect to the CX-Integrator and other Support Software).
When the Automatic generation Method is selected, do not set the lower 8 bits of the IP address to 000 or 255.
The NX102 CPU Unit has two EtherNet/IP ports. The FINS node address is set according to the IP address of port 2.

Online Connection

Connect the Sysmac Studio online to the CPU Unit.

Additional Information

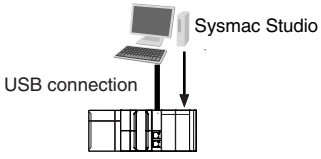
For the procedure to go online to the CPU Unit from the Sysmac Studio, refer to *Online Connections to a Controller* in the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)*.

● Types of Connection between the CPU Unit and Computer That Runs the Sysmac Studio

The CPU Unit and the computer that runs Sysmac Studio are connected via USB or Ethernet as shown below:

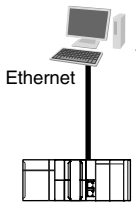
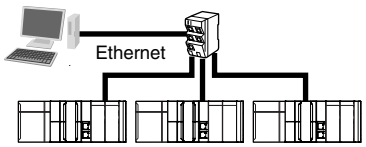
- USB Connection

Direct Connection via USB

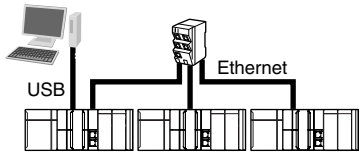


*1. NX701 CPU Units with hardware revision A or later and NX102 and NX1P2 CPU Units do not support USB connection.

- Ethernet Connection

Direct Connection via Ethernet (1:1 Connection with Auto IP)	Ethernet Connection via a Hub (1:N Connection)
	
*1. An Ethernet switch is not necessarily required. *2. You can use a straight or cross Ethernet cable to connect. *3. For NX701 and NX102 CPU Units, 1:1 connection is possible only on the built-in EtherNet/IP port 1.	*1. An Ethernet switch is required to connect. Refer to *2-1-4 Precautions for Ethernet Switch Selection* on page 2-4 for details.

- Connection from USB Across Ethernet

Remote Connection via USB (1:N Connection)

*1. An NJ/NX-series Controller with a USB port is required to connect. *2. An Ethernet switch is required to connect. Refer to *2-1-4 Precautions for Ethernet Switch Selection* on page 2-4 for details.

Precautions for Correct Use

If you connect the Sysmac Studio (computer) to the EtherNet/IP port on the CPU Unit, you cannot use direct connection via Ethernet. Use the Ethernet connection via a hub through an Ethernet switch. In that case, you must specify the destination IP address.

Additional Information

- Auto IP automatically assigns IP addresses in Windows 98 and later operating systems. Unique IP addresses are automatically assigned from the address *169.254.0.0 to 169.254.255.255*.
- If the Sysmac Studio is connected online via a built-in EtherNet/IP port, changing the IP address of the connected built-in EtherNet/IP port will cause a timeout on the Sysmac Studio. In the case, switch the Sysmac Studio status to offline, restore the original IP address of the connected built-in EtherNet/IP port, and then switch back the Sysmac Studio status to online. This will allow you to reconnect.

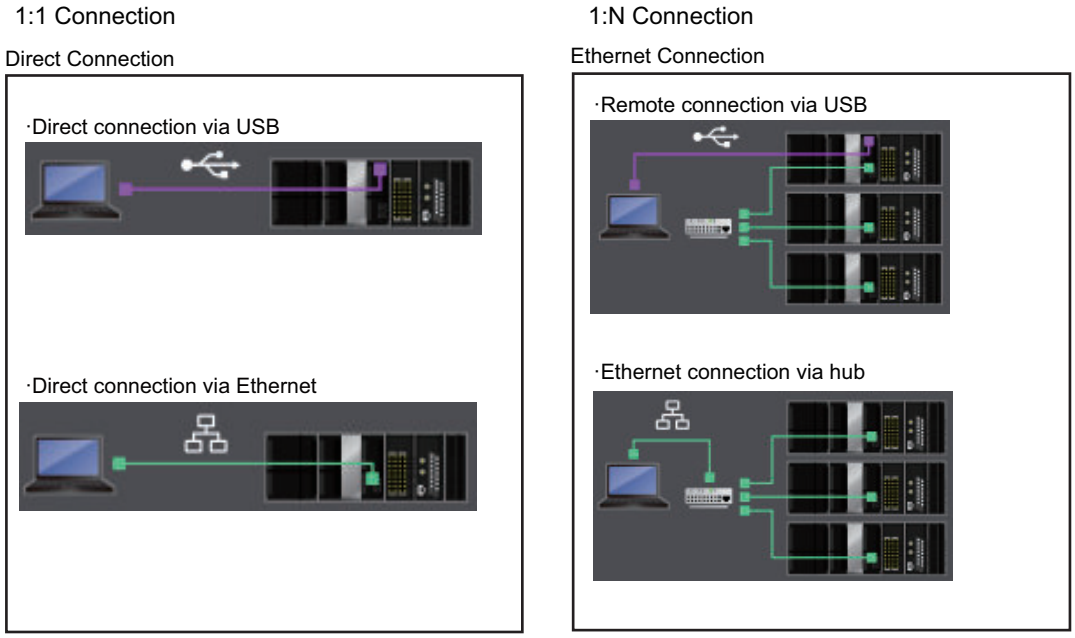
Precautions for Correct Use

If there is more than one node with the same IP address in the EtherNet/IP network, the built-in EtherNet/IP port will connect to the node that is detected first.
Note that an IP Address Duplication Error will not be detected in this case.

● Online Connection Procedure

Connect the CPU Unit and the computer that runs the Sysmac Studio via USB or Ethernet, and then perform the following procedure.

1 Select **Controller** - **Communications Setup** and click the **OK** Button in the Sysmac Studio Project Window.



Additional Information

If there is any error in the set IP address, the CPU Unit behaves as follows:

- The NET RUN indicator on the CPU Unit does not light and the NET ERR indicator flashes red. For NX701 CPU Units and NX102 CPU Units, indicators will indicate the status of each built-in EtherNet/IP port.
- An *IP Address Setting Error* is recorded in the event log.

Precautions for Correct Use

- If the IP address is duplicated or not set correctly, communications are not possible via the EtherNet/IP network. Use the Sysmac Studio to set the IP address again in direct connection via Ethernet.
- The IP address range shown below is used by the system and cannot be specified.
  169.254.0.0 to 169.254.255.255
  192.168.255.0 to 192.168.255.255
- Due to Ethernet restrictions, you cannot specify the following IP addresses.
  a) An IP address that is all 0's or all 1's
  b) IP addresses that start with 127, 0, or 255 (decimal)
  c) IP addresses that have a host ID that is all 0's or all 1's
  d) Class-D IP addresses (224.0.0.0 to 239.255.255.255)
  e) Class-E IP addresses (240.0.0.0 to 255.255.255.255)

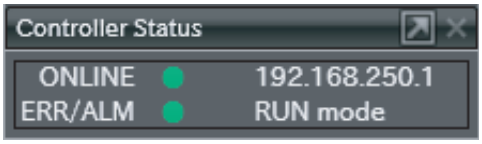
● Connecting from a Saved Project

The connection configuration that is set (via USB or EtherNet/IP) is saved in the project.
When you open a saved project on the Sysmac Studio, you can connect to the EtherNet/IP network without redoing the settings.

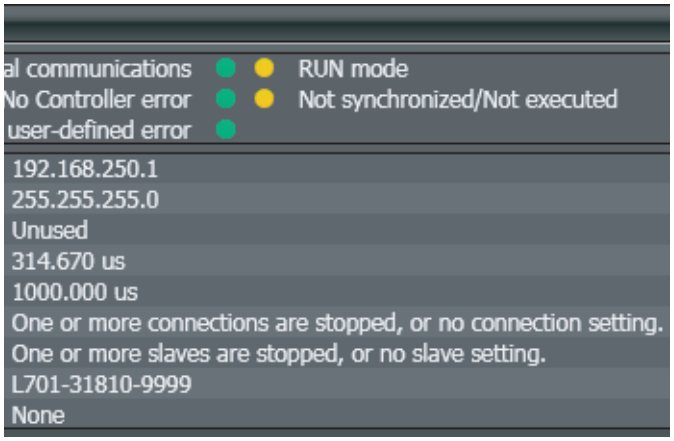
Checking the Current IP Address

The current IP address can be confirmed in the Controller Status Pane of the Sysmac Studio, whether it is manually set or obtained from the BOOTP server.
Display when using the NJ-series CPU Units and NX1P2 CPU Units

- Basic Controller Status Pane

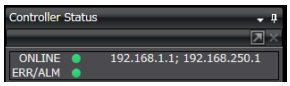


- Controller Status Pane with Details

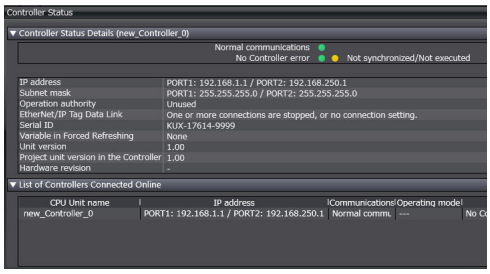


Display when using the NX701 CPU Unit and NX102 CPU Unit

- Basic Controller Status Pane



- Controller Status Pane with Details



Additional Information

- If the IP address of the built-in EtherNet/IP port is not registered due to the following reasons, the IP address field shows "0.0.0.0".
  - The IP address was not obtained from the BOOTP server.
  - The built-in EtherNet/IP port on the NX701 CPU Unit or NX102 CPU Unit is disabled. Refer to *4-1* ***TCP/IP Settings*** *Display* on page 4-2 for details on the settings for the IP address of the built-in EtherNet/IP port.

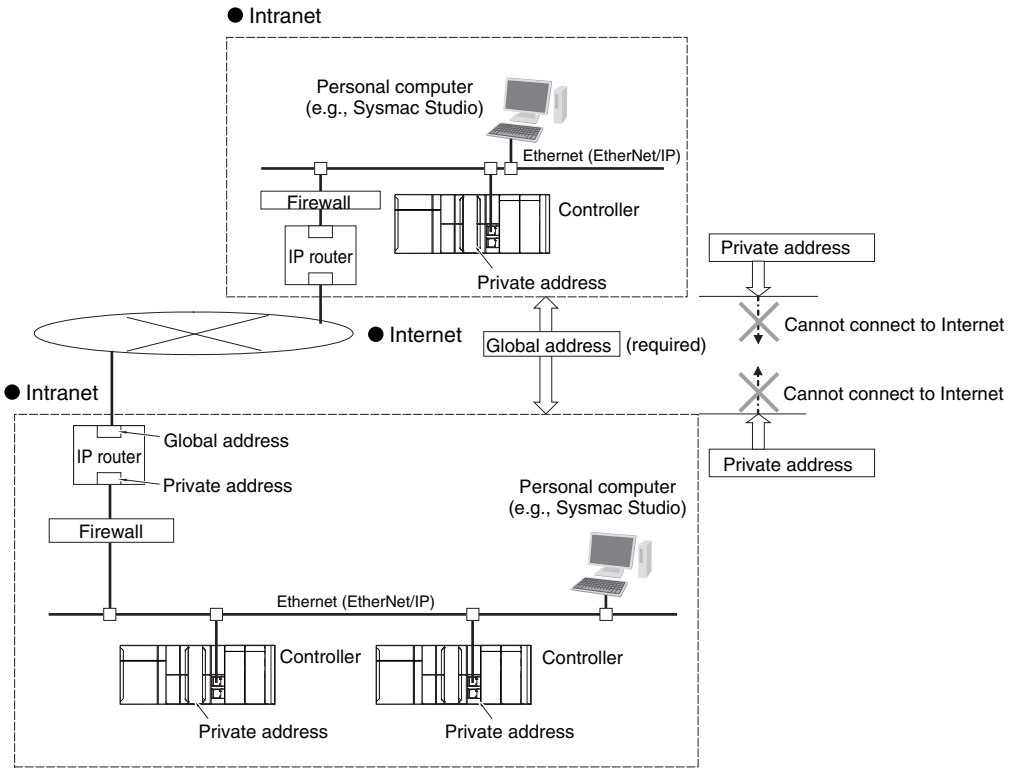
5-1-3 Private and Global Addresses

Private and Global Addresses

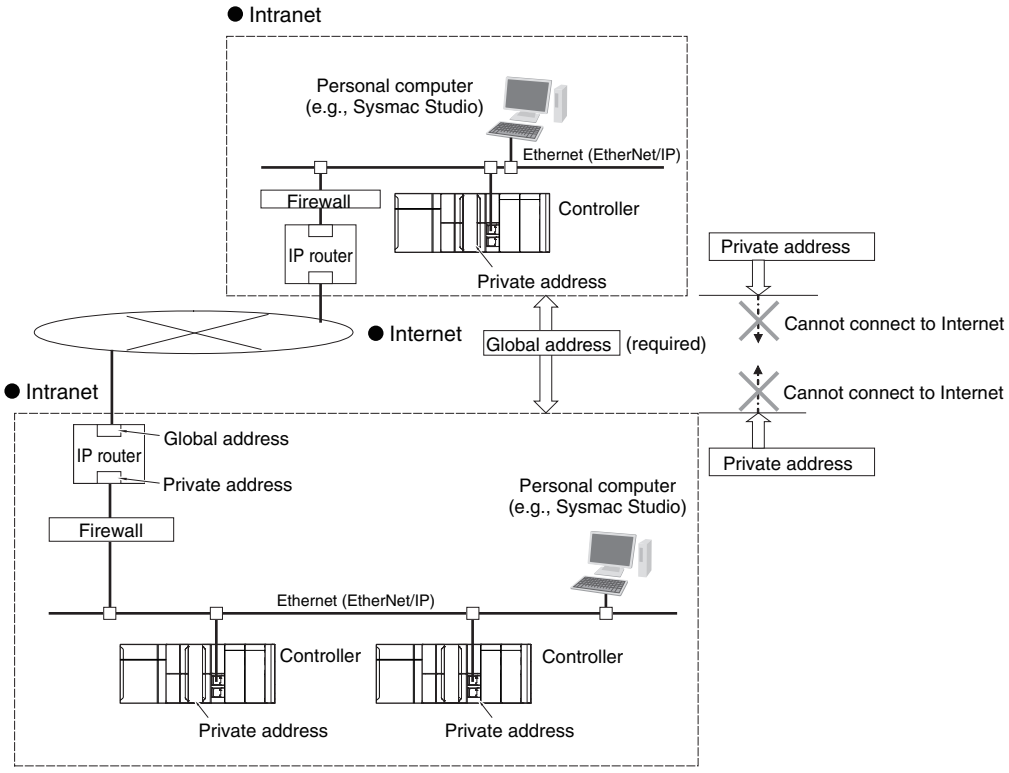
There are two kinds of IP addresses, private and global.

Global address	These are IP addresses that connect directly to the Internet. Allocated by application to NIC, each address is unique in the world, and as many as 4.3 billion can be allocated worldwide.
Private address	These are IP addresses for Intranet (LAN) use. Direct connection to the Internet is not possible. Frames that include private IP addresses are restricted by the router from being sent outside the LAN.

Generally, as shown below, global addresses in the intranet are allocated only to IP routers (such as broadband routers) interfaced with the Internet. All other nodes in the intranet, which includes the built-in EtherNet/ IP port, are allocated private addresses.



Using a Private Address for the Built-in EtherNet/IP Port



■ Conditions for Communications Applications

If the built-in EtherNet/IP port uses a private address, you can use explicit message communications service under the following conditions.

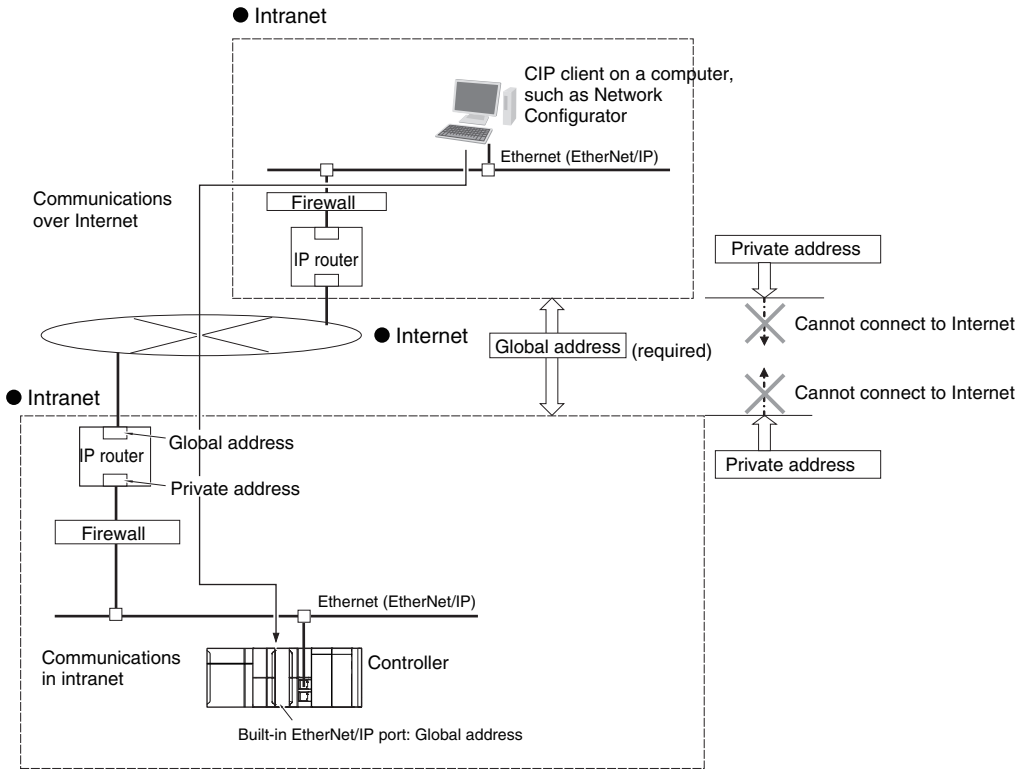
- The explicit message communications service can be executed on the intranet between built-in EtherNet/IP ports with private addresses only.
- A device such as a personal computer (CIP applications including the Network Configurator) cannot connect online and communicate over the Internet with a built-in EtherNet/IP port that has a private address.

  Explicit message communications are also not possible over the Internet between built-in EtherNet/IP ports with private addresses.

Precautions for Correct Use

- To set up an intranet through a global address involves network security considerations. Be sure to consult with a network specialist in advance and consider installation of a firewall.
- Some communication applications may not be available depending on the firewall settings made by the communications company. If there are communication applications that cannot be used, be sure to check with your communications company.
- When sending and receiving data over a global address, use secure communications, such as secure socket communications and OPC UA, that ensure confidentiality and integrity.

Using a Global Address for the Built-in EtherNet/IP Port



■ Conditions for Communications Applications

You can use the explicit message communications service over the Internet under the following conditions.

- A device such as a personal computer (a CIP application including the Network Configurator) can connect online and communicate over the Internet with a built-in EtherNet/IP port that has a global address.
- The TCP port number (44818) or UDP port number (44818) that is used for EtherNet/IP cannot be used because it is prohibited by a firewall in the communications path.

Precautions for Correct Use

- To set a global IP address for a built-in EtherNet/IP port involves network security considerations. It is recommended that the user contract with a communications company for a dedicated line, rather than for a general line such as a broadband line. Also, be sure to consult with a network specialist and consider security measures such as a firewall.
- Some communication applications may not be available depending on the firewall settings made by the communications company. If there are communication applications that cannot be used, be sure to check with your communications company.
- When sending and receiving data over a global address, use secure communications, such as secure socket communications and OPC UA, that ensure confidentiality and integrity.

5-2 TCP/ UDP Port Numbers Used for the Built-in EtherNet/IP Port

The following table shows TCP/UDP port numbers used by the built-in EtherNet/IP port, whether or not it is reserved by the system, whether or not the port number can be changed by the user, protocol used, default state, usage, and how to close the port from the default open state, for each application. TCP/UDP ports (servers) other than those shown below are not used.

Application	CPU Unit model	UDP port number	TCP port number	System reserved	Port number change	Protocol used	Default	Usage	How to close the port from open state
FTP server	All models	---	20	Reserved.	Not possible.	FTP	Close	Used when using the FTP server.	---
		---	21	---	Possible.		Close		---
SSH/ SFTP	All models	---	22	Reserved.	Not possible.	SSH/SFTP	Close	For maintenance	---
DNS client	All models	53	---	Reserved.	Not possible.	DNS	Close	Used when using the DNS client.	---
BOOTP client	All models	68	---	Reserved.	Not possible.	BOOTP	Close	Used when using the BOOTP client.	---
HTTP server	All models	---	80	Reserved.	Not possible.	HTTP	Close	Used for communications with the Sysmac Studio.	---
NTP client	All models	123	---	---	Possible.	NTP	Close	Used when using the NTP client.	---

Application	CPU Unit model	UDP port number	TCP port number	System reserved	Port number change	Protocol used	Default	Usage	How to close the port from open state
SNMP	All models	161	---	---	Possible.	SNMP (SNMPv1, SNMPv2C)	Close	Used when using the SNMP agent.	---
SNMP trap	All models	162	---	---	Possible.		Close	Used when using the SNMP trap.	---
HTTPS server	All models	---	443	Reserved.	Not possible.	HTTPS	Open	Used for communications with the Sysmac Studio.	Make one of the following settings. • Use the Packet Filter. *1 • Set the DIP switch to *Enable connections to the Sysmac Studio and NA that are not supporting secure communication*.
EtherNet/IP tag data links	All models	2222	---	Reserved.	Not possible.	CIP	Open	Used for the EtherNet/IP tag data links.	Set **Built-in EtherNet/IP Port Settings** - **CIP Settings** - **CIP Message Server** to **Do not use** on the Sysmac Studio.
	All NJ-series models	2223	---	Reserved.	Not possible.		Open		

Application	CPU Unit model	UDP port number	TCP port number	System reserved	Port number change	Protocol used	Default	Usage	How to close the port from open state
FINS/UDP	• All NJ-series models • All NX1P2 CPU Unit models • All NX102 CPU Unit models*2 • NX701-1□20*2	9600	---	---	Possible.	FINS (OMRON protocol)	Open	Used for the FINS/UDP.	Set **Built-in EtherNet/IP Port Settings** - **FINS Settings** - **FINS/UDP** to **Do not use** on the Sysmac Studio.
FINS/TCP	• All NJ-series models • All NX102 CPU Unit models*2 • NX701-1□20*2	---	9600	---	Possible.		Open	Used for the FINS/TCP.	Set **Built-in EtherNet/IP Port Settings** - **FINS Settings** - **FINS/TCP** to **Do not use** on the Sysmac Studio.
Sysmac Studio	All models	9600	---	Reserved.	Not possible.		Open	Used for communications with the Sysmac Studio.	Use the Packet Filter.*1
	CPU Units that support USB Port • All NJ-series models • All NX701 CPU Unit models*3	2224	---	Reserved.	Not possible.		Close*4		---
Maintenance	All models	---	9610	Reserved.	Not possible.	TCP (OMRON protocol)	Close	For maintenance	---
CIP messages	All models	44818	44818	Reserved.	Not possible.	CIP	Open	Used for the CIP messages.	Set **Built-in EtherNet/IP Port Settings** - **CIP Settings** - **CIP Message Server** to **Do not use** on the Sysmac Studio.
OPC UA	CPU Units that support OPC UA • NJ501-1□00 • NX102-□□□□*5 • NX701-1□□□*5	---	4840	---	Possible.	OPC UA	Close	Used when using the OPC UA.	---
TCP/UDP message service	CPU Units that support TCP/UDP message service • NX102-□□□□	64000	64000	---	Possible.	TCP/UDP	Close	Used when using the TCP/UDP message service.	---

Application	CPU Unit model	UDP port number	TCP port number	System reserved	Port number change	Protocol used	Default	Usage	How to close the port from open state
SECS/GEM connection service	CPU Units that support SECS/GEM • NJ501-1340	---	9700	Reserved.	Not possible.	TCP/UDP (OMRON protocol)	Open	Used when using the SECS/GEM connection service.	Use the Packet Filter.
		---	5000	---	Possible.	SECS-II	Close		---
DB connection service	CPU Units that support DB connection • NJ501-□□20 • NJ101-□□20 • NX102-□□20 • NX701-1□20	---	9800	Reserved.	Not possible.	TCP (OMRON protocol)	Open	Used when using the DB connection service.	Use the Packet Filter.
		---	9801	Reserved.	Not possible.		Open		
		---	9810	Reserved.	Not possible.		Open		
		---	9811	Reserved.	Not possible.		Open		

Application	CPU Unit model	UDP port number	TCP port number	System reserved	Port number change	Protocol used	Default	Usage	How to close the port from open state
Robot integrated	CPU Units that support Robot Integrated Controller • NJ501-R□□□	1989	---	Reserved.	Not possible.	UDP (OMRON protocol)	Open	Used for communications with the ACE (including Application Manager) or Sysmac Studio.	• Use the Packet Filter. • To close the ports on the left at once, remove the SD Memory Card, execute clear all memory operation, then restart the Controller.
		1992	---	Reserved.	Not possible.		Open		
		1997	1997	Reserved.	Not possible.	TCP/UDP (OMRON protocol)	Open		
		65533	---	Reserved.	Not possible.	UDP (OMRON protocol)	Open		
		65534	---	Reserved.	Not possible.		Open		
		1990	---	Reserved.	Not possible.		Close		
		1993	---	Reserved.	Not possible.		Close		
		69	---	Reserved.	Not possible.	TFTP	Close		
		---	43234	Reserved.	Not possible.	TCP (OMRON protocol)	Close		
		---	48987	Reserved.	Not possible.		Close		

*1. Closing the port may prevent communications with the Sysmac Studio. Refer to *Troubleshooting When You Cannot Go Online from the Sysmac Studio* in the *NJ/NX-series Troubleshooting Manual (Cat. No. W503)* on how to make corrections.
*2. This port number is supported only on Port 2. It cannot be used on Port 1.
*3. Only if the CPU unit has USB port.
*4. Always closed for the built-in EtherNet/IP port. Opened for the USB port only.
*5. This port number is supported only on Port 1. It cannot be used on Port 2.

Precautions for Correct Use

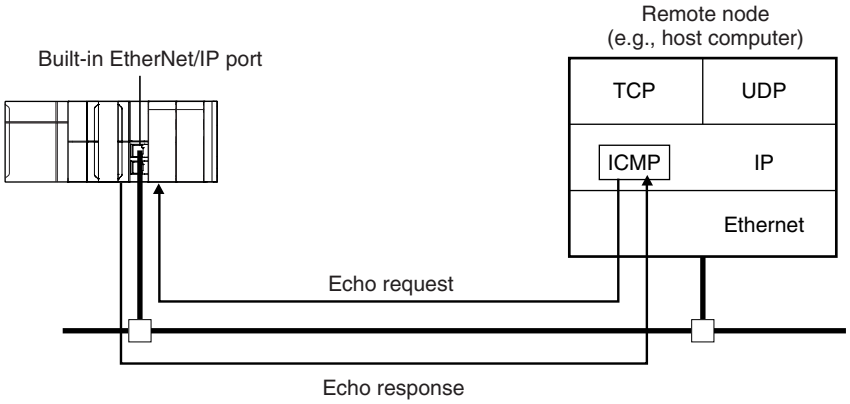
When using socket service instructions, specify the port number so that the port numbers used do not overlap. If the port numbers used are duplicated, an error will occur during instruction execution.

5-3 Testing Communications

If the basic settings (in particular the IP address and subnet mask) have been made correctly for the built-in EtherNet/IP port, then it is possible to communicate with nodes on the EtherNet/IP network. This section describes how to use the PING command to test communications with the built-in EtherNet/IP port.

5-3-1 PING Command

The PING command sends an echo request packet to a remote node and receives an echo response packet to confirm that the remote node communications are normal. The PING command uses the ICMP echo request and response. The echo response packet is automatically returned in the ICMP. The PING command is normally used to check the connections of remote nodes when you set up a network. The built-in EtherNet/IP port supports both the ICMP echo request and response functions. If the remote node returns a normal response to the PING command, then the node is physically connected correctly and Ethernet node settings are correct.



5-3-2 Using the PING Command

The built-in EtherNet/IP port automatically returns an echo response packet in response to an echo request packet sent by another node (e.g., host computer).

Precautions for Correct Use

When the **Use** Option is selected for Packet Filter of the built-in EtherNet/IP port, PING command cannot be received unless **icmp** is selected for **Protocol** of Pacekt Filter settings. For the details on the settings, refer to *Packet Filter* on page 4-7.

5-3-3 Host Computer Operation

The PING command can be executed from the host computer to send an echo request packet to a built-in EtherNet/IP port.
The following example shows how to use the PING command in the host computer.

Application Method

Input the following command at the host computer's prompt ($):

$ ping IP_address (host_name)

The destination is specified by its IP address or host name.

Additional Information

The PING command is not supported by some host computers.

Application Example

In this example, a PING command is sent to the node at IP address 130.25.36.8.
The "$" in the example represents the host computer prompt.

● Normal Execution

```
$ ping   130.25.36.8                                    ← Executes the PING command.
PING   130.25.36.8: 56   data   bytes
64   bytes   from   130.25.36.8:   icmp_seq=0.   time=0. ms
64   bytes   from   130.25.36.8:   icmp_seq=0.   time=0. ms
             :            :            :            :            :
64   bytes   from   130.25.36.8:   icmp_seq=0.   time=0. ms
                                                        ← Press the Ctrl+C Keys to cancel execution.
---- 130.25.36.8   PING   Statistics ----
9 packets   transmitted,   9 packets   received,   0% packets   loss
round-trip  (ms)     min/avg/max   = 0/1/16
$
```

● Error

```
$ ping   130.25.36.8                                    ← Executes the PING command.
PING   130.25.36.8: 56   data   bytes
                                                        ← Press the Ctrl+C Keys to cancel execution.
---- 130.25.36.8   PING   Statistics ----
9 packets   transmitted,   0 packets   received,   100% packets   loss
$
```

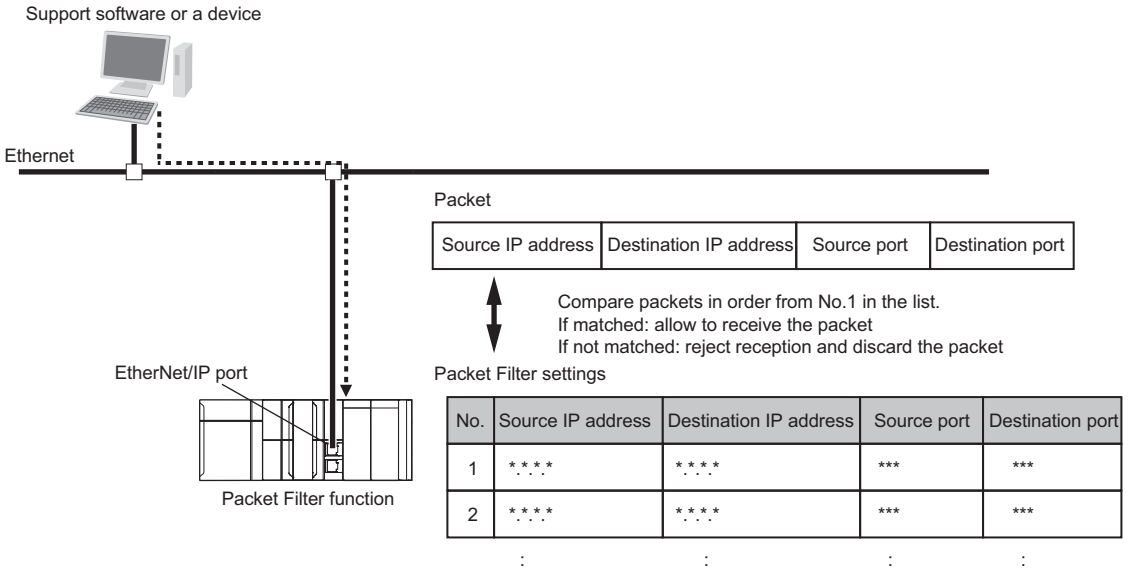
Refer to the command reference manual for your computer's OS for details on using the PING command.

5-4 Packet Filter

This section provides an overview of Packet Filter, explains the specifications, settings, and usage examples.

5-4-1 Introduction to Packet Filter

This function filters IP packets in the receive processing at the built-in EtherNet/IP ports. While Packet Filter (Simple) is used to restrict Sysmac Studio connections, Packet Filter performs general-purpose packet filtering that does not restrict communication partner to Sysmac Studio.
Packet Filter settings are configured in the permit list. If **any** is set in Packet Filter, all packets are allowed. If a value other than **any** is set in Packet Filter, the received packet is compared with Packet Filter settings. When a matching packet is received, reception is permitted. When a non-matching packet is received, reception is prohibited and the packet is discarded. Packet Filter settings include the source IP address, destination IP address, and TCP/UDP port number.



Precautions for Correct Use

- If you use an NX701 CPU Unit, NX102 CPU Unit, or NX1P2 CPU Unit and cannot go online with the Sysmac Studio because of forgetting the registered IP address, you can disable this function tentatively by starting the Unit in Safe Mode. Refer to *Troubleshooting When You Cannot Go Online from the Sysmac Studio* in the *NJ/NX-series Troubleshooting Manual (Cat. No. W503)* for details.
- If you use an NJ-series CPU Unit and cannot go online with the Sysmac Studio because of forgetting the registered IP address, connect using the USB port.
- Packet Filter supports the stateful inspection. Therefore, if the Controller is specified as a client, as in DNS, NTP, DB connection services, and communication instructions, you do not need to add the responses from other devices to Packet Filter settings. For example, if you execute the FTP client instruction of the Controller, you can receive responses from the FTP server through stateful inspection even if you have not registered the response from the FTP server in Packet Filter settings.

Version Information

Packet Filer is available in the following CPU Units of the stated versions.
- NJ-series, NX102, NX1P2 CPU Unit: Version 1.49 or later
- NX701 CPU Unit: Version 1.29 or later

5-4-2 Packet Filter Specifications

The specifications for Packet Filter are given below.

Item	Specification	Remarks
Filtering system	Permit list	The system enables reception of packets registered in Packet Filter settings and prohibits reception of unregistered packets.
Location to perform filtering	Receive processing at the built-in EtherNet/IP port (If the Controller has two built-in EtherNet/IP ports, you can configure the setting for each port.)	• No filtering is applied to the sending process of the built-in EtherNet/IP port. • Stateful inspection is supported.
Number of Packet Filter tables	32	
Settings for Packet Filter tables	• Source IP Address/Mask • Destination IP Address/Mask • Protocol (tcp, udp, igmp, icmp) If tcp or udp is selected for Protocol, specify the source port and destination port.	The IP address and TCP/UDP port can be specified as a range.

5-4-3 Packet Filter Settings

For details on Packet Filter settings, refer to *Packet Filter* on page 4-7.

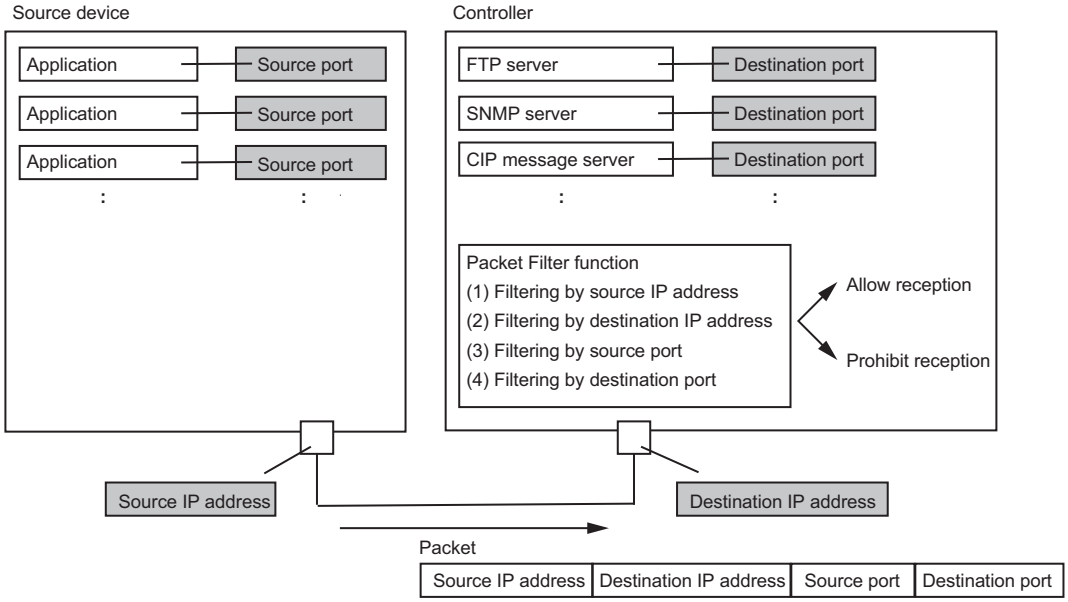
Additional Information

For set values of **Destination Port** for each communication, refer to *5-4-5 Settings for Devices That Access the Controller* on page 5-35.

5-4-4 Case Where Packet Filter is Used

Packets sent from a computer or a device to the Controller include the following four types of information.
- Source IP address
  Unique numbers assigned to each source device. This address can be used to identify the source device.
- Destination IP address
  Unique numbers assigned to each Controller that is the destination. This address can be used to identify the Controller to which the packets are sent.
- Source port
  A unique number assigned to the source application. This number can be used to identify the source application.
- Destination port

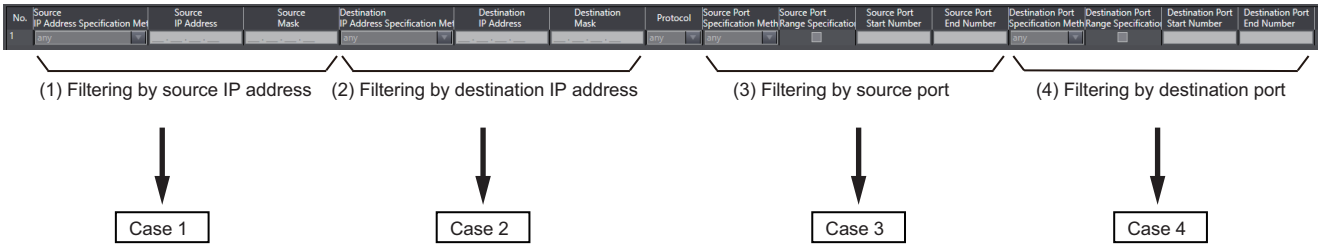
A unique number assigned to the destination application. This number can be used to identify the application to which the packets are sent.



Packet Filter function can enable or disable the reception of packets using these four types of information. If the content of the packet matches the permitted content in Packet Filter settings, reception is permitted. Otherwise, reception is disabled and the packet is discarded.
In the case where Packet Filter is used, the four types of information are related as follows.

Case Where Packet Filter is Used	Description	Reference
(1) Filtering by source IP address	Enables or disables reception of packets sent from a specific device.	*Case 1: Filtering by Source IP Address* on page 5-25
(2) Filtering by destination IP address	Only packets sent to a specific Controller are allowed to be received.	*Case 2: Filtering by Destination IP Address* on page 5-27
(3)Filtering by source port	Allow or disallow packets sent using a specific application.	*Case 3: Filtering by Source Port* on page 5-31
(4)Filtering by destination port	Allow and receive only packets sent to a specific application.	*Case 4: Filtering by Destination Port* on page 5-33

Packet Filter settings can also be set as shown below according to the case where the four types of Packet Filter are used.



The following describes usage examples and set values for each of the four types of cases.

Case 1: Filtering by Source IP Address

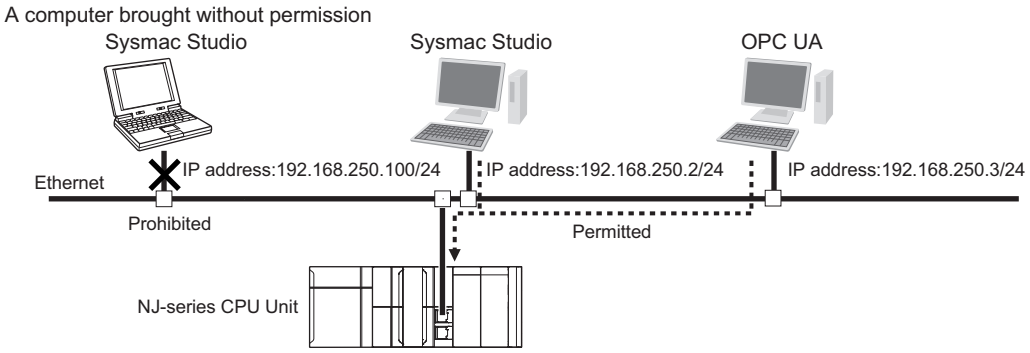
You can filter access to the Controller by source IP address. This is useful when the IP address can be used to distinguish client devices that are prohibited from communicating with client devices that are permitted to communicate. In Packet Filter's source IP address setting, set the IP address of the device that is allowed to communicate. Communications with devices whose IP addresses are not registered are prohibited.

● Application Example

An application example under the following conditions is shown below.

- Communications between the computer used in the facility and the Controller are permitted, and communications with a computer brought without permission are prohibited.
- The IP addresses of the computers that are permitted to communicate are fixed.
- The computers that are allowed to communicate have Sysmac Studio and OPC UA respectively.

The configuration of this application example is as follows.



Packet Filter settings are as follows. Enter the IP address of the computer to use Sysmac Studio in the No.1 Source **IP Address** field. Enter the IP address of the computer to use OPC UA in the No.2 Source **IP Address** field.

No.	Setting		Set value
1	Source	IP Address Specification Method	**IP address specification**
		IP Address	192.168.250.2
		Mask	255.255.255.255
	Destination	IP Address Specification Method	**any**
		IP Address	---
		Mask	---
	Protocol		**any**
	Source Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---
	Destination Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---

No.	Setting		Set value
2	Source	IP Address Specification Method	**IP address specification**
		IP Address	192.168.250.3
		Mask	255.255.255.255
	Destination	IP Address Specification Method	**any**
		IP Address	---
		Mask	---
	Protocol		**any**
	Source Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---
	Destination Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---

Additional Information

You can also mask the IP address to specify multiple devices that are allowed to communicate. The following is sample Packet Filter settings to allow communications with devices with IP addresses from 192.168.250.1 to 192.168.250.3.

No.	Setting		Set value
1	Source	IP Address Specification Method	**IP address specification**
		IP Address	192.168.250.0
		Mask	255.255.255.252
	Destination	IP Address Specification Method	**any**
		IP Address	---
		Mask	---
	Protocol		**any**
	Source Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---
	Destination Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---

Restrictions

When filtering by the source IP address is used, communication from devices that are not registered to the source IP address of Packet Filter settings is prohibited. Therefore, the IP addresses of all devices communicating with the Controller must be registered to the source IP addresses.
If the Controller cannot communicate with a device that you want to allow, make sure that the IP address of that device is correctly set to the source IP address.

Case 2: Filtering by Destination IP Address

You can filter access to the Controller by destination IP address in the packets received by the built-in EtherNet/IP port. This is useful in the following cases.

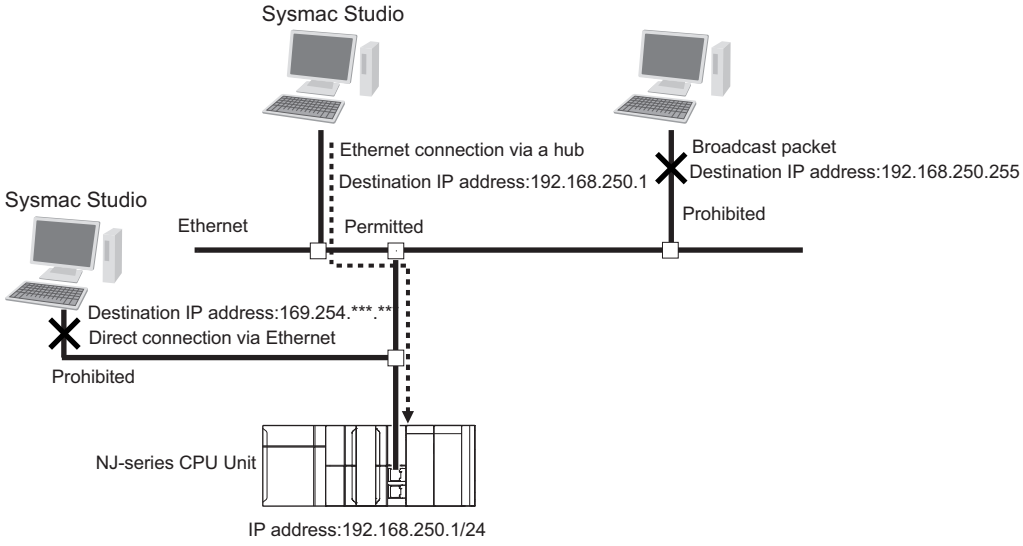
- Where you want to prohibit the receipt of broadcast packets that are unnecessary for the Controller
- Where you want to prohibit direct connection via Ethernet in the Sysmac Studio, but allow connection by a specified IP address

● Application Example 1

An application example under the following conditions is shown below.

- Reception of unnecessary broadcast packets for the Controller is prohibited.
- Connection of Sysmac Studio through **Ethernet connection via a hub** is allowed and connection through **Direct connection via Ethernet** is prohibited.

The configuration of this application example is as follows. Destination IP address for direct connection via Ethernet is 169. 254.***.***. Destination IP address of unnecessary broadcast packets for the Controller is 192.168.250.255.



Packet Filter settings are as follows. Set the IP address for the Controller's built-in EtherNet/IP port to the destination IP address.

No.	Setting		Set value
1	Source	IP Address Specification Method	**any**
		IP Address	---
		Mask	---
	Destination	IP Address Specification Method	**IP address specification**
		IP Address	192.168.250.1
		Mask	255.255.255.255
	Protocol		**any**
	Source Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---
	Destination Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---

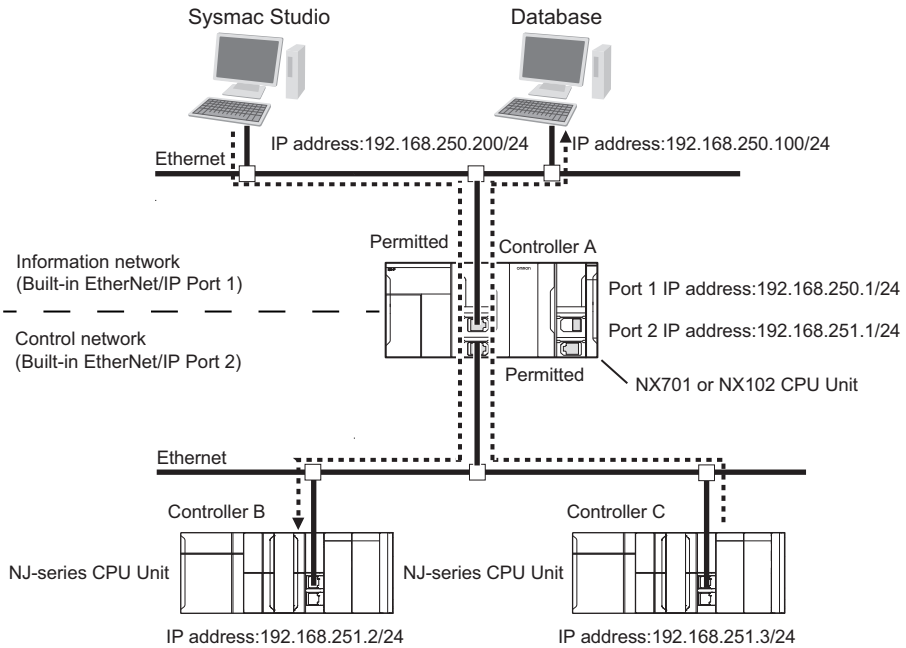
● Application Example 2

If filtering by destination IP address is enabled in a Controller between devices, it can restrict the devices that are allowed to communicate with each other .

An application example under the following conditions is shown below.

- Controller A has two built-in EtherNet/IP ports.
- Port 1 of Controller A is connected to the information network, and the computer with Sysmac Studio and the computer using Database are connected to the information network.
- Port 2 of Controller A is connected to the control network, and Controller B and Controller C are connected to the control network.
- The computer with Sysmac Studio communicates only with Controller A and Controller B. The computer using Database only communicates with Controller C.

The configuration of this application example is as follows.



Packet Filter settings of Controller A are as follows.
Enter the IP address of Controller A and Controller B to **Destination IP Address** field.
Enter the IP address of the computer using Database to Port 2 **Destination IP Address** field.

Port 1 Packet Filter Settings

No.	Setting		Set value
1	Source	IP Address Specification Method	**any**
		IP Address	---
		Mask	---
	Destination	IP Address Specification Method	**IP address specification**
		IP Address	192.168.251.2
		Mask	255.255.255.255
	Protocol		**any**
	Source Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---
	Destination Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---
2	Source	IP Address Specification Method	**any**
		IP Address	---
		Mask	---
	Destination	IP Address Specification Method	**IP address specification**
		IP Address	192.168.250.1
		Mask	255.255.255.255
	Protocol		**any**
	Source Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---
	Destination Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---

Port 2 Packet Filter Settings

No.	Setting		Set value
1	Source	IP Address Specification Method	**any**
		IP Address	---
		Mask	---
	Destination	IP Address Specification Method	**IP address specification**
		IP Address	192.168.250.100
		Mask	255.255.255.255
	Protocol		**any**
	Source Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---
	Destination Port	Specification Method	---
		Range Specification	---
		Start Number	---
		End Number	---

To route different networks, the computers, Controller B, and Controller C must be configured with a default gateway or an IP router table.

● Restrictions

When filtering by the destination IP address is used, communication to an IP address not registered in Packet Filter settings is prohibited. Therefore, all destination IP addresses of the packets that you want to allow must be set to the destination IP address in Packet Filter settings.
In addition, attention should be paid to the following.

- When you connect Sysmac Studio though **Direct connection via Ethernet**, set the **Destination IP Address** to 169.254.0.0 and the **Destination Mask** to 255.255.0.0, and allow 169.254. ***.***. Otherwise, the connection will fail.

Case 3: Filtering by Source Port

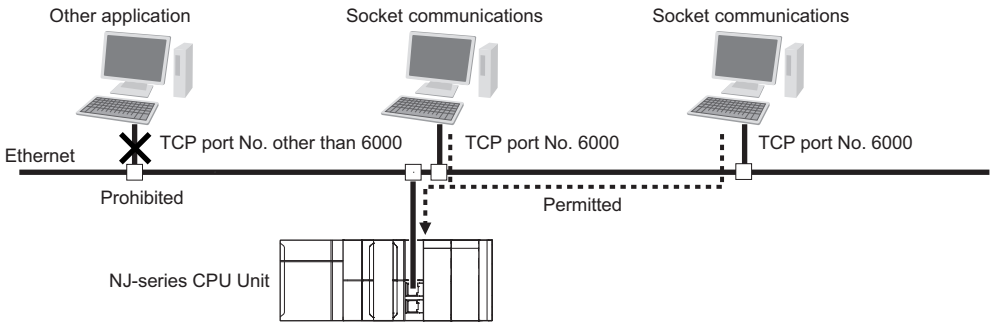
You can filter access to the Controller by the source TCP/UDP port. This is useful when the source TCP/UDP ports can be used to distinguish communications that are prohibited from communications that are permitted. In Packet Filter's source port settings, register TCP/UDP ports that are allowed to communicate. Communications with unregistered TCP/UDP ports are prohibited.

● Application Example

An application example under the following conditions is shown below.

- Communications between the computer used in the facility and the controller (source port: fixed to TCP6000) are permitted, and communications with applications that are not permitted (source port: other than TCP6000) are prohibited.
- An application running on the computer in the facility uses a socket communications program and has a fixed source port.

The configuration of this application example is as follows. The socket communications program that is allowed to communicate uses TCP port 6000.



Packet Filter settings are as follows: For Protocol, **tcp** is selected and 6000 for the source port.

No.	Setting		Set value
1	Source	IP Address Specification Method	**any**
		IP Address	---
		Mask	---
	Destination	IP Address Specification Method	**any**
		IP Address	---
		Mask	---
	Protocol		**tcp**
	Source Port	Specification Method	**Port specification**
		Range Specification	No check.
		Start Number	6000
		End Number	---
	Destination Port	Specification Method	**any**
		Range Specification	---
		Start Number	---
		End Number	---

Restrictions

If filtering by source port is used, communication from an unregistered TCP/UDP port is prohibited. Therefore, the TCP/UDP ports of all devices communicating with the Controller must be set as the source ports.

Omron's support software, such as Sysmac Studio, selects an unused port each time, so the user cannot specify the source port. Therefore, the destination port must be set according to the protocols used by the Omron's support software. For details on the destination port settings, refer to *Case 4: Filtering by Destination Port* on page 5-33.

If the Controller cannot communicate with a device that you want to allow, make sure that TCP/UDP port used by the device is set correctly to the source port.

Case 4: Filtering by Destination Port

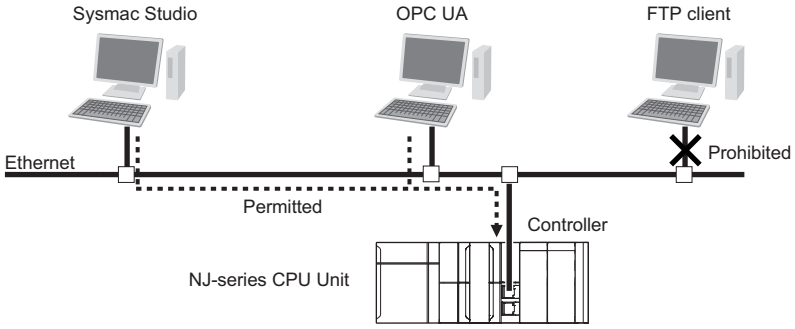
You can filter access to the Controller by destination port in the packets received by the built-in EtherNet/IP port. Because the destination port is determined for each communication protocol, this function is useful when the communication protocols used in the facility are fixed and you want to prohibit other communications protocols. Register the destination port of allowed communications in Packet Filter settings. Communications using unregistered destination ports are prohibited.

● Application Example

An application example under the following conditions is shown below.

- Communication protocols used in the facility are permitted, and the communication protocols not used in the facility are prohibited.
- Access to the Controller from sources other than Sysmac Studio and OPC UA is prohibited in the facility.

The configuration of this application example is as follows.



Packet Filter settings are as follows. When Sysmac Studio version 1.50 or higher is connected, it uses TCP port 443. OPC UA uses TCP port 4840.

Settings that allow Sysmac Studio to connect

No.	Setting		Set value
1	Source	IP Address Specification Method	**any**
		IP Address	---
		Mask	---
	Destination	IP Address Specification Method	**any**
		IP Address	---
		Mask	---
	Protocol		**tcp**
	Source Port	Specification Method	**any**
		Range Specification	---
		Start Number	---
		End Number	---
	Destination Port	Specification Method	**Port specification**
		Range Specification	No check.
		Start Number	443
		End Number	---

Settings that allow OPC UA to connect

No.	Setting		Set value
2	Source	IP Address Specification Method	**any**
		IP Address	---
		Mask	---
	Destination	IP Address Specification Method	**any**
		IP Address	---
		Mask	---
	Protocol		**tcp**
	Source Port	Specification Method	**any**
		Range Specification	---
		Start Number	---
		End Number	---
	Destination Port	Specification Method	**Port specification**
		Range Specification	No check.
		Start Number	4840
		End Number	---

Restrictions

If filtering by destination port is used, communications to an unregistered destination port are prohibited. Therefore, all destination ports used by the devices to communicate with must be registered to the destination port.

If the destination ports are not registered, the devices may time out.

If communication with a device that you want to allow fails, make sure that the destination port used by the device is set correctly to the destination port of Packet Filter.

Additional Information

Selecting the **Do not use** Option for each communications protocol closes the TCP/UDP port used for the communications protocol. This allows you to filter communications by destination port in the same way as in Case 4.

5-4-5 Settings for Devices That Access the Controller

This section shows the set values of Packet Filter for each device that accesses the Controller.

Settings for Connecting Sysmac Studio

This section describes how to configure the destination port of Packet Filter to allow connections with the Sysmac Studio.
The setting values for the destination port differ as shown below depending on the connection type and setting on enabling connections to the Sysmac Studio and NA that are not supporting secure communication.

Connection type*1	Setting on enabling connections to the Sysmac Studio and NA that are not supporting secure communication*2	Destination port settings				
		Protocol	Destination Port Specification Method	Destination Port Range Specification	Destination Port Start Number	Destination Port End Number
Direct connection via Ethernet*3	Enable	tcp	Port specification	No check.	80	---
		udp	Port specification	No check.	9600	---
		tcp*4	Port specification	No check.	44818	---
		udp*4	Port specification	No check.	44818	---
		icmp*4	---	---	---	---
	Disable	tcp	Port specification	No check.	443	---
		udp	Port specification	No check.	9600	---
Ethernet connection via a hub	Enable	tcp	Port specification	No check.	80	---
		tcp*4	Port specification	No check.	44818	---
		icmp*4	---	---	---	---
	Disable	tcp	Port specification	No check.	443	---
Remote connection via USB*5	Enable	tcp	Port specification	No check.	80	---
		tcp	Port specification	No check.	44818	---
		udp*6	Port specification	No check.	44818	---
	Disable	tcp	Port specification	No check.	443	---
		tcp	Port specification	No check.	44818	---
		udp*6	Port specification	No check.	44818	---

*1. For this setting, select **Communications Setup** from the **Controller** Menu, and select **Connection type** on the Sysmac Studio.

*2. Set with the DIP switch. Refer to *Troubleshooting When You Cannot Go Online from the Sysmac Studio* in the *NJ/NX-series Troubleshooting Manual (Cat. No. W503)* for details.

*3. For direct connection via Ethernet, the source and destination IP addresses must be set to 169.254. ***.***. When filtering by IP address is used, set 169.254.0.0 to IP address and 255.255.0.0 to mask to allow this address.

*4. This setting is required only when EtherNet/IP connection settings are made in the Sysmac Studio. This setting is not required if no EtherNet/IP connection settings are made.

*5. For remote connection via USB, specify the IP address of the relayed controller as the source IP address. When filtering by IP address is used, allow this address.

*6. This packet is sent by local broadcast. Allow this address if filtering by destination IP address is enabled. For example, if the Controller's IP address is 192.168.250.1/24, specify 192.168.250.255 to the destination IP address.

Additional Information

- To use the **Ethernet Communications Test**, which can be started by selecting **Controller** - **Communications Setup** on the Sysmac Studio in the environment where Ethernet direct connection is made, the following settings are required. Since this packet is sent by local broadcast, allow this address when filtering by the destination IP address is used. For example, if the controller IP address is 192.168.250.1/24, specify 192.168.250.255 to the destination IP address.

Protocol	Destination Port Specification Method	Destination Port Range Specification	Destination Port Start Number	Destination Port End Number
udp	Port specification	No check.	9600	---

- When the **Use** Option is selected for Packet Filter of the relayed Controller in the environment where remote connection is made via USB, the following settings are required. In this case, this packet has the connected Controller's IP address as the source IP address and the relayed Controller's IP address as the destination IP address. When filtering by IP address is used, allow these addresses.

Protocol	Destination Port Specification Method	Destination Port Range Specification	Destination Port Start Number	Destination Port End Number
udp	Port specification	No check.	44818	---

Settings for Connecting Support Software Other Than Sysmac Studio

The settings for connecting the support software other than Sysmac Studio are as follows.

Support software	Connection type	Destination port settings				
		Protocol	Destination Port Specification Method	Destination Port Range Specification	Destination Port Start Number	Destination Port End Number
Network Configurator for EtherNet/IP	Either **Ethernet I/F** or **NJ/NX/NY Series Ethernet Direct I/F**	tcp*1	Port specification	No check.	44818	---
		udp*2	Port specification	No check.	44818	---
		icmp*1	---	---	---	---

Support software	Connection type	Destination port settings				
		Protocol	Destination Port Specification Method	Destination Port Range Specification	Destination Port Start Number	Destination Port End Number
CX-ConfiguratorFDT	Either **Ethernet I/F** or **NJ/NX/NY Series Ethernet Direct I/F** (Communication DTM: OMRON EtherNet/IP)	Same settings as for Network Configurator.				
	Any of **Direct connection via Ethernet**, **Ethernet connection via a hub**, or **Remote connection via USB** (Communication DTM: Nx built-in EtherCAT or NX CPU Unit Bus)	Same settings as for Sysmac Studio. Refer to *Settings for Connecting Sysmac Studio* on page 5-35 for settings for the Sysmac Studio. The setting value differs depending on the version of CX-ConfiguratorFDT. For CX-ConfiguratorFDT version 2.57 or higher, set tcp: 443 for the destination port in Packet Filter settings. For CX-ConfiguratorFDT version 2.56 or lower, set tcp: 80 for the destination port in Packet Filter settings.				
CX-Integrator, CX-Protocol	**Direct connection via Ethernet**[*3] (Network type: Ethernet (FINS/TCP))	tcp	Port specification	No check.	9600	---
		udp	Port specification	No check.	9600	---
	Ethernet connection via a hub (Network type: Ethernet (FINS/TCP))	tcp	Port specification	No check.	9600	---
	Ethernet connection via a hub (Network type: Ethernet)	udp	Port specification	No check.	9600	---
CNC Operator	---	tcp	any[*4]	---	---	---
		icmp	---	---	---	---
SECS/GEM Configurator	---	tcp	any[*4]	---	---	---
		icmp	---	---	---	---

Support software	Connection type	Destination port settings				
		Protocol	Destination Port Specification Method	Destination Port Range Specification	Destination Port Start Number	Destination Port End Number
Sysmac Conrtoller Log Upload Tool	Any of **Direct connection via Ethernet**, **Ethernet connection via a Hub**, or **Remote connection via USB**	Same settings as for Sysmac Studio. Refer to *Settings for Connecting Sysmac Studio* on page 5-35 for settings for the Sysmac Studio. The setting value differs depending on the version of Sysmac Studio that is installed. For Sysmac Studio version 1.50 or higher, allow tcp: 443 for the destination port. For Sysmac Studio version 1.49 or lower, allow tcp: 80 for the destination port.				

*1. For **NJ/NX/NY Series Ethernet Direct I/F** connection, specify 169.254.***.*** for the source IP address and destination IP address. When filtering by IP address is used, set 169.254.0.0 to IP address and 255.255.0.0 to mask to allow this address.

*2. When filtering by IP address is used, allow the following IP addresses.
- NJ/NX/NY Series Ethernet Direct I/F: allow the following two addresses
  a) Source IP address: Controller's IP address, Destination IP address: Local broadcast to the Controller's network (When the controller's IP address is 192.168.250.1/24, allow 192.168.250.255.)
  b) Source IP address :169.254.***. ***, Destination IP address :169.254.***. *** (IP address 169.254.***.*** is allowed by setting 169.254.0.0 to the IP address and 255.255.0.0 to the mask.)
- Ethernet I/F Connection
  a) Source IP address: Computer's IP address, Destination IP address: Local broadcast to the computer's network (When the computer's IP address is 192.168.250.100/24, allow 192.168.250.255.)

*3. For **Direct connection via Ethernet**, the source and destination IP addresses must be set to 169.254.***.***. When filtering by IP address is used, set 169.254.0.0 to IP address and 255.255.0.0 to mask to allow this address.

*4. This is selected to connect in FTP passive mode. Because the port used for data connection is not uniquely determined, **any** must be selected for specification method.

Settings for Connecting a Programmable Terminal

The settings for connecting Programmable Terminals are as follows.

Programmable Terminal	Destination port settings				
	Protocol	Destination Port Specification Method	Destination Port Range Specification	Destination Port Start Number	Destination Port End Number
NA-series	tcp	Port specification	No check.	80 or 443*1	---
NS-series	tcp	Port specification	No check.	80	---
	tcp	Port specification	No check.	44818	---
NB-series	udp	Port specification	No check.	9600	---

*1. For NA Runtime version 1.161 and NA5 system program version 10.0.0 or higher, set the destination port start number to 443.

Settings for Each Communications Protocol

The settings for each communications protocol are as follows.

Communications protocol	Destination port settings				
	Protocol	Destination Port Specification Method	Destination Port Range Specification	Destination Port Start Number	Destination Port End Number
BOOTP client	udp	Port specification	No check.	68	---
OPC UA server	tcp	Port specification	No check.	4840*1	---
FINS/TCP server	tcp	Port specification	No check.	9600*1	---
FINS/UDP server	udp	Port specification	No check.	9600*1	---
SNMP agent	udp	Port specification	Checked.	161*1	162*1
FTP server*2 In Active mode	tcp	Port specification	Checked.	20	21*1
FTP server*2 In Passive mode	tcp	any*3	---	---	---
TCP/UDP message service	udp	Port specification	No check.	64000*1	---
	tcp	Port specification	No check.	64000*1	---

*1. If the port number has been changed, the new port number must be set.
*2. If the Controllers are FTP-clients, no Packet Filter settings are required.
*3. Because the port used for data connection is not uniquely determined, **any** must be selected for specification method.

Settings for Using EtherNet/IP Communications

Make the following settings to use EtherNet/IP communications.

Communications	Communications protocol	Condition	Destination port settings				
			Protocol	Destination Port Specification Method	Destination Port Range Specification	Destination Port Start Number	Destination Port End Number
CIP messages	UCMM	Server	tcp	Port specification	No check.	44818	---
			icmp*1	---	---	---	---
	Class3	Server	tcp	Port specification	No check.	44818	---
			icmp*1	---	---	---	---
Tag data links	Class1	Originator	igmp*2	---	---	---	---
		Target	tcp	Port specification	No check.	44818	---
			icmp*3	---	---	---	---
CIP Safety communications	Class0	Originator	igmp*2	---	---	---	---
		Target	tcp	Port specification	No check.	44818	---

*1. Select this if CX-Compolet/SYSMAC Gateway is a client.
*2. Select this for Multicast.
*3. Select this when SYSMAC Gateway is the originator.

6

Tag Data Link Functions

6-1 Introduction to Tag Data Links

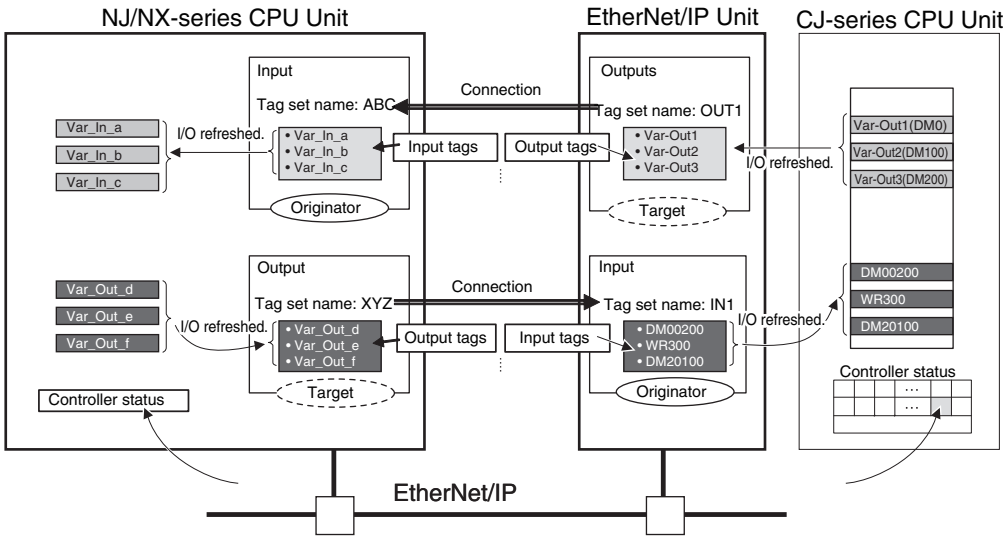
6-1-1 Tag Data Links

Tag data links enable cyclic tag data exchanges on an EtherNet/IP network between Controllers or between Controllers and other devices. Variables are assigned to tags. (You can also assign I/O memory addresses to tags.)
The settings for tag data links are made with the Network Configurator. Refer to *6-2 Setting Tag Data Links* on page 6-19 for information on how to make the settings.

Additional Information

You can also use the Sysmac Studio to set the tag data links.
Refer to *A-2 Use the Sysmac Studio to Set the Tag Data Links (EtherNet/IP Connections)* on page A-4 for details on setting the tag data links on the Sysmac Studio.

With a tag data link, one node requests the connection of a communications line to exchange data with another node.
The node that requests the connection is called the originator, and the node that receives the request is called the target.



For communications between Controllers, the connection information is set in the built-in EtherNet/IP port of the Controller that receives (consumes) the data (i.e., the originator).

Additional Information

For communications between a Controller and an I/O device, the connection information is set in the built-in EtherNet/IP port that is the originator. If an I/O device is used, the Network Configurator must have an EDS file installed that includes connection information for the I/O device.
Refer to *A-3 EDS File Management* on page A-41 for the installation procedure.

The output words and input words for each node for which data is exchanged must be set in the connection information. These words are called an output tag set and an input tag set, respectively. Each tag set must contain at least one tag.

The size of data for data exchange is the total size of tags included in the tag set. The size of the output tag set and the size of the input tag set must match.

Precautions for Correct Use

- Select the **Use** Option for the CIP message server of the built-in EtherNet/IP port. If the **Do not use** Option for the CIP message server is selected, tag data links cannot be performed. For the details on the settings, refer to *CIP Message Server* on page 4-19.
- If the **Use** Option is selected for Packet Filter of the built-in EtherNet/IP port, make sure to permit packets to be used for tag data links. If they are not permitted, tag data links are not possible. For the details on the settings, refer to *Packet Filter* on page 4-7.

6-1-2 Data Link Data Areas

Tags

A tag is a unit that is used to exchange data with tag data links.
Data is exchanged between the local network variables and remote network variables specified in the tags or between specified I/O memory areas.

Precautions for Correct Use

To maintain concurrency in the values of network variables that are assigned to tags, you must set refreshing tasks.
Refer to *6-1-7 Concurrency of Tag Data Link Data* on page 6-12 for details.

Tag Sets

When a data link connection is established, one or more tags (up to eight tags including Controller status) are configured as a collective set of tags for the connection. This is called a tag set. Each tag set represents a unit of data for one tag data link connection.
Tag data links are therefore created through a connection between one tag set and another tag set. A tag set name must be set for each tag set.

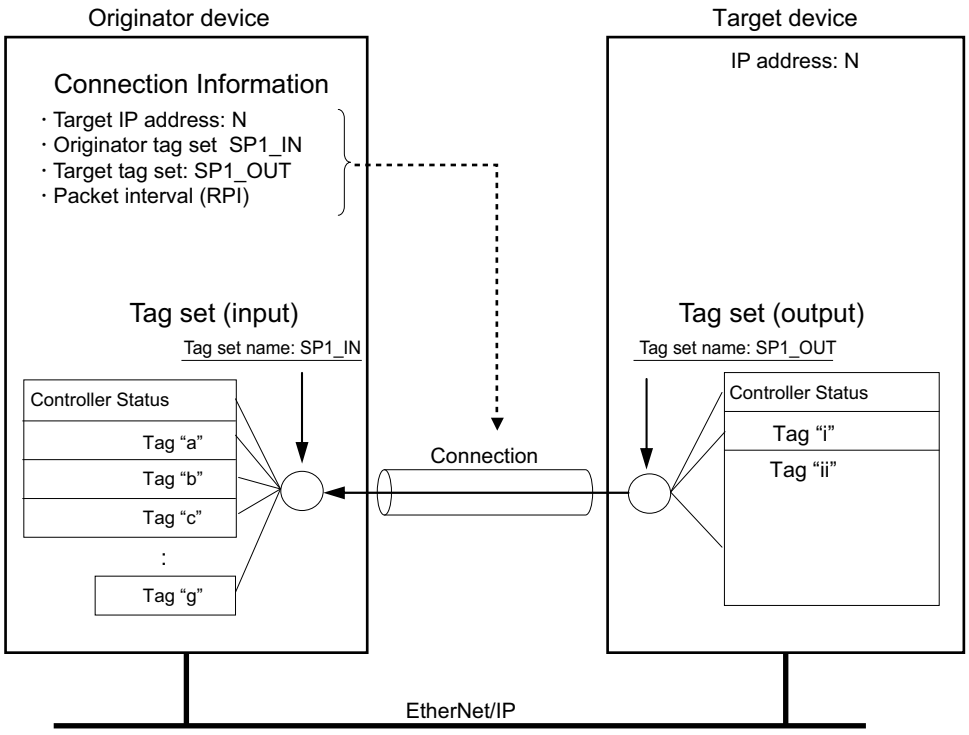
Note A connection is used to exchange data as a unit within which data concurrency is maintained.
Thus, data concurrency is maintained for all the data exchanged for one or more tags in one tag set.

Precautions for Correct Use

Data with tags is exchanged in the order that the tags are registered in the tag set. Register the tags in the same order of the input and output tag sets.

● Example

In the following example, input tags "a" to "g" at the originator are a tag set named *SP1_IN* and output tags "I" and "ii" are a tag set named *SP1_OUT*. A connection is set between these two tag sets.

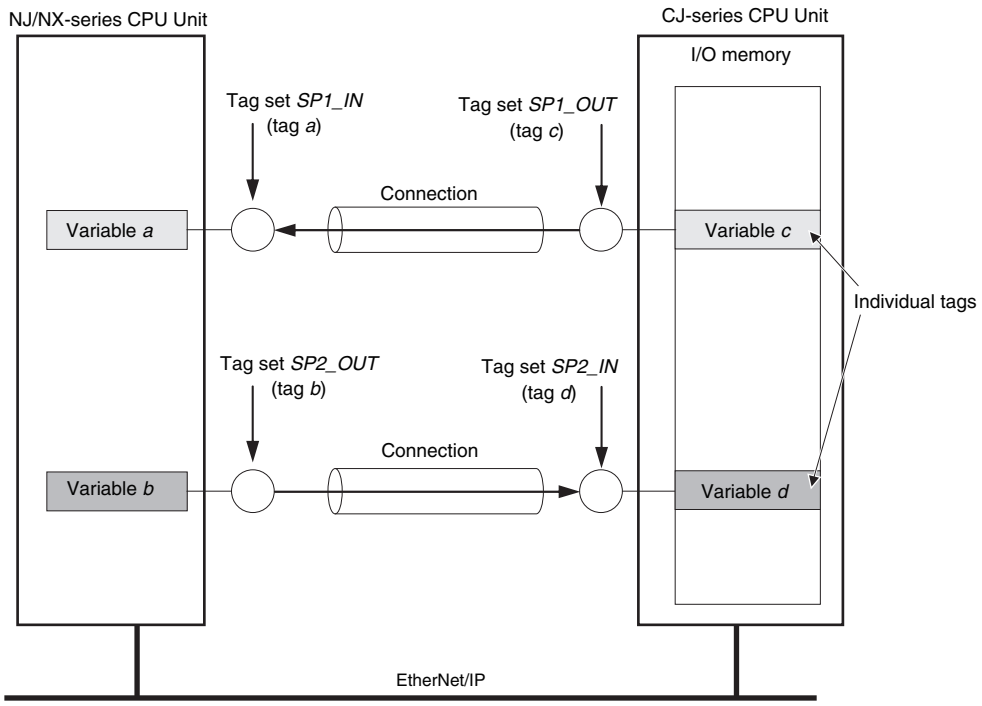


There are input (consume) and output (produce) tag sets. Each tag set can contain either input tags or output tags. The same input tag cannot be included in more than one input tag set.

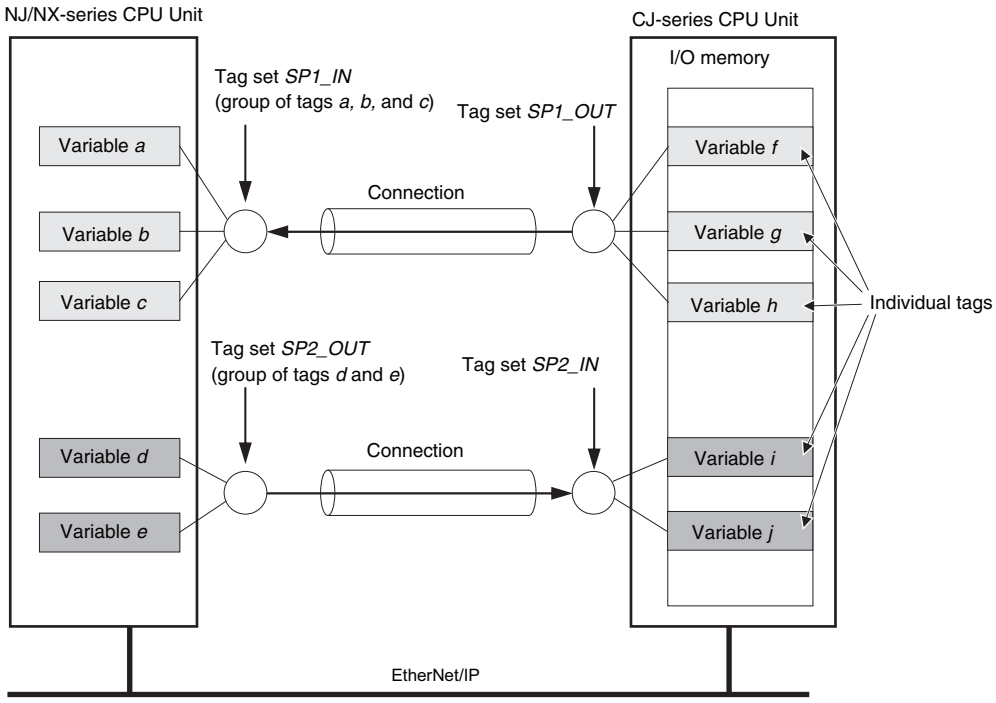
● Number of Tags in Tag Sets

You can set one or more tags for each of the input and output tag sets for one connection.
For example, you can set the input tag set with one tag, and the output tag set with more than one tag.

- Tag Set with Only One Tag Each
  With basic Network Configurator procedures, each tag set contains only one tag.



- Tag Sets Each with Multiple Tags
  As shown below, multiple tags can be grouped. You can assign up to eight tags (up to 722 words in total for an NX701 CPU Unit, and up to 300 words in total for an NX102, NX1P2, and NJ-series CPU Unit) in one tag set.



Note To enable a connection, each tag set must include only one of ether input tags or output tags. (Both input and output tags cannot be included in one tag set.)

6-1-3 Tag Data Link Functions and Specifications

The tag data link and performance specifications of the NJ/NX-series CPU Unit are given below.

<table>
<tr><th colspan="2" rowspan="3">Item</th><th colspan="6">Specification</th></tr>
<tr><th rowspan="2">NX701-□□□□</th><th rowspan="2">NX102-□□□□</th><th rowspan="2">NX1P2-□□□□□□□□</th><th colspan="2">NJ501-□□□□/NJ301-□□□□/NJ101-□□□□</th></tr>
<tr><th>Unit version 1.00 to 1.02</th><th>Unit version 1.03 or later</th></tr>
<tr><td colspan="2">Communications type</td><td colspan="6">Standard EtherNet/IP implicit communications (connection-type cyclic communications)</td></tr>
<tr><td colspan="2">Setting method</td><td colspan="6">After you have set tags, tag sets, and connections with the Network Configurator, you must download tag data link parameters to all devices on the EtherNet/IP network. After the parameters are downloaded, the EtherNet/IP Units are restarted to start the tag data links.
You can export network variables that you created on the Sysmac Studio to a CSV file.
You can then import the file to the Network Configurator and assign the network variables to the tags.</td></tr>
<tr><td rowspan="3">Tags*1</td><td>Supported variable types</td><td colspan="6">You can specify the following network variables as tags. *2, *3
• Global variables</td></tr>
<tr><td>Maximum number of words per tag</td><td>722 words (1,444 bytes)</td><td colspan="5">300 words (600 bytes)</td></tr>
<tr><td>Maximum number of tags</td><td colspan="3">256
(total of 512 with two ports)</td><td colspan="3">256*4</td></tr>
<tr><td rowspan="3">Tag sets</td><td>Maximum number of tags per tag set</td><td colspan="6">8 (7 when Controller status is included)</td></tr>
<tr><td>Maximum number of words per tag set</td><td>722 words (1,444 bytes)</td><td colspan="5">300 words (600 bytes)</td></tr>
<tr><td>Maximum number of tag sets</td><td>256 (total of 512 with two ports)</td><td>32 (total of 40 with two ports)*5</td><td colspan="4">32</td></tr>
<tr><td colspan="2">Connections</td><td>Maximum number of connections per Unit: 512 (256 per port)</td><td>Maximum number of connections per Unit: 64 (32 per port)</td><td colspan="4">Maximum number of connections per Unit: 32</td></tr>
<tr><td colspan="2">Connection type</td><td colspan="6">Each connection can be set for 1-to-1 (unicast) or 1-to-N (multicast) communications.</td></tr>
<tr><td rowspan="2">Packet interval (RPI)</td><td>0.5 to 10,000 ms in 0.5-ms increments</td><td>1 to 10,000 ms in 1-ms increments</td><td>2 to 10,000 ms in 1-ms increments</td><td>10 to 10,000 ms in 1-ms increments</td><td colspan="2">1 to 10,000 ms in 1-ms increments</td></tr>
<tr><td colspan="6">The packet interval can be set separately for each connection.</td></tr>
<tr><td rowspan="2">Allowed communications bandwidth per Unit (pps)</td><td>40,000 pps*6</td><td>12,000 pps *6</td><td>3,000 pps</td><td>1,000 pps</td><td colspan="2">3,000 pps</td></tr>
<tr><td>Note: The heartbeat is included.</td><td>Note: The heartbeat and the CIP Safety routing are included. *7</td><td colspan="4">Note: The heartbeat is included.</td></tr>
</table>

*1. When you specify a specific I/O memory address for a tag for an NX102, NX1P2 or NJ-series CPU Unit, create a variable with an AT specification for the I/O memory address on the Sysmac Studio, and then

specify the variable with the AT specification for the tag. For NX102 and NX1P2 CPU Units, you need to set memory used for CJ-series Unit to use the I/O memory address.
For details on memory settings used for CJ-series Unit, refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)*.

*2. You can import network variables created in the Sysmac Studio to the Network Configurator as tags.
If variables for which Network publish attribute is set on the Sysmac Studio contain I/O memory addresses, such as "0000" and "H0000", they are not exported to CSV files.

*3. The following table lists variables that you can specify as tags.

Data types		Example	Specification
Variables with basic data types		aaa	Supported
Enumerated variables		bbb	Supported
Array variables	**Arrays**	ccc	Supported
	Elements	ccc[2]	Supported
Structure variables	**Structures**	ddd	Supported
	Members	ddd.xxx	Supported
Union variables	**Unions**	eee	Supported
	Members	eee.yyy	Supported

*4. The maximum number of tags is given for the following conditions.
- All tag sets contain eight tags.
- The maximum number of tag sets (32) is registered.

*5. If more than 40 tag sets are set in total, a Number of Tag Sets for Tag Data Links Exceeded (840E0000 hex) event occurs.

*6. If the two built-in EtherNet/IP ports are used simultaneously, the maximum communications data size means the maximum data size of the total of the two ports.

*7. An NX102 CPU Unit with unit version 1.31 or later is required to use the CIP Safety routing.

6-1-4 Overview of Operation

In this manual, the connection information that is set is called tag data link parameters.
This section describes how to set tag data links with the Sysmac Studio and the Network Configurator.

Setting Network Variables (Sysmac Studio)

First, create any variables that you want to use for tag data links as network variables on the Sysmac Studio.

1 Set the Network Publish attribute to **Input** or **Output** in the Global Variable Table for variables you want to use for tag data links (i.e., as tags).

2 To maintain concurrency in tag data within a tag set, set all tags (i.e., variables with a Network Publish attribute) within the same tag set as follows:
Set a refreshing task for variables with a Network Publish attribute to maintain concurrency for tag data link data as described below.
- Maintain concurrency in the tag data in a tag set.
- The timing of updating network variables that are assigned to tags is synchronized with the execution period of a program that accesses the network variables.

Refer to *6-1-7 Concurrency of Tag Data Link Data* on page 6-12 for details on the concurrency of tag data link data.

Precautions for Correct Use

- If a variable with an AT specification is used as a tag, you do not need to set a refreshing task.
  It is refreshed in the primary periodic task.
- You cannot use the following notation, which specifies an I/O memory address, in the variable name of any variable used in a tag data link.
  a) Variable names that contain only single-byte numerals (Example: 001)
  b) Variable names with the following single-byte letters (uppercase or lowercase) followed by single-byte numerals
     1) H (Example: H30)
     2) W (Example: w30)
     3) D (Example: D100)
     4) E0_ to E18_
- When the server function of CIP message communications is disabled, the tag data links cannot be used. Enable the server function of CIP message communications. Refer to *CIP Message Server* on page 4-19 for details on setting CIP message server.

Setting and Downloading Tag Data Link Parameters (Network Configurator or Sysmac Studio)

The following tag data link parameters (e.g., connection information) are created with the Network Configurator or the Sysmac Studio, and then the parameters are downloaded to all originator devices on the EtherNet/IP network.
When the tag data links are used on built-in EtherNet/IP ports, use the Network Configurator to make the following settings.

Additional Information

In the settings of the following tag data link parameters, the specifications of the settable numbers and the ranges differ depending on the CPU Unit or the version of the CPU Unit. For details, refer to *1-3-1 Specifications* on page 1-9.

1 Creating the Configuration Information
Register EtherNet/IP ports and EtherNet/IP Units to create connections that define the tag data links. For details, refer to *6-2-3 Registering Devices* on page 6-21.

2 Setting Tags
Create CPU Unit variables for input (consume) tags and output (produce) tags.
You can import and export network variables that are created on the Sysmac Studio to CSV files. This allows you to register them as tags on the Network Configurator.
Output (produce) tags can be defined to clear output data to 0 or to hold the output data from before the error when a fatal error occurs in the CPU Unit.

3 Setting Tag Sets
Create output tag sets and input tag sets and assign tags to them. (You can create up to eight I/O tag sets.) You can specify the Controller status that indicates the CPU Unit's operating status (operating information and error information) in a tag set.

4 Setting connections

Link the output tag sets for the target device and the input tag sets for the originator device as connections.

● Connection Setting Parameters

The connection settings in step 4 above have the following setting parameters.

- Setting the Requested Packet Interval (RPI)
  The RPI (Requested Packet Interval) is the I/O data refresh cycle on the Ethernet line when tag data links are established. With EtherNet/IP, data is exchanged on the communications line at the RPI that is set for each connection, regardless of the number of nodes.
  With the built-in EtherNet/IP port, you can set RPI for each connection.

- Setting Multi-cast or Unicast Communications
  You can select a multicast connection or unicast (point-to-point) connection as the connection type in the tag data link connection settings.
  With a multicast connection, you can send an output tag set in one packet to multiple nodes and make allocations to the input tag sets.
  A unicast connection separately sends one output tag set to each node, and so it sends the same number of packets as the number of input tag sets.
  Therefore, multicast connections can decrease the communications load if one output tag set is sent to multiple nodes.
  To use a multicast connection and send an output tag set in one packet to multiple nodes, the following settings for the receiving node must be the same as the settings of the sending node: the connection type (multicast), the connection I/O type, packet internal (RPI), and timeout value.

Precautions for Correct Use

- The performance of communications devices is limited to some extent by the limitations of each product's specifications. Consequently, there are limits to the packet interval (RPI) settings.
  Refer to *14-2 Adjusting the Communications Load* on page 14-7*Checking the Device Bandwidth Usage* on page A-24 and set an appropriate packet interval (RPI).
- If multicast connections are used, however, use an Ethernet switch that has multicast filtering, unless packets are received by all nodes in the network.
  If an Ethernet switch without multicast filtering is used, multicast packets are broadcast to the entire network, and so the packets are sent to nodes that do not require them, which will cause the communications load on those nodes to increase.
- If you use data tag links with multicast traffic at a baud rate over 100 Mbps, use an Ethernet switch that supports a baud rate of 1000 Mbps.
  If there is an Ethernet device on the same network that communicates at 100 Mbps or less, the device may affect tag data link communications and cause tag data links to be broken, even if the device is not related to tag data link communications.

Additional Information

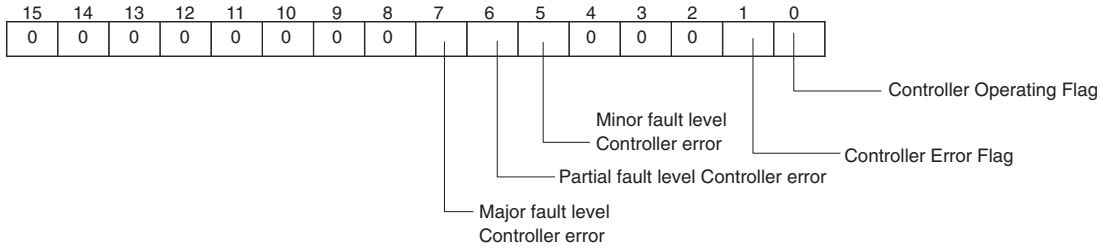
- To calculate the number of connections of each connection type, refer to *14-1-2 Calculating the Number of Connections* on page 14-4.
- If the maximum number of connections is exceeded, you must review the number of connections for the built-in EtherNet/IP port, or the number of nodes. When you use an NJ-series CPU Unit, you can also consider adding EtherNet/IP Units.

6-1-5 Starting and Stopping Tag Data Links

Tag data links are automatically started when the data link parameters are downloaded from the Network Configurator and the power supply to the NJ/NX-series Controller is turned ON.
Thereafter, you can start and stop tag data links for the entire network or individual devices from the Network Configurator. Starting and stopping tag data links for individual devices must be performed for the originator.
Furthermore, you can use system-defined variables to start and stop the entire network. Refer to *6-2-12 Starting and Stopping Tag Data Links* on page 6-69 for details.

6-1-6 Controller Status

You can include the Controller status as a member of a tag set in the data sent and received.
The Controller status is a set of flags that indicate the operating status of the CPU Unit (operating information, error information, Controller error level).
If the Controller status is specified as an output (send) tag, the Controller status is added to the start of the tag set in the following format.
(Select the **Include** Option for **Controller Status** in the upper right of the **Edit Tag Set** Dialog Box.)



Note Of the flags in bits 5 to 7 that indicate the current error level, only the flag for the highest error level changes to TRUE.
For example, if a minor fault level Controller error and a major fault level Controller error occur at the same time, only the flag for the major fault level Controller error (bit 7) will change to TRUE and the flag for the minor fault level Controller error (bit 5) will remain as FALSE.

To receive the Controller status, specify the Controller status for the In - Consume Tab Page in the dialog box used to edit the receive tag set.
(Select the **Include** Option for **Controller Status** in the upper right of the **Edit Tag Set** Dialog Box.)
When a tag data link is started, the contents of the Controller status is stored in the system variables that are given below.

- Target PLC Operating Mode

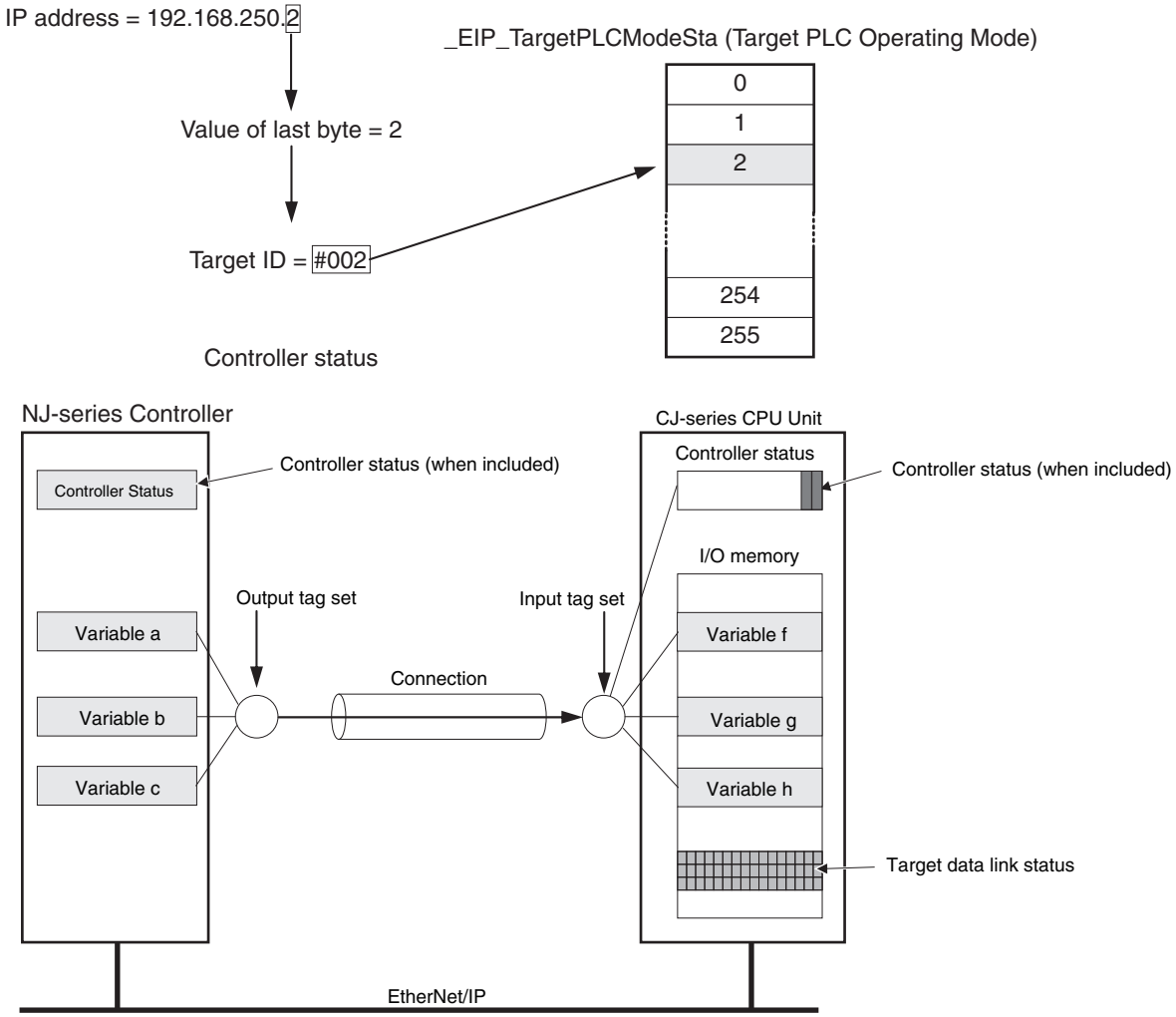
  NX701 CPU Unit: *_EIP1_TargetPLCModeSta* (for the built-in EtherNet/IP port 1)
  _EIP2_TargetPLCModeSta (for the built-in EtherNet/IP port 2)
  NX102 CPU Unit: *_EIP1_TargetPLCModeSta* (for the built-in EtherNet/IP port 1)
  _EIP2_TargetPLCModeSta (for the built-in EtherNet/IP port 2)
  NX1P2 CPU Unit: *_EIP1_TargetPLCModeSta* (for the built-in EtherNet/IP port 1)
  NJ-series CPU Unit *_EIP_TargetPLCModeSta*

- Target PLC Error Information

  NX701 CPU Unit: *_EIP1_TargetPLCErr* (for the built-in EtherNet/IP port 1)
  _EIP2_TargetPLCErr (for the built-in EtherNet/IP port 2)
  NX102 CPU Unit: *_EIP1_TargetPLCErr* (for the built-in EtherNet/IP port 1)
  _EIP2_TargetPLCErr (for the built-in EtherNet/IP port 2)

NX1P2 CPU Unit: *_EIP1_TargetPLCErr* (for the built-in EtherNet/IP port 1)
NJ-series CPU Unit: *_EIP_TargetPLCErr*

Example: Using an NJ-series CPU Unit to send the Target PLC Operating Mode of the Target Node with an IP Address of 192.168.250.2



Additional Information

The target node ID may be duplicated depending on the IP address of the target node. In this case, it is necessary to change the target node ID on the Network Configurator so that the same address could not be used by more than one node.
For information on how to change the target node ID, refer to Step 4 under *Registering Devices in the Register Device List* in *Connection Settings* in *6-2-5 Connection Settings* on page 6-36.

When you use multiple connections to communicate with one specific node, the information of the Controller status is stored in the following variables if the Controller status is specified in the input tags and the output tags for all the connections.

Controller status	Variable name	Description of operation
Controller Operating Flag	Target PLC Operating Mode • NX701 CPU Unit *_EIP1_TargetPLCModeSta* (for the built-in EtherNet/IP port 1), or *_EIP2_TargetPLCModeSta* (for the built-in EtherNet/IP port 2) • NX102 CPU Unit *_EIP1_TargetPLCModeSta* (for the built-in EtherNet/IP port 1), or *_EIP2_TargetPLCModeSta* (for the built-in EtherNet/IP port 2) • NX1P2 CPU Unit *_EIP1_TargetPLCModeSta* (for the built-in EtherNet/IP port 1) • NJ-series CPU Unit *_EIP_TargetPLCModeSta*	This flag shows the operation information of the Controller at the target node. (When the Built-in EtherNet/IP Port Is the Originator of the Connection) The array element that corresponds to the target node ID at the target is set to TRUE when all information for all the connections to the relevant target node shows operating status. You can change the target node ID for the IP address from the Network Configurator. This status information is enabled when the Controller status is included in the communications data for both the originator and the target node. This variable is updated when necessary.
Controller Error Flag	Target PLC Error Information • NX701 CPU Unit *_EIP1_TargetPLCErr* (for the built-in EtherNet/IP port 1), or *_EIP2_TargetPLCErr* (for the built-in EtherNet/IP port 2) • NX102 CPU Unit *_EIP1_TargetPLCErr* (for the built-in EtherNet/IP port 1), or *_EIP2_TargetPLCErr* (for the built-in EtherNet/IP port 2) • NX1P2 CPU Unit *_EIP1_TargetPLCErr* (for the built-in EtherNet/IP port 1) • NJ-series CPU Unit *_EIP_TargetPLCErr*	This variable shows the error status (logical OR of fatal and non-fatal errors) of the target node Controllers. (When the Built-in EtherNet/IP Port Is the Originator of the Connection) You can change the target node ID for the IP address from the Network Configurator. The Controller Error Flags are enabled when the Controller status is included in the communications data for both the originator and target node. This variable is updated when necessary.

Additional Information

Even if you specify including the Controller status in output (produce) tags, you do not necessarily need to include the Controller status in input (consume) tags.
If you do not include the Controller status in an input (consume) tag, the contents of the Controller status are not updated in the Target PLC Operating Mode and Target PLC Error Information variables, but they are sent in the input (consume) tag.
Therefore, you can use the Controller status data that was received in the input (consume) tag as receive data.

6-1-7 Concurrency of Tag Data Link Data

To maintain the concurrency of data in a tag data link, you must set a refreshing task for each network variable that is assigned to a tag.

- Maintain concurrency in tag data in a tag set.
- The timing of updating network variables that are assigned to tags is synchronized with the execution period of the program that accesses the network variables

Additional Information

A refreshing task maintains concurrency of the value of a global variable from all tasks that access that global variable. This is achieved by specifying a single task that can write to that global variable and not allowing any other task to write to that global variable.
For details on refreshing tasks, refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)*.

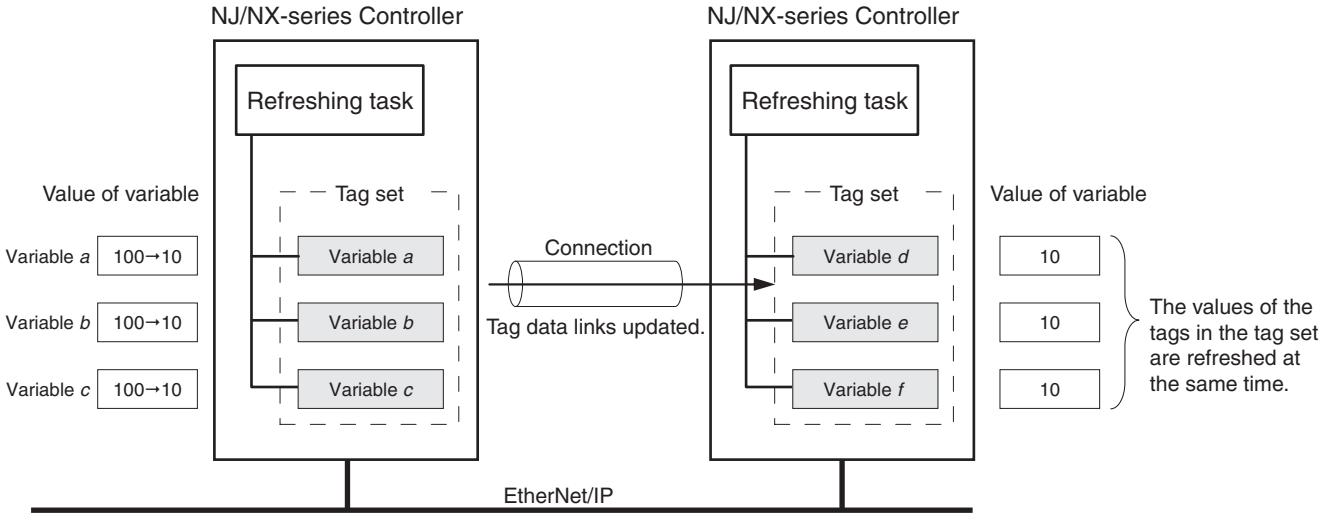
Maintaining Concurrency in the Tag Data in a Tag Set

To maintain concurrency in the values of multiple tags in a tag set, the tags (variables with a Network Publish attribute) must satisfy the following four conditions.

a. The tags must be assigned to the same tag set (connection).
b. A refreshing task must be set for network variables assigned to the tags, and the refreshing task must be the same for all the tags in the tag set. *1
c. For NX102, NX1P2, and NJ-series CPU Units, a tag with an AT specification must not be included in the tag set.
d. The variable access time set for each task must be set to a higher value than is required to transfer the tag data.
   Refer to *14-3-3 Relationship between Task Periods and Packet Intervals (RPIs)* on page 14-26 for details on the variable access time and data transfer.

*1. If you set a refreshing task for network variables, you must set a variable access time to allocate enough time to access the network variables from outside of the Controller.

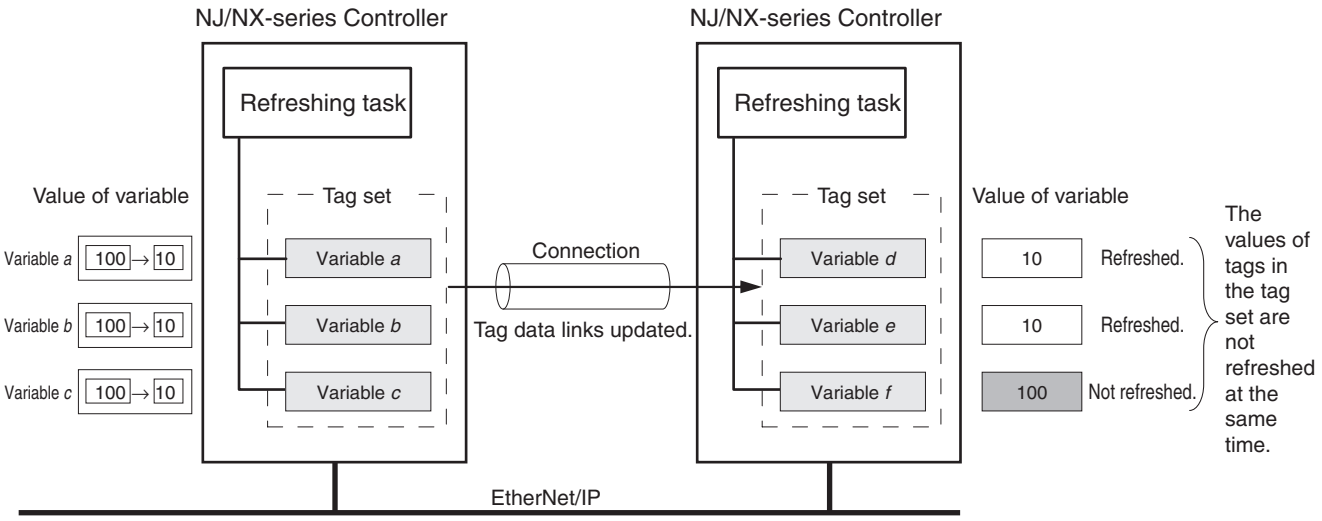
- Setting Refreshing Tasks for Tags (Network Variables)
  Concurrency of the tags in the tag set is maintained.



Additional Information

For NX102, NX1P2, and NJ-series CPU Units, you do not need to set a refreshing task for variables (tags) with AT specifications since they are updated in the primary periodic task.

- Not Setting Refreshing Tasks for Tags (Network Variables)
  Concurrency of the tags in the tag set is not maintained.



Synchronizing the Update Timing of Network Variables (Tags) with the User Program Execution Period

To have the values of network variables (tags) updated to the latest tag data values each time the user program that accesses those network variables is executed, set the refreshing task for the network variables (tags) to the same type of the task as for the user program that accesses the network variables (tags).

The difference between the operation of tags with a refreshing task that is the same as the user program and tags without a refreshing task is described below.

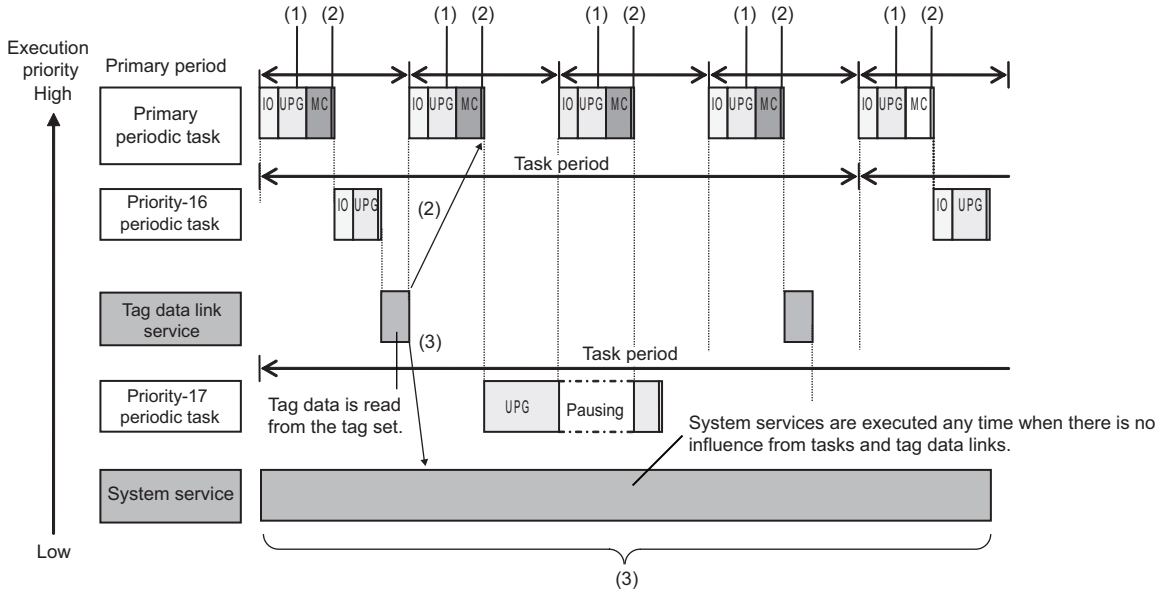
- Tag (network variable) with a refreshing task
  The tag is refreshed each time the program with the task that is set as the refreshing task is executed.
- Tag (network variable) without a refreshing task
  The tag is refreshed by the system service. Refreshing is not synchronized to the execution timing of the program.

The following figures show the refreshing timing of network variables for the respective CPU Units.

NX701 CPU Unit

- The tag data link service is executed without being affected by the tasks and system services.
- The system services are executed at the required time without being affected by the tasks and tag data link service.

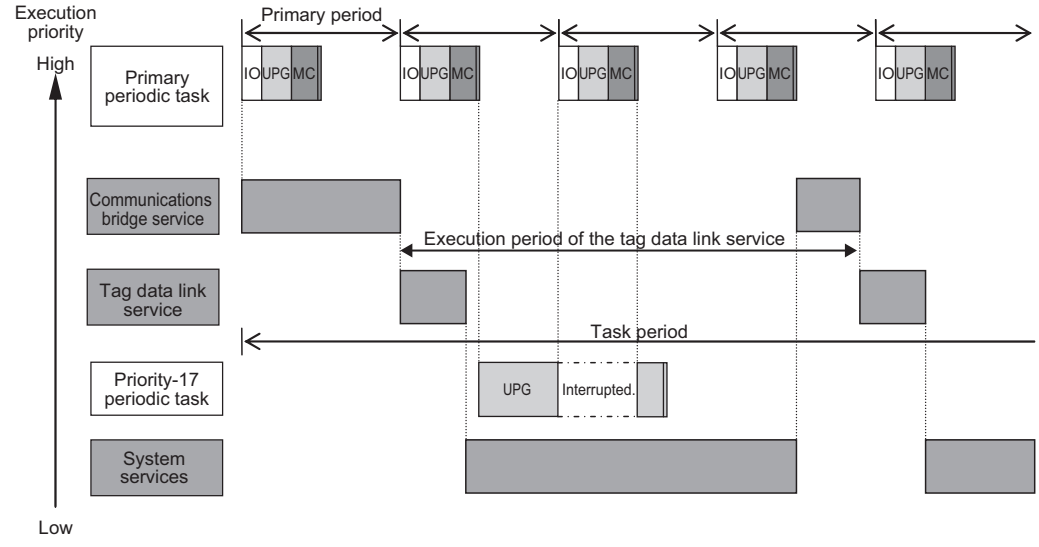
(1) Execution timing of the program
(2) Refresh timing of network variables (tags) with the primary periodic task set as the refreshing task*
(3) Refresh timing of network variables (tags) that do not have the primary periodic task set as the refreshing task



*: Refreshed during system common processing 2 in the task processing.

NX102 CPU Units

- The communications bridge service, tag data link service and system service can be executed in parallel with the tasks.
- The execution priority is higher in the order of communications bridge service, tag data link service and then system service.

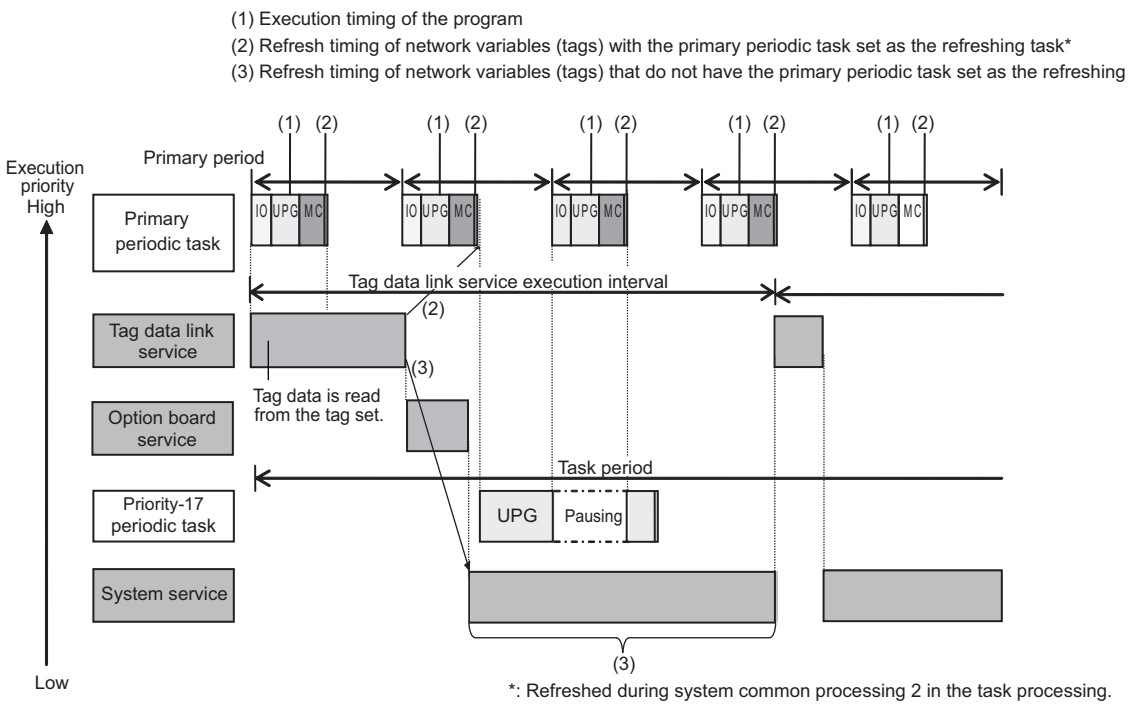


Version Information

The communications bridge service is provided for NX102 CPU Units with unit version 1.31 or later.

NX1P2 CPU Unit

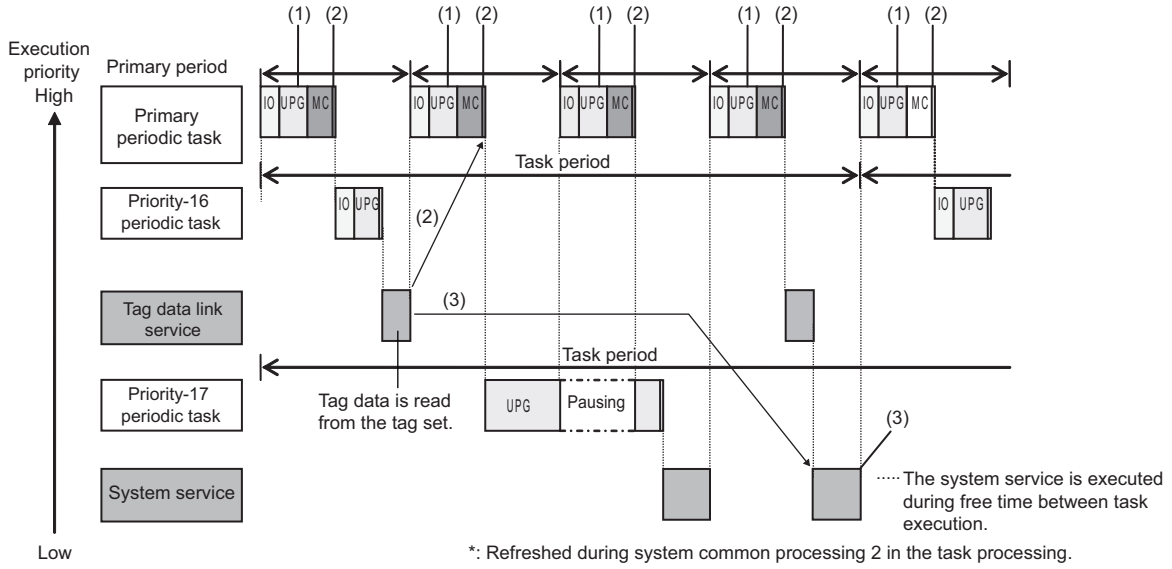
- You can execute the tag data link service, option board service or system services in parallel with the execution of tasks.
- The order of execution priority is tag data link service, option board service and then system services.



NJ-series CPU Units

- Execution of the tag data link service is given priority over execution of the priority-17 periodic task. However, execution of the primary periodic task and priority-16 periodic task is given even higher priority.
- System services are executed in unused time between execution of all of the tasks and tag data link service.

(1) Execution timing of the program
(2) Refresh timing of network variables (tags) with the primary periodic task set as the refreshing task*
(3) Refresh timing of network variables (tags) that do not have the primary periodic task set as the refreshing task



*: Refreshed during system common processing 2 in the task processing.

Additional Information

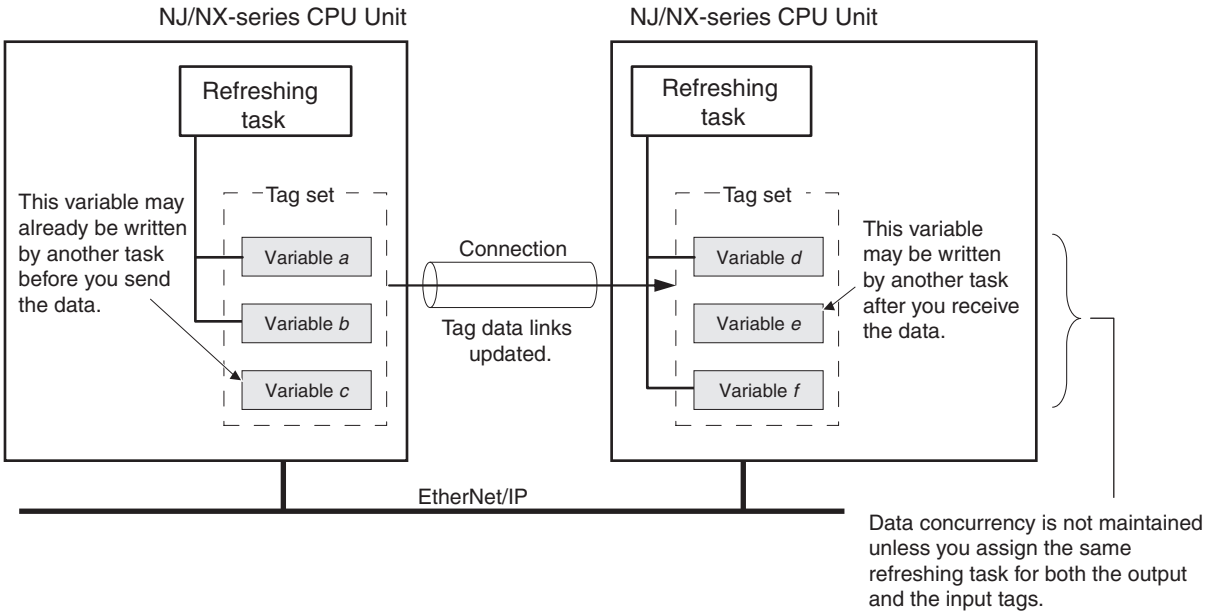
If a program needs to access a network variable with an AT specification, set the program in the primary periodic task so that it matches the refresh timing of the network variable with the AT specification. (This applies to NX102, NX1P2 and NJ-series CPU Units.)

Additional Information

Relationship between Refreshing Tasks and Data Concurrency in Tag Data Links

If you do not specify a refreshing task for global variables in tag data links, the following may occur.

1. When data is sent for the output tag set, another task may have already written different values before that data is sent, depending on the timing of the task.
2. When data is received by an input tag set, another task may write different values after that data is received, depending on the timing of the task.
   Therefore, to maintain concurrency of data in tag data links, you must specify the same refreshing task on both the output CPU Unit and the input CPU Unit.



Required Processing Time to Maintain Concurrency

When you set a refreshing task for tags (network variables) to maintain the concurrency of data link data, the processing time required for that specified task increases. Due to this increase in task processing time, tag data link data may not be refreshed at the packet interval (RPI) period set for each connection.

Therefore, you need to adjust the packet interval (RPI) settings to match the period of the task specified as the refreshing task.

Refer to *14-3-3 Relationship between Task Periods and Packet Intervals (RPIs)* on page 14-26 for details.

Task Setup Procedure

1. Set the global variables for which to specify a refreshing task, and set the refreshing tasks and accessing tasks in the **Settings for Exclusive Control of Variables in Tasks** in the **Task Setup** Tab Page on the Sysmac Studio.
2. Set the variable access time for each refreshing task.

For details, refer to *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)*.

6-2 Setting Tag Data Links

Additional Information

You can also use the Sysmac Studio to set the tag data links.
Refer to *A-2 Use the Sysmac Studio to Set the Tag Data Links (EtherNet/IP Connections)* on page A-4 for details on setting the tag data links on the Sysmac Studio.

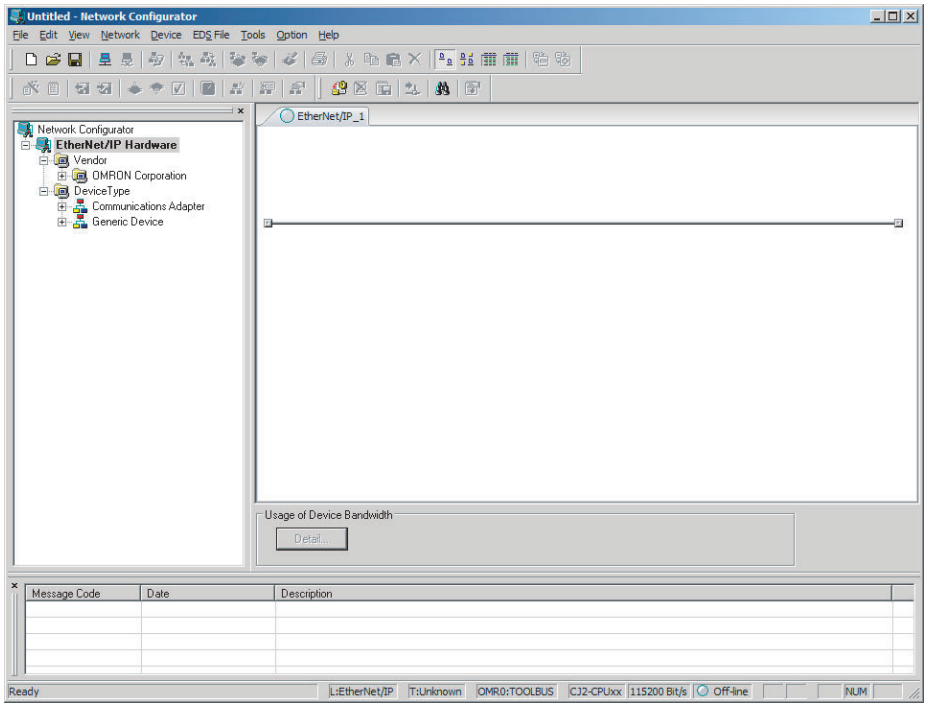
6-2-1 Starting the Network Configurator

Procedure

Tag data links are set from the Network Configurator. Use the following procedure to start the Network Configurator.

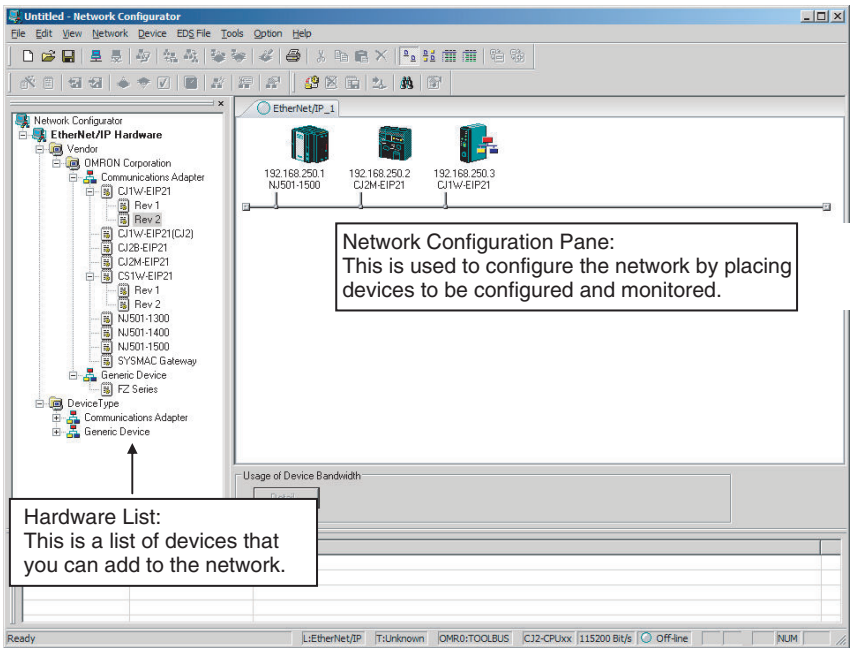
● Using the Windows Start Menu

To start the Network Configurator, select **OMRON** – **Sysmac Studio** – **Network Configurator for EtherNetIP** – **Network Configurator**.
When the Network Configurator starts, the following window is displayed.

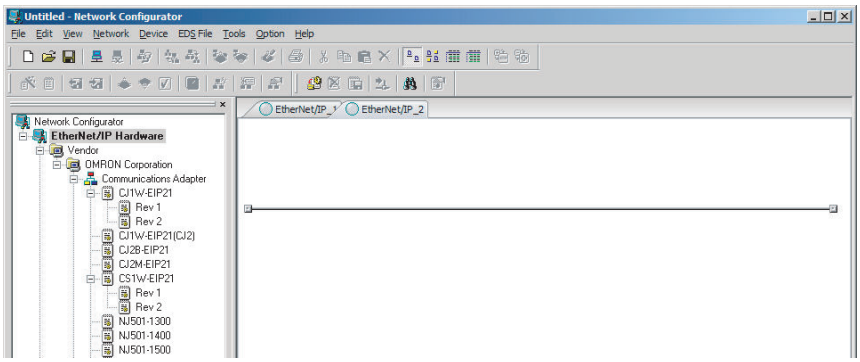


Main Window

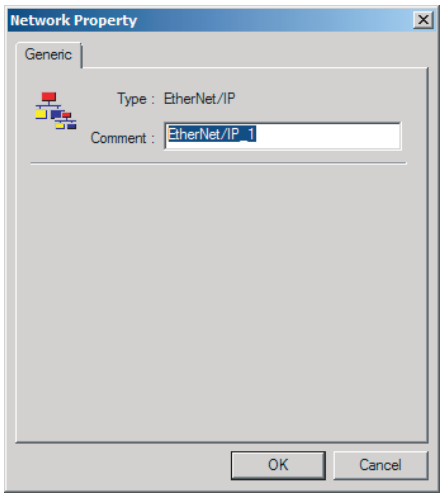
The Main Window consists of a Hardware List and a Network Configuration Pane, as shown in the following diagram.



To manage two or more networks, you can select **Network** – **Add**. You can add a new Network Configuration Pane.



To change the network name displayed in the Network Tab Page, select **Network** – **Property**. You can change the network name as set in the Comment Field of the Network Property Dialog Box.



6-2-2 Tag Data Link Setting Procedure

This section describes the procedure to set tag data links (i.e., connection information).
For data links between Controllers, the connection information is set only in the originator, i.e., the node that receives data.

1 **Create the network configuration.**

1. Register all the built-in EtherNet/IP ports for which to create connections, in the EtherNet/IP Network Configuration Pane. (Refer to *6-2-3 Registering Devices* on page 6-21)
   Note If a system has already been installed, connect online to the EtherNet/IP network and upload the network configuration. (Refer to *6-2-10 Uploading Tag Data Link Parameters* on page 6-62)

↓

2 **Create the tag and tag set connections.**

1. Create tags and tag sets for all the registered devices (built-in EtherNet/IP ports). (Refer to *6-2-4 Creating Tags and Tag Sets* on page 6-23)
2. Create a connection for the originator device (i.e., the registered device that receives data as input data). (Refer to *6-2-5 Connection Settings* on page 6-36)

↓

3 **Download the tag data link parameters. (Refer to *6-2-9 Downloading Tag Data Link Parameters* on page 6-59)**

↓

4 **Make sure that the tag data links are operating normally, by using the indicators for the built-in EtherNet/IP port (refer to *NJ/NX-series Troubleshooting Manual (Cat. No. W503)*) and the device monitor function of the Network Configurator (refer to *15-2 Checking Status with the Network Configurator* on page 15-3).**

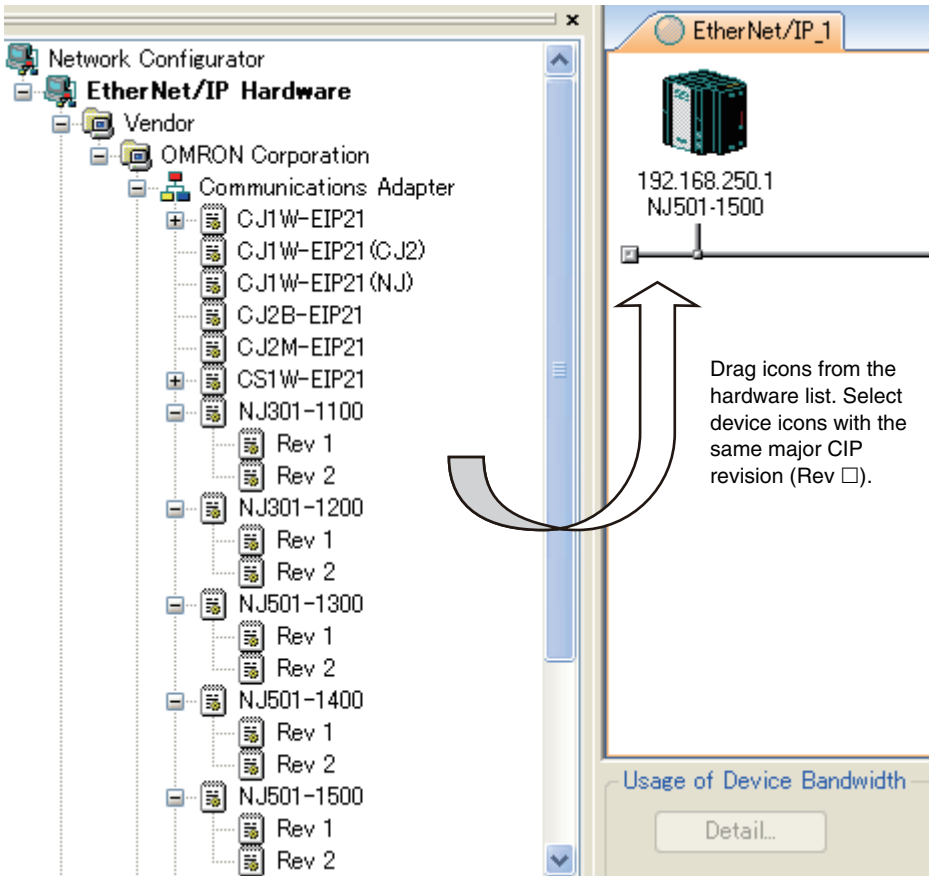
↓

5 **Make sure that the output tag data is reflected in the input tags by checking the Watch Tab Page on the Sysmac Studio.**
Refer to the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for the procedure.

6-2-3 Registering Devices

Register all of the devices required in the equipment (such as EtherNet/IP Units performing tag data links) in the network configuration.

1 Register the devices that will participate in the tag data links by dragging the devices from the Hardware List and dropping them in the Network Configuration Pane on the right. (To drag and drop an icon, click and hold the left mouse button over the icon, move the icon to the destination, and release the mouse button.)
You can also select a device in the Hardware List and press the Enter Key to register it. The icon of the device is displayed in the Network Configuration Pane, as shown in the following picture.



The device names and major CIP revisions (Rev ☐) are displayed in the hardware list.
For the NJ/NX-series CPU Units, device names and major CIP revisions are as shown in the following table.

Device name in Hardware List	Unit version	CIP revisions	
		Major revision	Revision name in Hardware List
NX701	Unit version 1.10 or later	2	None
NX102-☐☐☐☐	Unit version 1.30 or later	2	None
NX1P2	Unit version 1.13 or later	2	None
NJ501-☐☐☐☐	Unit version 1.00 to 1.02	1	Rev1
	Unit version 1.03 or later	2	Rev2
NJ301-☐☐☐☐	Unit version 1.01 or 1.02	1	Rev1
	Unit version 1.03 or later	2	Rev2
NJ101	Unit version 1.10 or later	2	None

Precautions for Correct Use

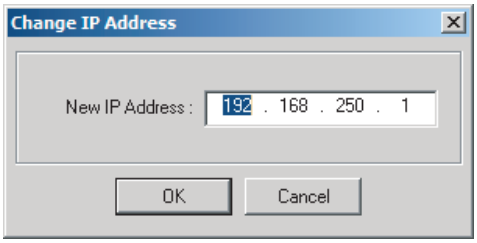
Make sure that you select the devices with the same device names and the same major CIP revisions as the devices you use in the actual operation. The following will occur if any device name or CIP revision is incorrect when you attempt to download tag data link parameters on the Network Configurator.

- If a device name is incorrect, an error message will be displayed saying "**Specified device can not be accessed, or wrong device type**", and the download will fail.
- If a revision is incorrect, a message will be displayed saying "**Wrong unit revision**", and the download will fail.

Similarly, the above will occur when performing upload or comparison of the tag data link parameters.

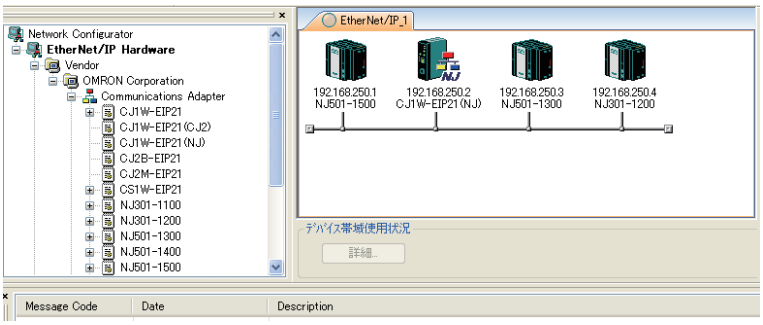
In any of the above cases, refer to *6-2-17 Changing Devices* on page 6-78 and change the device.

2 Right-click the registered device's icon to display the pop-up menu, and select **Change IP Address**.



3 Set the IP address to match the node address (IP address) actually used in the device, and click the **OK** Button.

4 Repeat steps 1 to 3, and register all devices to which tag data links are made.



6-2-4 Creating Tags and Tag Sets

You must create tag sets and member tags that are required to create connections for a registered EtherNet/IP port and EtherNet/IP Unit. You can set the network variables used in control programs for tags.

This section first describes the basic procedure to create tags and tag sets, as described in (1) below. Then it explains how to import variables with a Network Publish attribute from the Sysmac Studio to the Network Configurator, as described in (2) below.

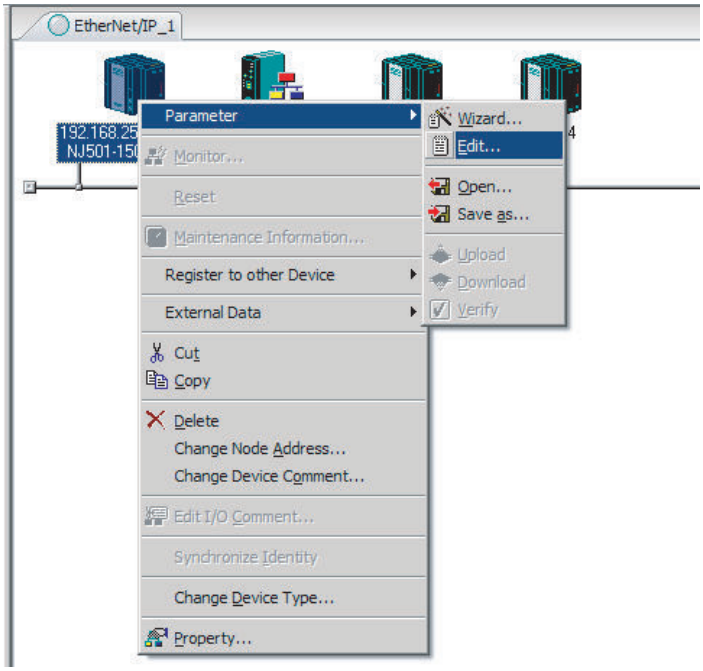
1. Creating Tags and Tag Sets with the Network Configurator's Device Parameter Editing Function

2. Importing Variables with a Network Publish Attribute Created in the Sysmac Studio to the Network Configurator

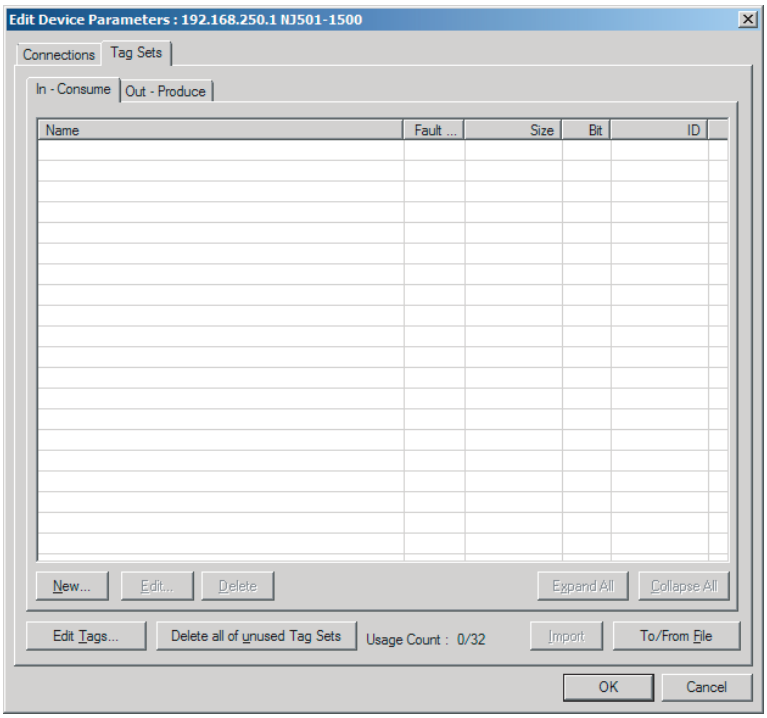
(1) Creating Tags and Tag Sets with the Network Configurator's Device Parameter Editing Function

● Creating a Tag Set

1 Double-click the icon of the device for which to create a tag set to display the **Edit Device Parameters** Dialog Box. Or, right-click the icon to display the pop-up menu, and select **Parameter** – **Edit**.



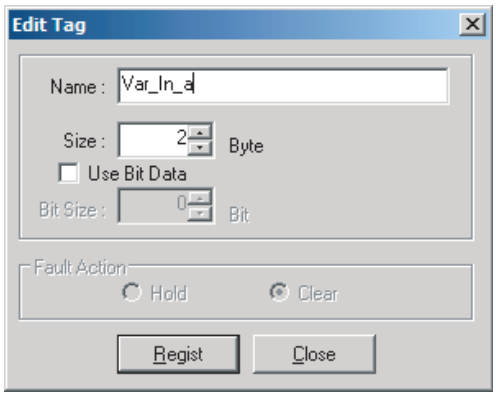
2 Click the **Tag Sets** Tab at the top of the **Edit Device Parameters** Dialog Box.
There are two kinds of tag sets: input (consume) and output (produce).



● Creating and Adding Tags

1 Click the **Edit Tags** Button.
The **Edit Tags** Dialog Box is displayed. Register input (consume) tags and output (produce) tags separately.

2 Click the **In - Consume** Tab, and then click the **New** Button.
The **Edit Tag** Dialog Box is displayed.



3 Enter the variable name directly into the **Name** Box. (Example: Var_In_a)

Additional Information

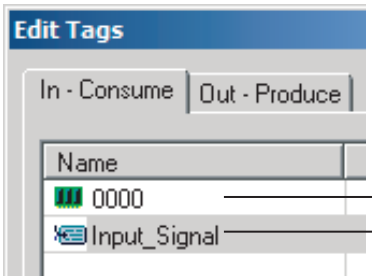
- You can use the following characters in tag names.
  0 to 9, A to Z, a to z, single-byte kana, _ (underbar), and multi-byte characters (e.g., Japanese)
- You cannot use the following characters in tag names.
  ! “ # $ & ‘ ( ) * + , - . / : ; < = > ? @ [ ] ^ ‘ % spaces or text strings that start with numerals (0 to 9)
- The maximum length of a tag name is 255 bytes.
- Specify array variables, structure variables, and union variables, if any, as shown below.
  - Specifying array elements
    Example: array [2][3] (or array [2,3]) and array [2][3][4] (or array [2,3,4])
  - Specifying structure members
    Example: Struct.member (Separate the member name with a period.)
  - Specifying union members
    Example: Union.member (Separate the member name with a period.)

Precautions for Correct Use

NX102 CPU Unit, NX1P2 CPU Unit, and NJ-series CPU Unit

- To specify an I/O memory address for a tag, create a variable with an AT specification of the I/O memory address on the Sysmac Studio, and then specify the variable with the AT specification for the tag.
  For NX102 and NX1P2 CPU Units, you need to set CJ memory to use the I/O memory address. For details on CJ memory setting, refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)*.
- If you enter the following I/O memory addresses for tag names on the Network Configurator, the tags are directly assigned to the I/O memory addresses in the CPU Unit, and not to the variables. Always specify variable names for tags.
  a) Variable names that contain only single-byte numerals from 0000 to 6143
  b) Variable names with the following single-byte letters (uppercase or lowercase) followed by single-byte numerals
     - H (H000 to H511)
     - W (W000 to W511)
     - D (D00000 to D32767)
     - E0_ to E18_ (E0_00000 to E0_32767, to E18_00000 to E18_32767)

  You can check the memory address or variable to which a tag is assigned, with icons in the **Edit Tags** Dialog Box.



NX701 CPU Unit

- If you apply the notation that specifies the above I/O memory address for a tag name, a Tag Name Resolution Error occurs. A tag data link will not be started.

4 Input the size of the tag in bytes in the **Size** Field.
Specify the tag size to be the same as the data type size of the variable.
To use a BOOL variable, select the **Use Bit Data** Check Box, and enter *1* in the **Size** Field.

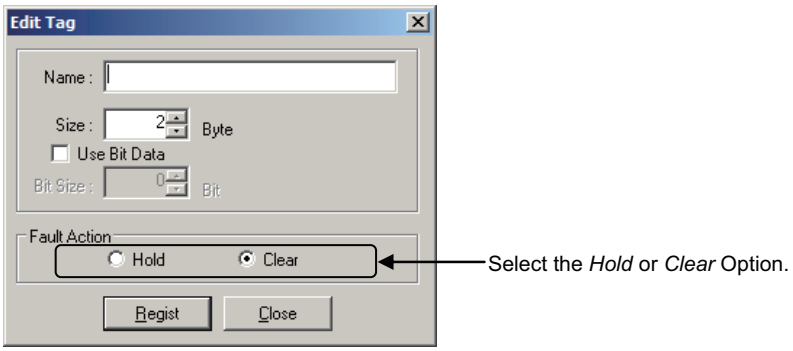
5 Click the **Regist** Button to register the tag.
If an I/O memory address is specified as the tag name, another **Edit Tag** Dialog Box will be displayed with the next address as the tag name so that you can register the next tag consecutively.
After you register the tags, click the **Close** Button.

6 Click the **Out - Produce** Tab, and then click the **New** Button.
The **Edit Tag** Dialog Box is displayed. Input output tags in the same way.
In case a major fault occurs in the CPU Unit, use the **Fault Action** setting of the output (produce) tag to select whether to clear output data or continue to send data.

The **Fault Action** setting is not required for input (consume) tag sets.

- Retain output after major fault: **Hold** (default)
  Output data maintains its previous status even after a major fault occurs.

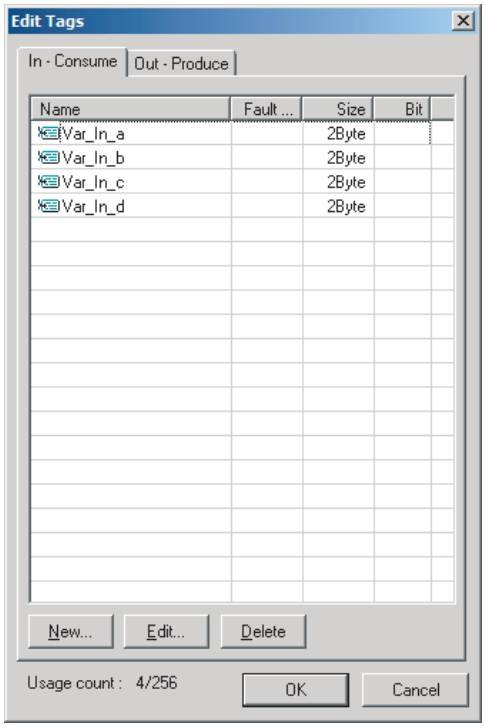
- Clear output at major fault: **Clear**
  Output data is cleared to 0 when a major fault occurs.



Precautions for Correct Use

Connections are cut off if any of the following errors occurs in the CPU Unit that is the originator while tag data links are active.
- Major fault level Controller error
- Partial fault level Controller error

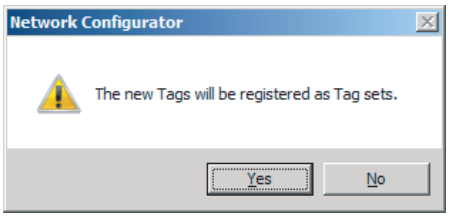
7 After you register all of the required tags, click the **OK** Button in the **Edit Tags** Dialog Box.



Precautions for Correct Use

Make the following settings to refresh all of the tag data in one tag set at the same time.
- Use the Sysmac Studio, in advance, to specify the same refreshing task for all of the variables that are assigned to tags in the tag set.
- Do not place tag variables that have AT specifications in I/O memory and tag variables that do not have AT specifications in the same tag set.

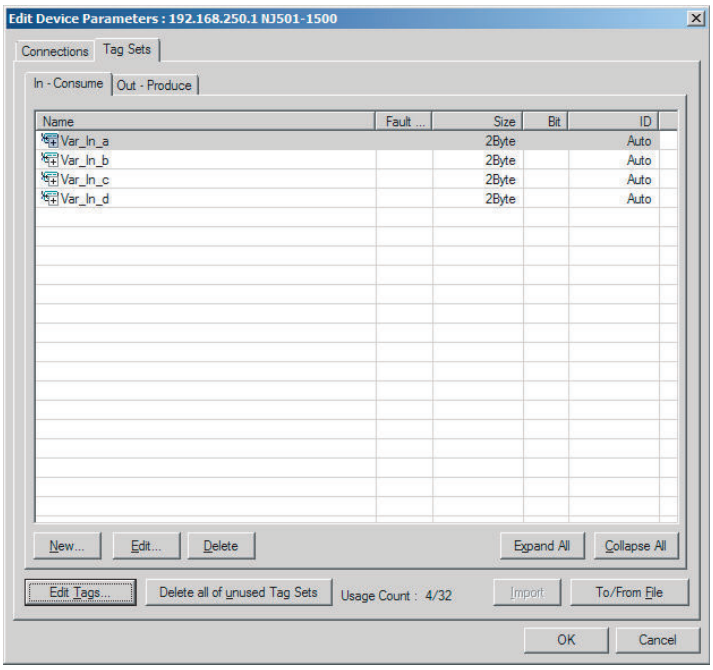
8 At this point, a confirmation dialog box is displayed to check whether the registered tag names are used as the tag set names. A tag set can contain up to eight tags, but tag sets are registered with one tag per tag set if the tag names are registered as tag set names. In this case, click the **Yes** Button.



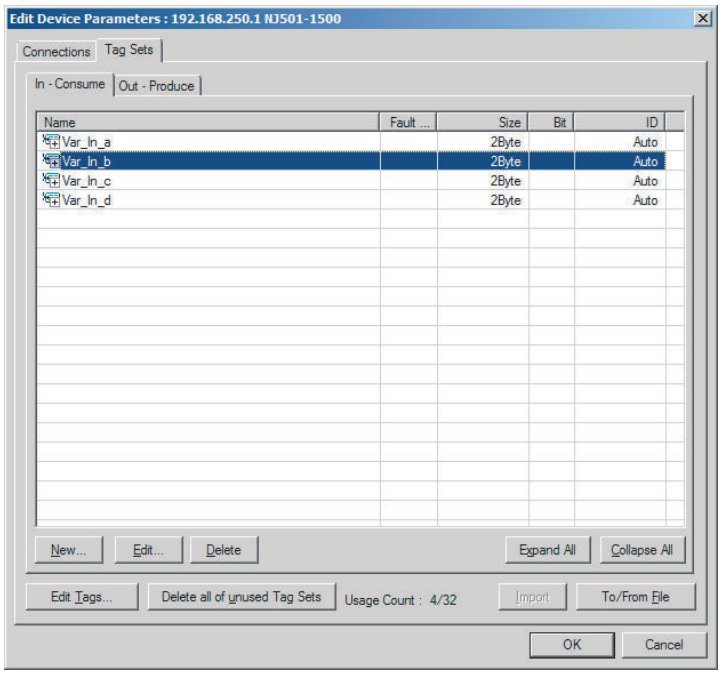
If the **No** Button is clicked, you can add more tags to the tag set. Refer to step 8 in Changing and Registering Tag Sets for details on how to register new tags first and add more tags to the tag set later.

● Changing and Registering Tag Sets

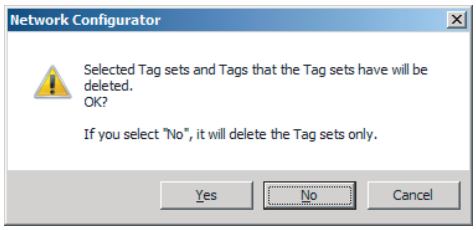
1 The following dialog box is displayed when the tags in the **Edit Tags** are registered directly as tag sets.



2 If an input tag is already registered in an input tag set, and you want to change its registration to a different input tag set, it is necessary to delete the tag from the tag set in which it was originally registered.
Open the **Edit Device Parameters** Dialog Box, select the tag set containing the tag that you want to delete on the **Tag Sets** Tab Page, and click the **Delete** Button. (If there are other tags registered in the tag set, it is possible to delete just one tag by selecting the tag that you want to delete in the **Edit Tag Set** Dialog Box and clicking the Button.)

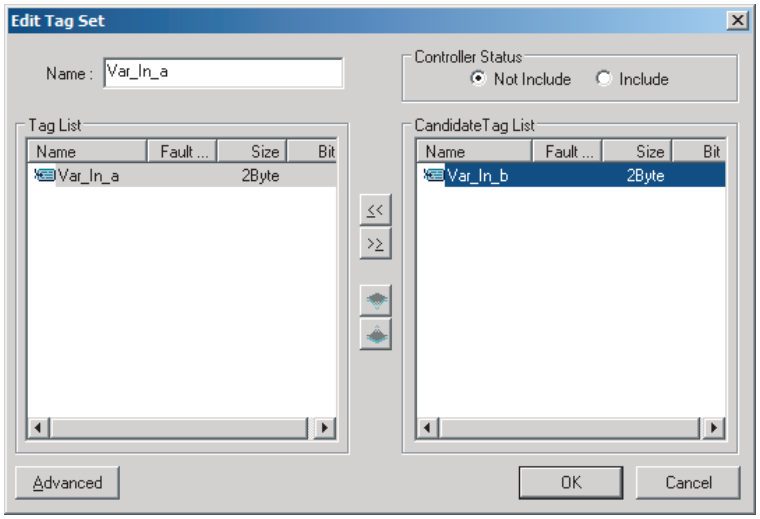


A confirmation message is displayed.



If the **No** Button is clicked, only the selected tag set is deleted. Click the **No** Button.

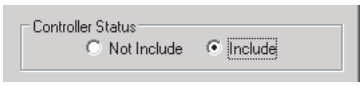
3 To edit a registered tag set and add tags, either double-click the tag set, or select the tag set and click the **Edit** Button.
The **Edit Tag Set** Dialog Box is displayed.



The **Tag List** on the left side of the dialog box shows tags that are already registered, and the **Candidate Tag List** on the right side of the dialog box shows the other tags that are not registered yet.

To add a tag, select it in the **Candidate Tag List** and click the Button.

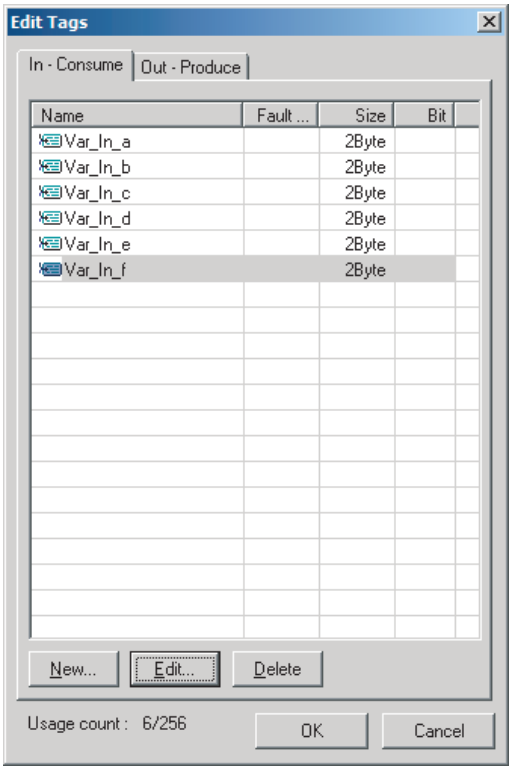
4 To include the Controller status in the tag set, select the **Include** Option for the **Controller Status** at the upper-right corner of the **Edit Tag Set** Dialog Box.



5 To confirm the change, click the **OK** Button in the **Edit Tag Set** Dialog Box.

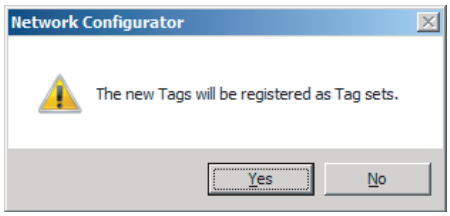
6 Click the **OK** Button in the **Edit Device Parameters** Dialog Box.

7 If you want to just add a new tag and register it in an existing tag set, first register the new tag by following steps 1 in Creating a Tag Set to 7 in Creating and Adding Tags. In this example, input tags, Var_In_e and Var_In_f, are newly added.

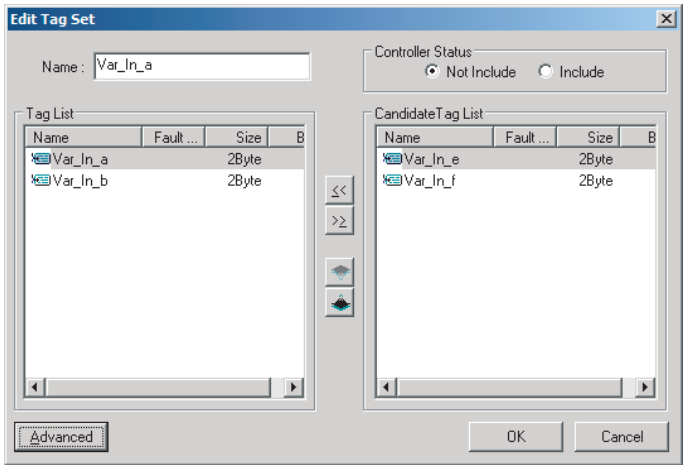


8 After you register the tags, click the **OK** Button in the **Edit Tags** Dialog Box.

9 At this point, a confirmation dialog box is displayed to check whether you want to use the registered tag names as tag set names. They are supposed to be added as tags in this case, so click the **No** Button. Then, the tags are registered just as tags but not as tag sets.

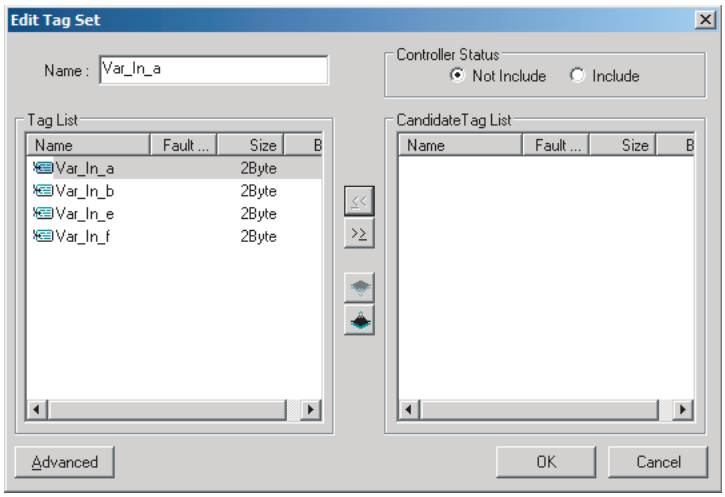


10 To register the newly added tags in an existing tag set, either double-click the desired tag set, or select the tag set and click the **Edit** Button.



The Tag List on the left side of the dialog box shows tags that are already registered in the tag set, and the Candidate Tag List on the right side of the dialog box shows the other tags that are not registered yet.

11 Select the tags that you want to add from the Candidate Tag List and click the  Button.



You can register up to eight tags in a tag set. (If you include the Controller status in the tag set, you can register up to only seven tags, and two bytes are added to the size.)
Tag data is sent and received in the order of tags displayed in the tag list. To change the order of tag data, select a tag and click the  or  Button.

12 To confirm the change, click the **OK** Button in the **Edit Tag Set** Dialog Box.

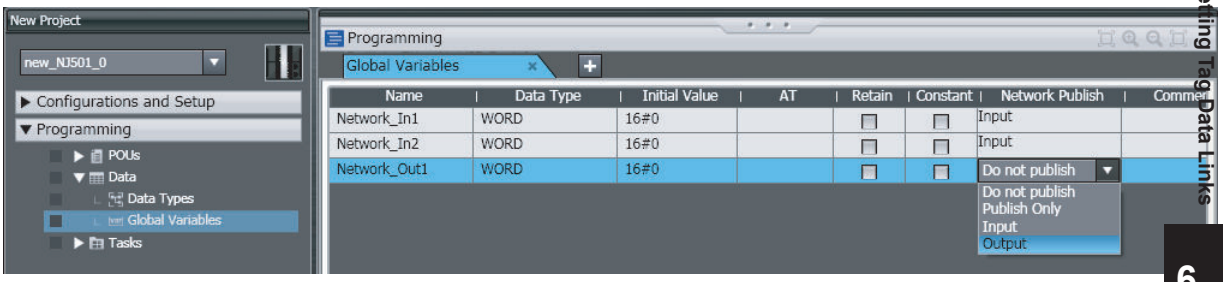
13 Click the **OK** Button in the **Edit Device Parameters** Dialog Box.

Importing Variables with a Network Publish Attribute Created in the Sysmac Studio to the Network Configurator

You can create network variables in the Sysmac Studio and import these variables to the Network Configurator to assign them to tags and tag sets. Use the following procedure.

● Exporting Global Variables on the Sysmac Studio

1 Create a global variable on the global variable table of the Sysmac Studio and select **Input** or **Output** for the Network Publish attribute of the variable.



2 Select **Export Global Variables** - **Network Configurator...** from the **Tools** Menu.
Any global variables with **Input** or **Output** set for the Network Publish attribute are imported from the csv file through the import procedure described below (Importing to the Network Configurator).

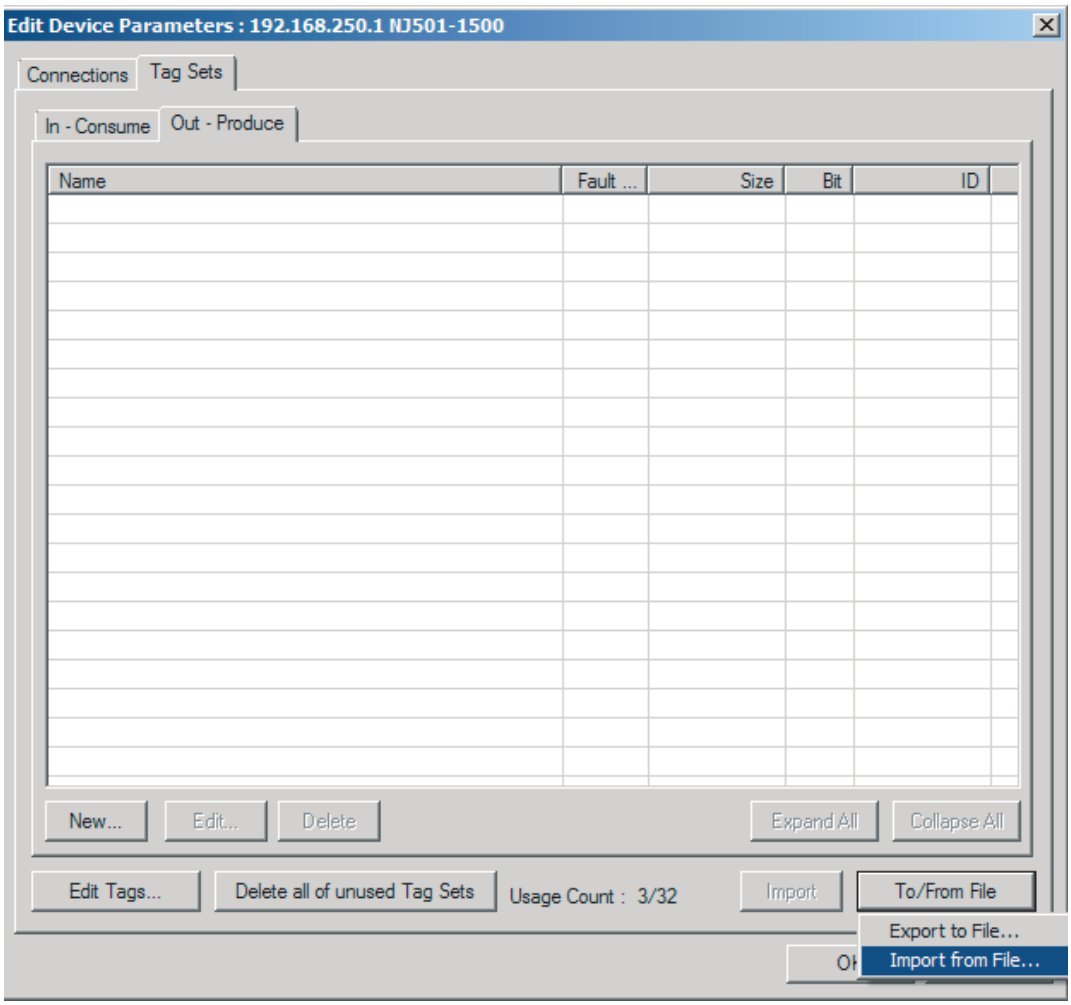
● Importing to the Network Configurator

Precautions for Correct Use

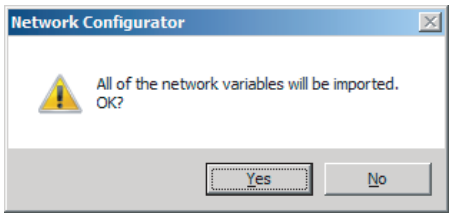
Variables with a Network Publish attribute that have variable names that are the same as the I/O memory address notation, such as, "0000" and "H0000" are not exported to CSV files.
- Variable names that contain only single-byte numerals (Example: 001)
- Variable names with the following single-byte letters (uppercase or lowercase) followed by single-byte numerals
  - H (Example: H30)
  - W (Example: w30)
  - D (Example: D100)
  - E0_ to E18_ (Example: EA_100)

1 From the devices registered on the Network Configurator, select and double-click the icon of the device for which you want to import the variable with a Network Publish attribute. Then, the **Edit Device Parameters** Dialog Box is displayed.
Or, right-click the icon to display the pop-up menu, and select **Device** - **Parameter** - **Edit**.

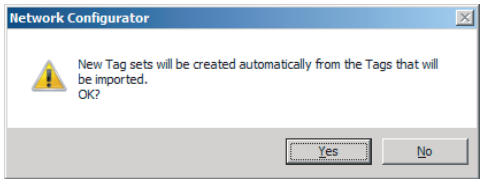
2 Click the **Tag Sets** Tab at the top of the **Edit Device Parameters** Dialog Box. Select **Import from File** from the **To/From File** Button.



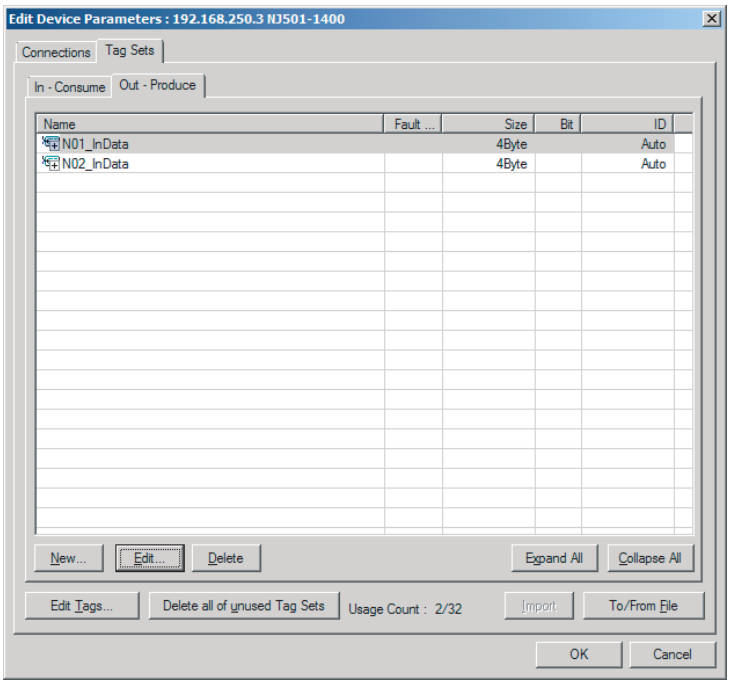
To import all variables with a Network Publish attribute, click the **Yes** Button. To import only some of these variables, click the **No** Button.



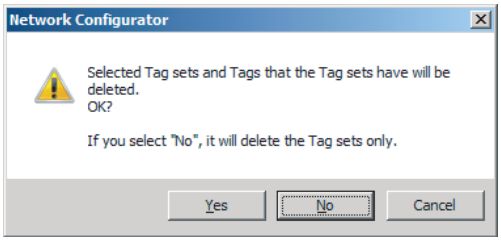
After you import the variables to the tags, click the **Yes** Button to automatically create tag sets, or click the **No** Button to set up tag sets manually.



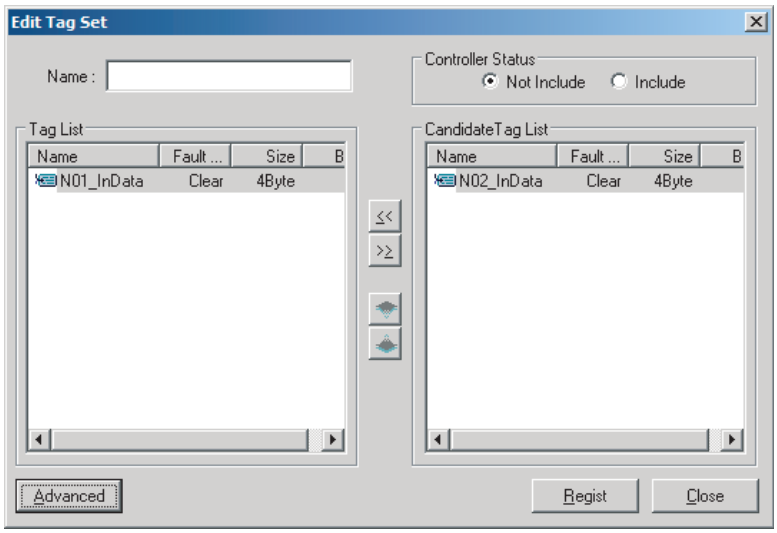
If you select the **Yes** Button in the previous step, the variables will be imported as shown below on the **Tag Sets** Tab Page. Each variable will be imported into a separate tag set and the device parameters will be automatically edited. (The variable name will be used for the tag set name.)



To place more than one input variable (input tag) imported from the Sysmac Studio into one tag set, you must delete the input tags that were registered.
Select the tag set containing the variables you want to put into a tag set, then click the **Delete** Button. A message box is displayed to confirm that you want to delete the selected tag set and the tags contained in that tag set. You only want to delete the tag set, so click the **No** Button.



Click the **New** Button to create a new tag set. To place more than one tag in an existing tag set, double-click the tag set, or select it and click the **Edit** Button.
The **Edit Tag Set** Dialog Box is displayed. Imported tags that are not registered in another tag set are displayed in the **Candidate Tag List** on the right side of the **Edit Tag Set** Dialog Box. Click the Button to add tags individually.



3 You can change tag set names in this dialog box. To confirm a change, click the **Regist** Button in the **Edit Tag Set** Dialog Box.

4 Perform steps 1 to 3 for all the devices to which tag data links are made to import variables and to create tag sets.

6-2-5 Connection Settings

After you create the tag sets, click the **Connections** Tab at the top of the **Edit Device Parameters** Dialog Box, and set the following connection information.

- The target devices and tag sets with which connections are opened
- The connection type (multicast or unicast)
- The length of the packet intervals (RPI)
- Connection name (optional)

Make the connections settings on the originator only. The connections settings are not necessary on the target device.

Precautions for Correct Use

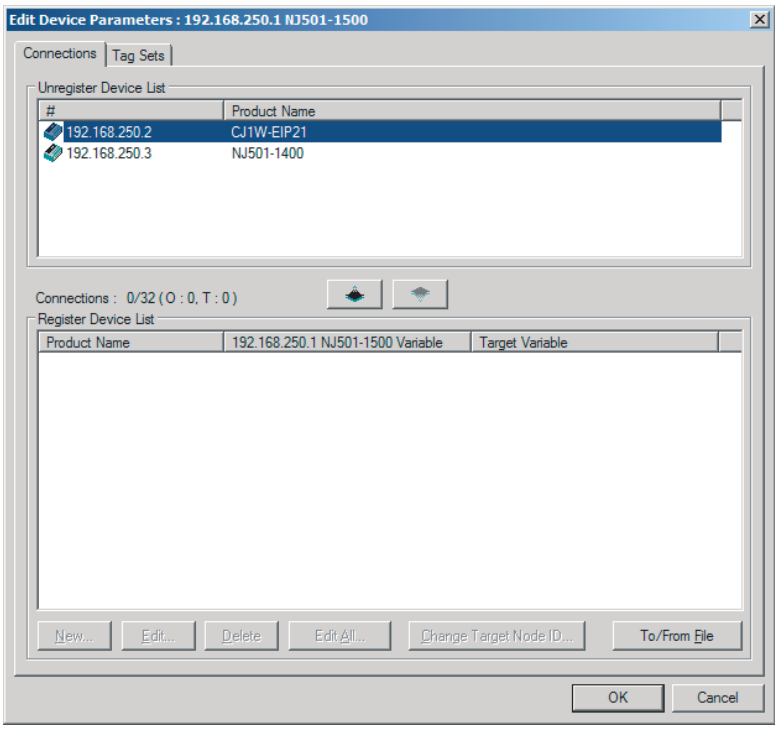
Make the connections settings after you create tag sets for all of the devices involved in tag data links.

Connection Settings (Connections Tab Page)

● Registering Devices in the Register Device List

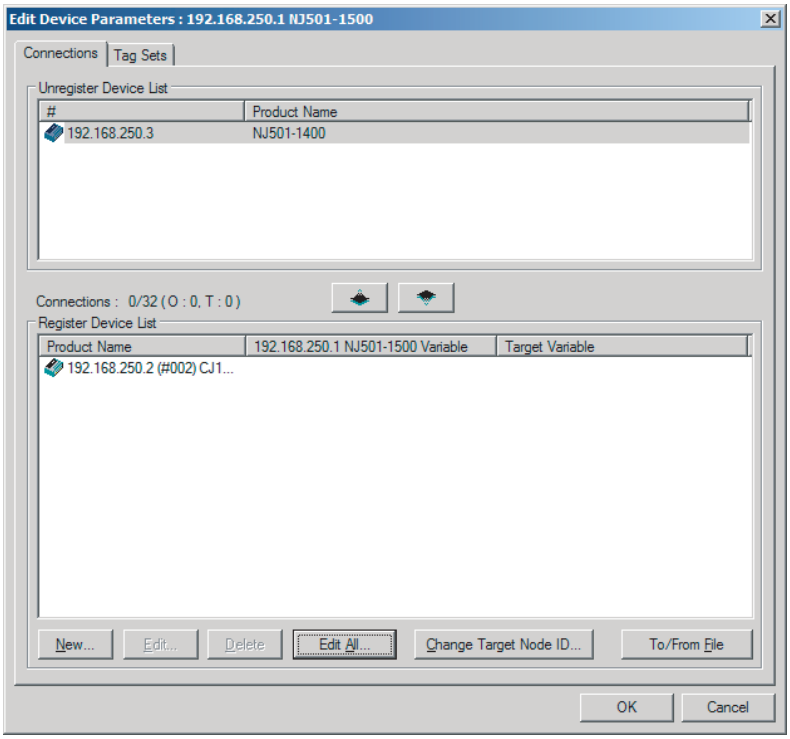
1 Double-click the icon of the device for which to make originator settings in the Network Configuration Pane of the Network Configurator. The **Edit Device Parameters** Dialog Box is displayed. Or, right-click the icon to display the pop-up menu, and select **Parameter** – **Edit**.

2 Click the **Connections** Tab in the **Edit Device Parameters** Dialog Box.
All of the devices registered in the network (except the local node) are displayed.



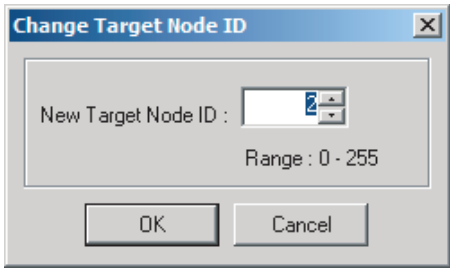
3 In the **Unregister Device List**, click the target device that requires connection settings so its color changes to gray, and click the  Button.

The selected target device is displayed in the **Register Device List**, as shown below.



4 Target node IDs are assigned to the devices that are registered in the **Register Device List**. The target node ID serves as the bit array position for the following variables in the originator Controller: Target Node Controller Mode, Target Node Controller Error Information, Target

Node Error Information, Registered Target Node Information, and Normal Target Node Information. By default, the target ID is automatically set to the rightmost 8 bits of the IP address. In the example above, the target device's IP address is 192.168.250.2, so the target node ID is #002. If a target node ID is duplicated and you want to change the target node ID, click the **Change Target Node ID** Button and change the target ID.



● Editing Settings for Individual Connections

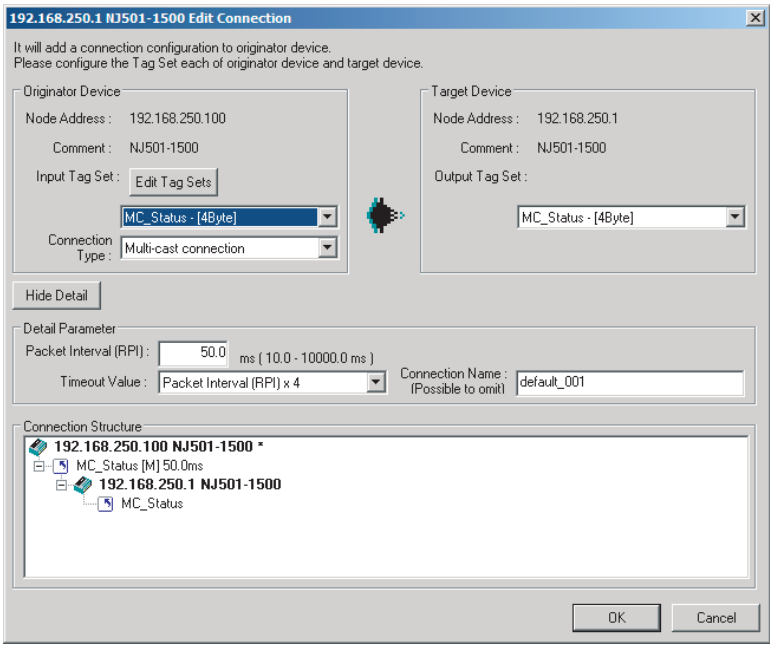
You can edit each connection separately.

Refer to *Editing Settings for All Connections* on page 6-40 for information on how to edit all the connections in a table format.

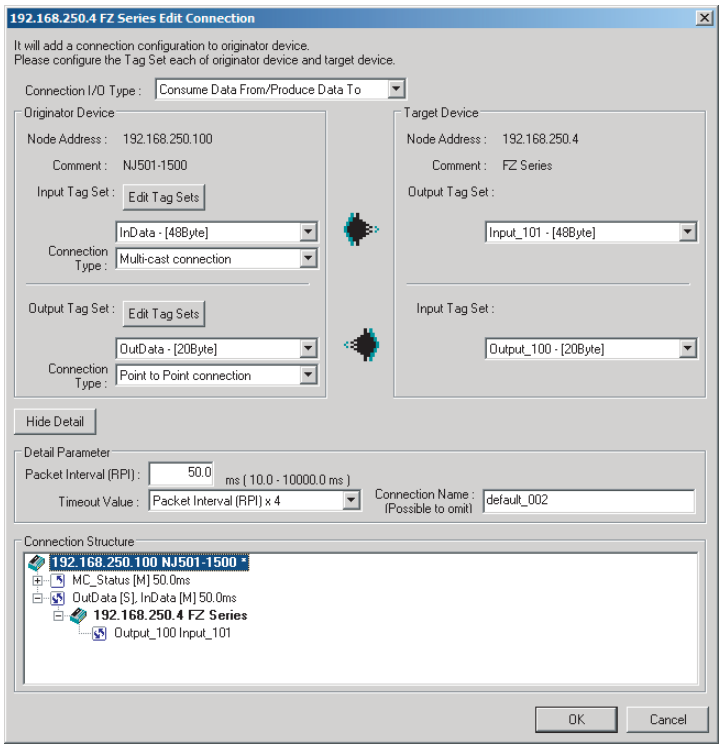
1 Click the **Connections** Tab and then the click the **New** Button.

The following **Edit Connection** Dialog Box is displayed according to the type of device that is selected.

- (A) Using Built-in EtherNet/IP Ports as Targets (for Input Only)



- (B) Using Other EtherNet/IP Devices as Targets (for Settings Other Than Input Only)



The settings are as follows:

Setting	Description
Connection I/O Type	Select **Input Only (Tag type)** to use tag data links with a CS1W-EIP21, CJ1W-EIP21, CJ2B-EIP21, CJ2M-EIP21, CJ1W-EIP21(CJ2), CJ1W-EIP21(NJ), NX701, NX102-□□□□, NX1P2, NJ501-□□□□, NJ301-□□□□ or NJ101 CPU Unit. When you create tag data links for other devices, select the connection I/O type specified in that device's EDS file. Use the **Input Only (ID type)** setting when another company's node is the originator and does not support connection settings with a Tag type setting.
Connection Type	Select whether the data is sent in multicast or unicast (point-to-point) form. The default setting is multicast. • Multi-cast connection: Select when the same data is shared by multiple nodes. This setting is usually used. • Point-to-point connection: Select when the same data is not shared by multiple nodes. In a unicast transmission, other nodes are not burdened with an unnecessary load. Refer to *6-1-4 Overview of Operation* on page 6-7 for details on using multi-cast and unicast connections, and counting the number of connections.
The **Connection Structure** Area and the following items are not displayed if the **Hide Detail** Button is clicked.	

Setting	Description
Packet Interval (RPI)	Set the data update cycle (i.e., the packet interval) of each connection between the originator and target. The default setting is 50 ms (i.e., data is updated once every 50 ms). • NX701 CPU Unit: Set the RPI between 0.5 and 10,000 ms in 0.5-ms increments. • NX102 CPU Unit: Set the RPI between 1 and 10,000 ms in 1-ms increments. • NX1P2 CPU Unit: Set the RPI between 2 and 10,000 ms in 0.5-ms increments. • NJ-series CPU Unit: Set the RPI between 1 and 10,000 ms in 1-ms increments.*1
Timeout Value	Set the time elapsed until a connection timeout is detected. The timeout value is set as a multiple of the packet interval (RPI) and can be set to 4, 8, 16, 32, 64, 128, 256, or 512 times the packet interval. The default setting is 4 times the packet interval (RPI).
Connection Name	Set a name for the connection. (32 single-byte characters max.)

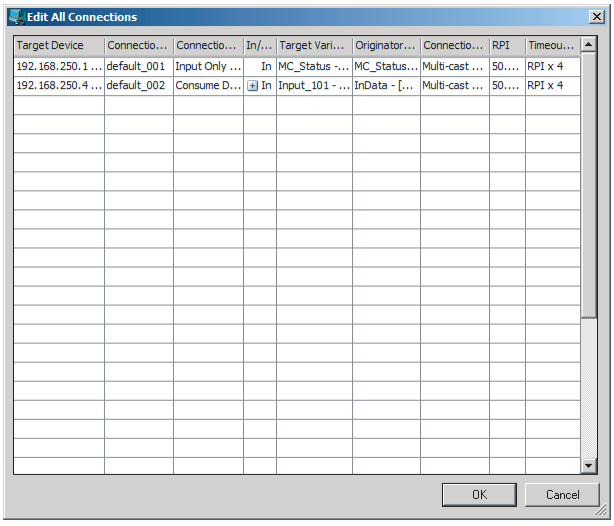
*1. For CPU unit version 1.02 or earlier, you can set the RPI between 10 and 10,000 ms in 1-ms increments.

2 After you make all of the settings, click the **OK** Button.

● Editing Settings for All Connections

You can edit the connection settings between the originator and all of the target devices selected in the Register Device List together in a table.

1 Click the **Connections** Tab, and then click the **Edit All** Button.
The following **Edit All Connections** Dialog Box is displayed.



The settings are as follows:

Setting	Description
Target Device	Select the target device.
Connection Name	Any name can be given to the connection. (32 single-byte characters max.) If this field is left blank, a default name is assigned. The connection name is used as a comment.

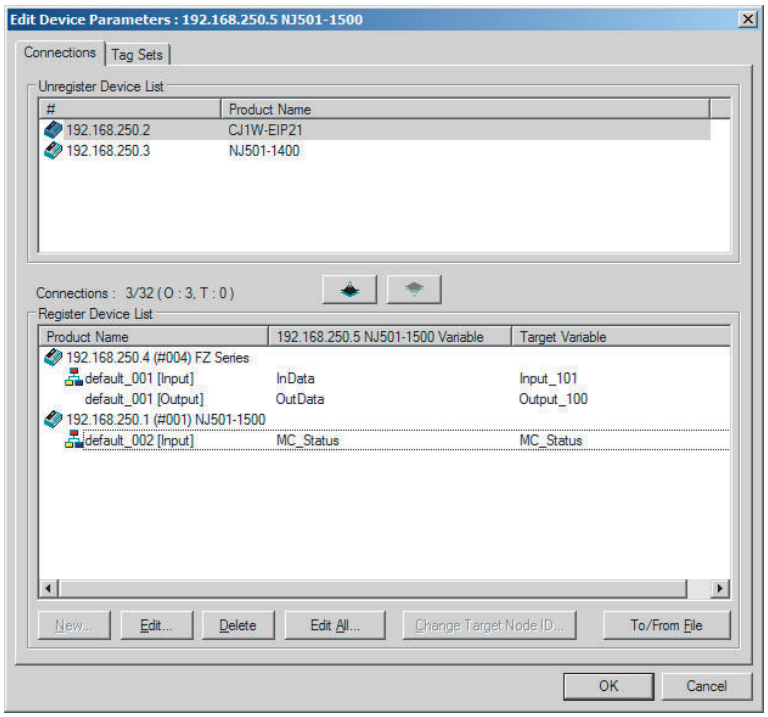
Setting	Description
Connection I/O Type	Select **Input Only (Tag type)** to use tag data links with a CS1W-EIP21, CJ1W-EIP21, CJ2B-EIP21, CJ2M-EIP21, CJ1W-EIP21(CJ2), CJ1W-EIP21(NJ), NX701, NX102-□□□□, NX1P2, NJ501-□□□□, NJ301-□□□□ or NJ101 CPU Unit. When you create tag data links for other devices, select the connection I/O type specified in that device's EDS file. Use the **Input Only (ID type)** setting when another company's node is the originator and does not support connection settings with a Tag type setting.
In/Out	The connection's I/O is automatically displayed based on the selected connection. Input Only: Just In is displayed.
Target Variable	Select the target node's tag set to assign. • In: Select the target's output (produce) tag set. • Out: Select the target's input (consume) tag set.
Originator Variable	Select the originator node's tag set to assign. • In: Select the originator's input (consume) tag set. • Out: Select the originator's output (produce) tag set.
Connection Type	Select whether the data is sent in multi-cast or unicast (point-to-point) form. The default setting is multi-cast. • Multi-cast connection: Select when the same data is shared by multiple nodes. This setting is usually used. • Point-to-point connection: Select when the same data is not shared by multiple nodes. In a unicast transmission, other nodes are not burdened with an unnecessary load. Refer to *6-1-4 Overview of Operation* on page 6-7 for details on using multi-cast and unicast connections, and counting the number of connections.
RPI	Set the data update cycle (i.e., the packet interval) of each connection between the originator and target. The default setting is 50 ms (i.e., data is updated once every 50 ms). • NX701 CPU Unit: Set the RPI between 0.5 and 10,000 ms in 0.5-ms increments. • NX102 CPU Unit: Set the RPI between 1 and 10,000 ms in 1-ms increments. • NX1P2 CPU Unit: Set the RPI between 2 and 10,000 ms in 0.5-ms increments. • NJ-series CPU Unit: Set the RPI between 1 and 10,000 ms in 1-ms increments.*1
Timeout Value	Set the time elapsed until a connection timeout is detected. The timeout value is set as a multiple of the packet interval (RPI) and can be set to 4, 8, 16, 32, 64, 128, 256, or 512 times the packet interval. The default setting is 4 times the packet interval (RPI).

*1. For CPU unit version 1.02 or earlier, you can set the RPI between 10 and 10,000 ms in 1-ms increments.

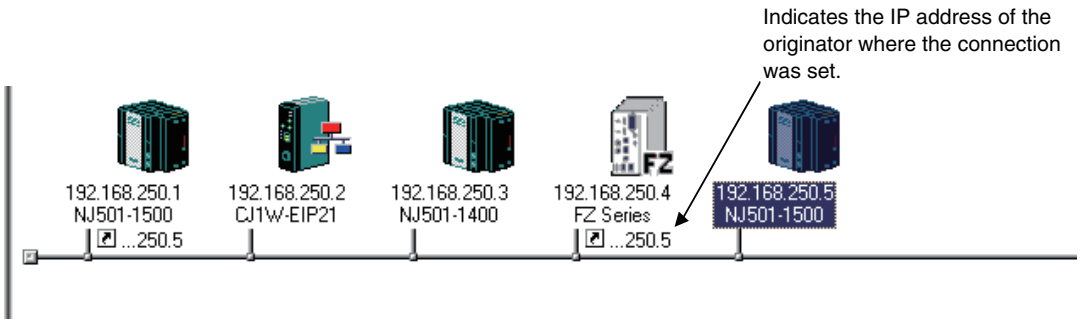
2 After you make all of the settings, Click the **OK** Button.

● Confirming the Connection Settings

1 An overview of the connections that were set in the Register Device List is displayed in the Connections Tab Page.



2 Click the **OK** Button. The following figure is displayed.



3 Repeat the connections setting procedure until all of the connections are set.

Precautions for Correct Use

After you have made all of the settings, always click the **OK** Button before you close the **Edit Device Parameters** Dialog Box. If the **Cancel** Button is clicked and the dialog box is closed, all the settings you made here are discarded.

4 If you change the size of a tag set for the originator or a target node after the connection settings, a parameter data mismatch will occur due to the size difference between them.
if you change the connection settings, be sure to check the connections. (Refer to *6-2-16 Checking Connections* on page 6-77 for details.)

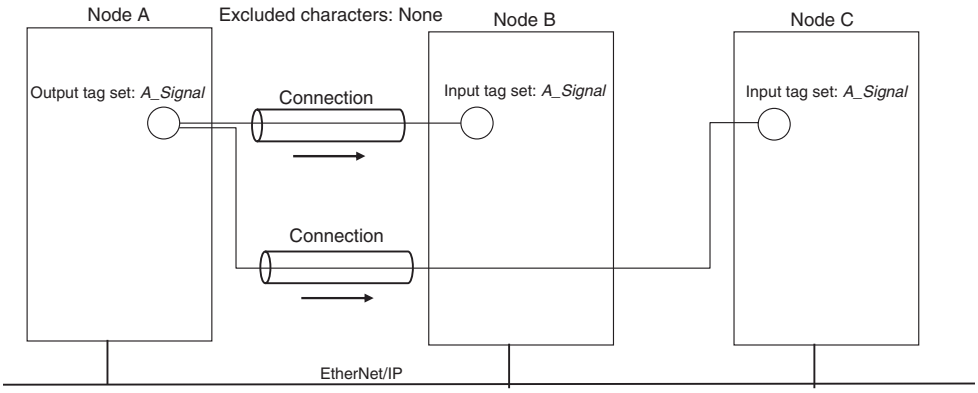
Automatically Setting Connections (Network - Auto Connection)

You can use automatic detection of the tag set names that are set for devices to automatically set connections between input and output tag sets with the same name (or the same names excluding specified ellipses).
Connections are automatically set under the following conditions.

Output tag set names for connection setting	Except for specified ellipses, the output tag set name must be the same as the input tag set name. Ellipses can be set for the beginning or end of tag set names.
Input tag set names for connection settings	Except for specified ellipses, the input tag set name must be the same as the output tag set name. Ellipses can be set for the beginning or end of tag set names.
Connection type	The connection I/O type must be Input Only. Multicast or unicast connections can be specified for a connection.
RPI	The default setting is used.
Timeout	The default setting is used.

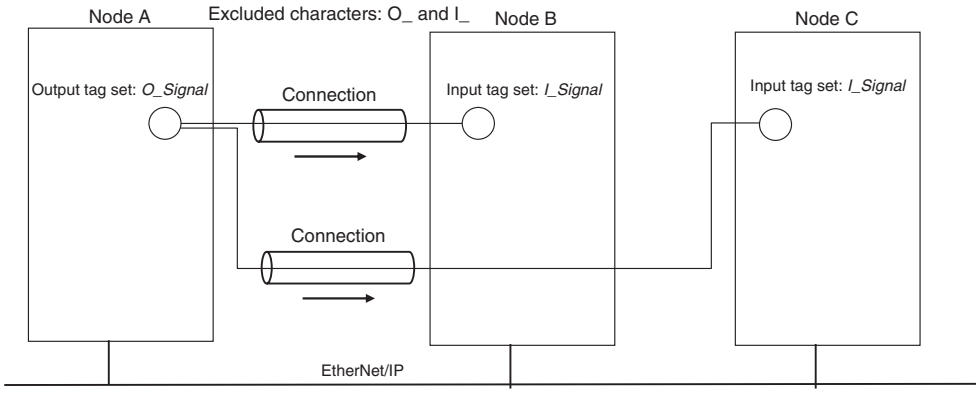
Example 1: Automatic Connections with the Same Tag Set Names

The following connections are automatically set with the same tag set name (*A_Signal*) if there is an output (produce) tag set named *A_Signal* at node A, and input (consume) tag sets named *A_Signal* at nodes B and C.



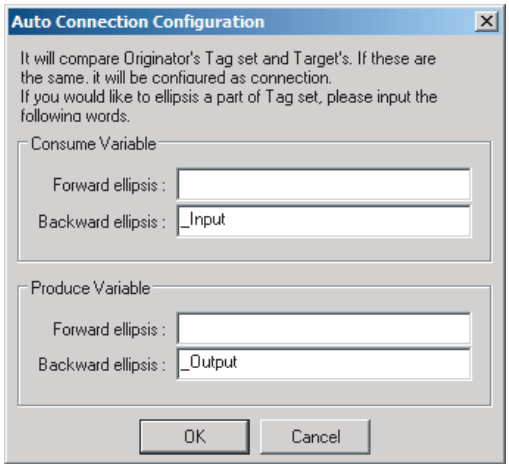
Example 2: Automatic Connections with the Ellipses

The following connections are automatically set with the same tag set name (*Signal*) if there is an output (produce) tag set named *O_Signal* at node A, and input (consume) tag sets named *I_Signal* at nodes B and C, and *O_* and *I_* are set as forward ellipses.



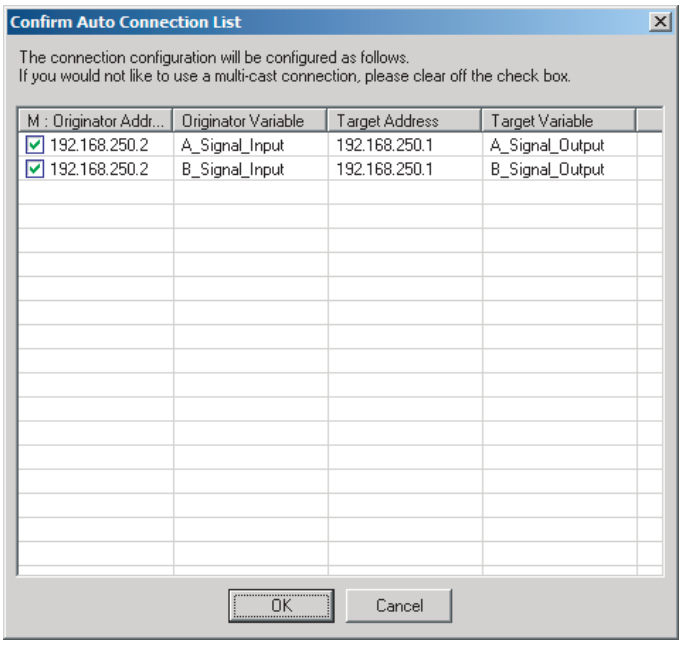
1 Set the same tag set names for the output and input tag sets for the connection. The tag set names can also include forward and backward ellipses.

2 Select **Auto Connection Configuration** from the **Network** Menu.
A dialog box will appear to set forward and backward ellipses for both output and input tag sets as soon as automatic connection setting processing starts.



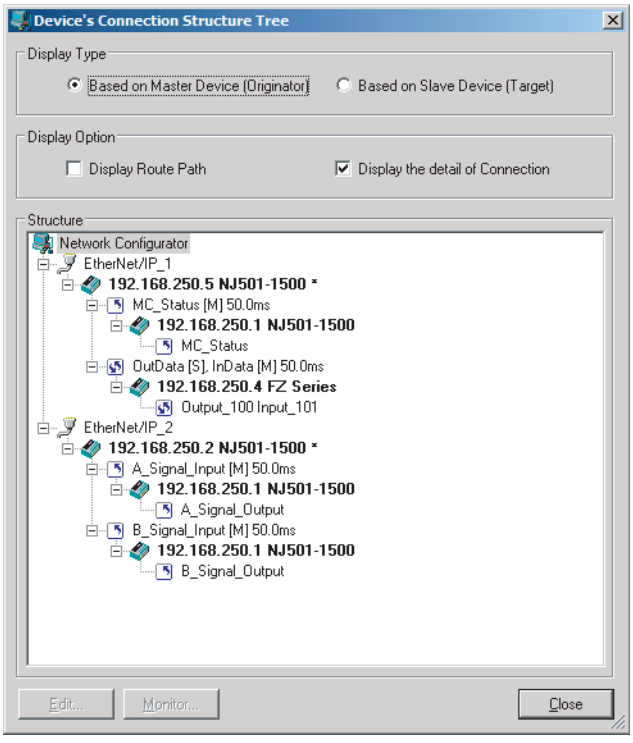
Input the ellipses and click the **OK** Button. Processing for automatic setting is started.

3 If there are tag sets that meet the conditions for automatic connection setting, they are displayed.



Click the **OK** Button. Processing for automatic setting is started.

4 A device connection structure tree is displayed when processing is completed.

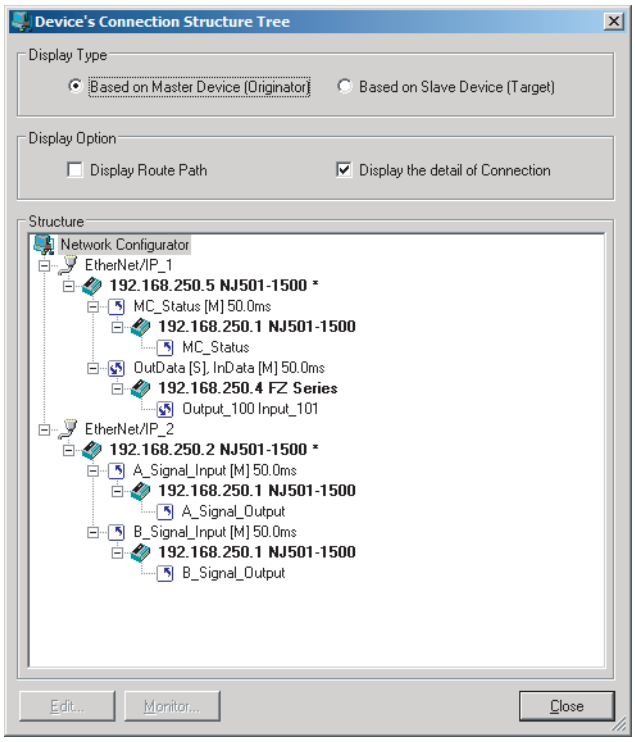


5 Use the device connection structure tree to change the RPI and timeout settings if necessary.

Device Connection Structure Tree

Connection settings can be displayed on the network configuration.

Select **View Device's Connection Structure Tree** from the **Network** Menu.



- You can check the **Display the detail of Connection** Check Box to switch between device-level and connection-level views of tag data link communications.
- An asterisk is displayed after the device name of the originator set for the connection.
- The **Edit Device Parameters** Dialog Box is displayed if you select a connection and click the **Edit** Button. You can edit the connections in this dialog box.

6-2-6 Creating Connections Using the Wizard

You can use the Network Configurator's Wizard to easily create connections between OMRON PLCs following the instructions provided by the Wizard.

Additional Information

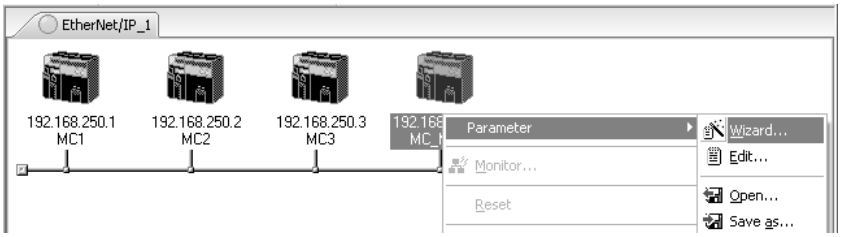
The Wizard can be used only with the following OMRON EtherNet/IP devices.

Device name	Remarks
CJ1W-EIP21 (NJ)	CJ1W-EIP21 mounted to NJ-series Controller
CJ1W-EIP21	CJ1W-EIP21 mounted to CJ1 CPU Unit
CJ1W-EIP21 (CJ2)	CJ1W-EIP21 mounted to CJ2 CPU Unit
CJ2B-EIP21	Built-in EtherNet/IP port in CJ2H CPU Unit
CJ2M-EIP21	Built-in EtherNet/IP port in CJ2M CPU Unit
CS1W-EIP21	CS1W-EIP21 mounted to CS1 CPU Unit
NX701	Built-in EtherNet/IP port on NX-series CPU Unit
NX102-□□□□	
NX1P2	
NJ501-□□□□	Built-in EtherNet/IP port on NJ-series CPU Unit
NJ301-□□□□	
NJ101	

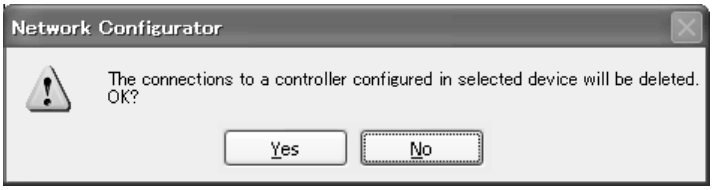
Use the following procedure to create connections (i.e., tag data links) with the Wizard.

1 Set tags and tag sets for all the devices before starting the Wizard. Refer to *6-2-4 Creating Tags and Tag Sets* on page 6-23 for the setting procedure.

2 For tag data links between OMRON PLCs, a connection is created in the PLC (i.e., the originator device) that receives data as input data. First, select the registered device for which you want to create a connection in the Network Configuration Window of the Network Configurator, and then select **Device** - **Parameters** - **Wizard** from the menu.

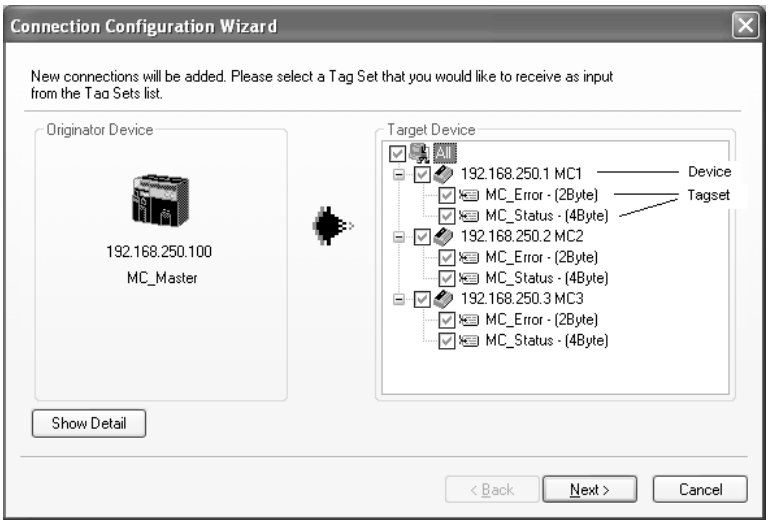


The following message box will be displayed before the Wizard starts.



Click the **Yes** Button to delete the connections that are set with OMRON PLCs before starting the Wizard.

3 Create the connection following the instructions that are given by the Wizard after the Wizard starts. (See the following figure.)



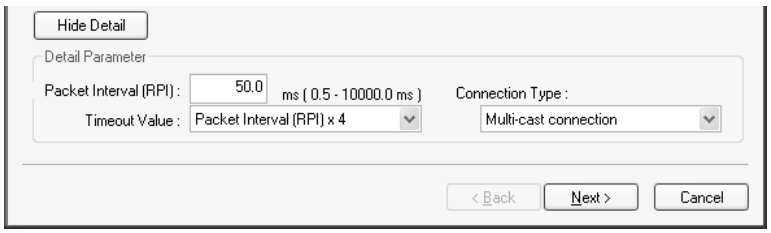
4 A list of tag sets is displayed on the right side of the Wizard with target devices that support receiving input data.
Select the tag sets that you want to receive at the originator device.

The following table describes the meanings of the icons and check marks displayed in the tag set list.

Icon	Display position	Status
☑	All	All output tag sets for all devices are selected.
	Device	All output tag sets for the applicable device are selected.
	Tag set	The applicable output tag sets are selected. These are the tag sets that will be set in the connection.
☑	All	All or some output tag sets for some devices are selected.
	Device	Some output tag sets for applicable devices are selected.
☐	All	All output tag sets for all devices are not selected.
	Device	All output tag sets for applicable devices are not selected.
	Tag set	The applicable output tag sets are not selected. The connections for this tag set will be deleted.
☐	Device	No applicable tag sets.

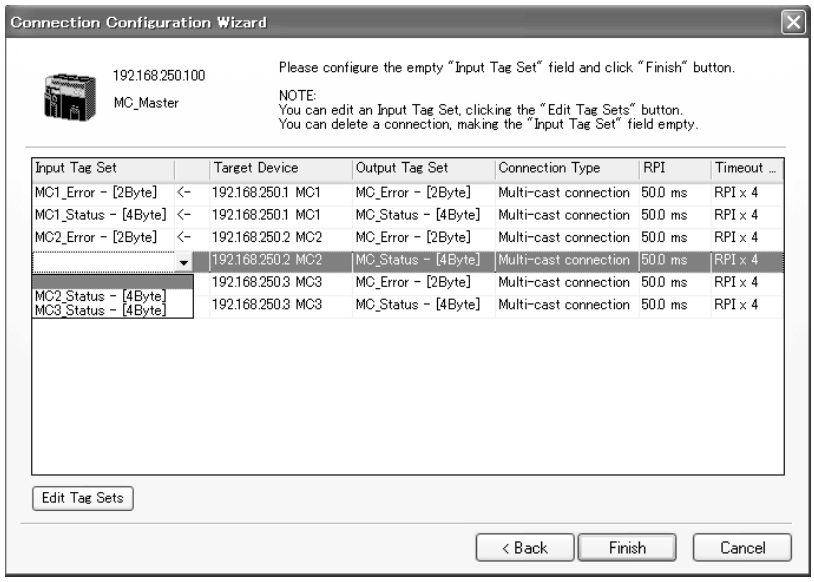
Note Tag sets used in connections that are already set are not displayed.

The following display will appear when you click the **Show Detail** Button.



The preset values for detailed parameters will be displayed. Change the values as required. The connection name cannot be set. They are automatically created using the following rule.
default_N (where N is a 3-digit number (001, 002, etc.) starting from 1)

5 Click the **Next** Button to switch to the table in the following Wizard Dialog Box. Follow the instructions to select the input tag set of the originator device that receives the output tag set of the target device from the list box.



- The blank area in the Input Tag Set Column is for the connection that you are creating.
- For the connections that are already set, values are already given in the Input Tag Set Column.
- To prevent duplicate settings, input tag sets that are used are not displayed in the list box for input tag sets.
- If there is no applicable input tag set, you can edit a tag set or create a new one by using the **Edit Tag Sets** Button and the **Edit Tag** Button.

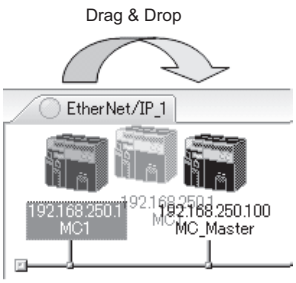
6 Once the input tag set settings are completed, click the **Finish** Button. You can check the set connection by selecting **Network - View Devices Connection Structure Tree** from the menu.

- The Wizard can be ended even if the input tag set includes a blank row. In that case, a connection is not created for the blank row.
- You can delete a connection by deleting the input tag sets that were previously set.

6-2-7 Creating Connections by Dragging and Dropping Devices

You can create a connection to the originator by dragging a target device and dropping it at the originator device.

Example) Drag the target device at 192.168.250.1 and drop it at the originator device at 192.168.250.100.



Additional Information

The EtherNet/IP originator device (i.e., a device in which connections can be set) must be one of the following OMRON EtherNet/IP devices.

Device name	Remarks
CJ1W-EIP21 (NJ)	CJ1W-EIP21 mounted to NJ-series Controller
CJ1W-EIP21	CJ1W-EIP21 mounted to CJ1 CPU Unit
CJ1W-EIP21 (CJ2)	CJ1W-EIP21 mounted to CJ2 CPU Unit
CJ2B-EIP21	Built-in EtherNet/IP port in CJ2H CPU Unit
CJ2M-EIP21	Built-in EtherNet/IP port in CJ2M CPU Unit
CS1W-EIP21	CS1W-EIP21 mounted to CS1 CPU Unit
NX701	Built-in EtherNet/IP port on NX-series CPU Unit
NX102-□□□□	
NX1P2	
NJ501-□□□□ NJ301-□□□□ NJ101	Built-in EtherNet/IP port on NJ-series CPU Unit

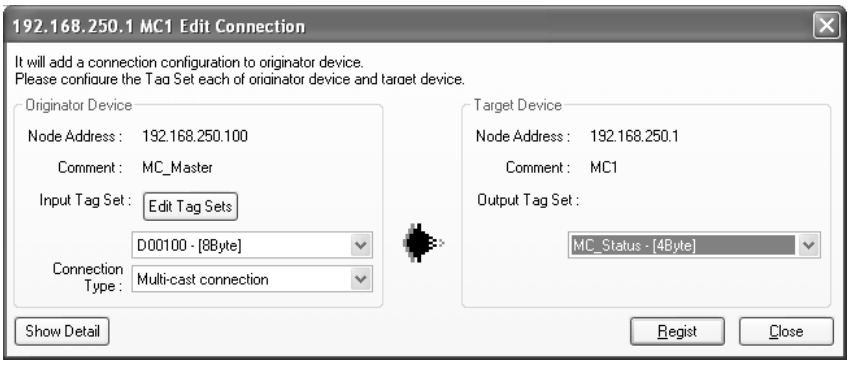
Use the following procedure to create connections (i.e., tag data links) by dragging and dropping devices.

1 Set the tags and tag sets for the target device that will be dragged.

- Refer to *6-2-4 Creating Tags and Tag Sets* on page 6-23 for information on the settings if the target is one of the OMRON EtherNet/IP devices given above.
- If the target is another EtherNet/IP device, refer to the manual of that device and perform settings as required.

2 A dialog box as in the following figure for connection allocation will be displayed when you drag the target device and drop it at the OMRON EtherNet/IP device.

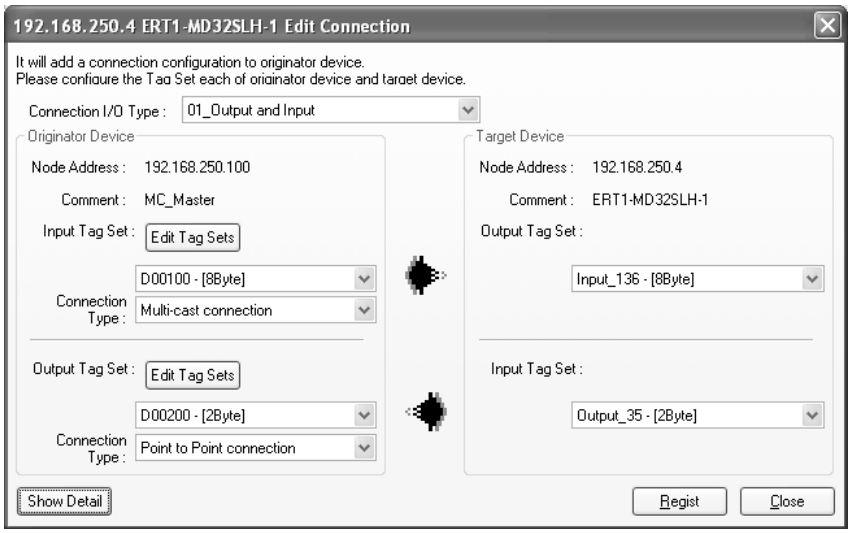
- Using One of the Above OMRON EtherNet/IP Devices As Target



Select an output tag set from the **Target Device** Area on the right side of the **Edit Connection** Dialog Box, and then select an input tag set to receive the output tag set in the **Originator Device** Area on the left.

- If there is no applicable input tag set at the originator, you can create a new one by using the **Edit Tag Sets** Button and the **Edit Tag** Button.

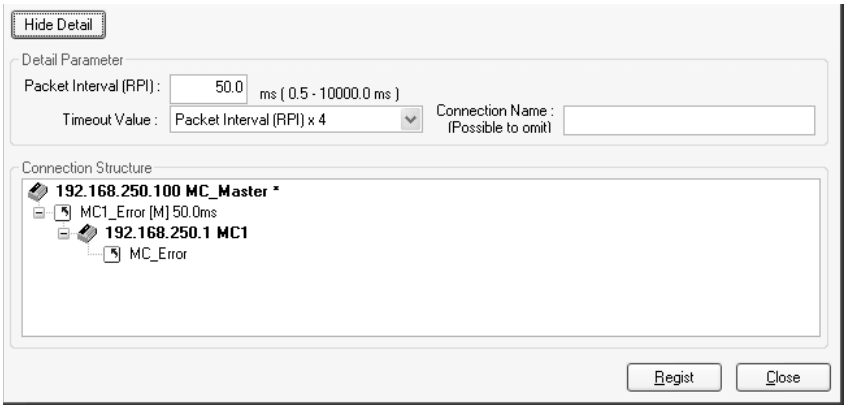
- Using Other EtherNet/IP Devices as Target



The **Connection I/O Type** list box in the upper part of the **Edit Connection** Dialog Box lists connection I/O types. Select a connection I/O type according to your application.

- The connection I/O types that can be selected depend on the target device.
- Items that can be selected depend on the connection I/O type that is selected.
- Select the output, input, or both output and input tag sets at the target and specify the corresponding input, output, or both input and output tag sets at the originator.
- If there is no applicable tag set at the originator, you can create a new one by using the **Edit Tag Sets** Button and the **Edit Tag** Button.

The following view will appear when you click the **Show Detail** Button.



The specified values for detailed parameters will be displayed. Change the values as required.
Connection names are automatically created using the following rule.
default_N (where N is a 3-digit number (001, 002, etc.) starting from 1)

Additional Information

The following dialog box will be displayed if a target device that does not have I/O data is dropped.

Before dropping again, refer to the manual of the applicable device and create the I/O data (i.e., output tag sets) required to create a connection.

3 After you complete the settings, click the **Regist** Button to create the connection. When the connection is completed, the input tag set box and the output tag set box will be blank. You can continue to create another connection by selecting a next connection I/O type and setting a tag set.

6-2-8 Connecting the Network Configurator to the Network

This section describes how to connect the Network Configurator to the network.

Precautions for Correct Use

Connection may not be possible if the following settings are made on an NJ/NX-series Controller on the connection path or on a connection destination NJ/NX-series Controller. If connection fails, check the following settings. For the details on the settings, refer to *CIP Message Server* on page 4-19 and *Packet Filter* on page 4-7.

- The **Do not use** Option is selected for the CIP message server.
- The **Use** Option is selected for Packet Filter.

Additional Information

Although NX102 and NX701 CPU Units provide two EtherNet/IP ports, the Network Configurator treats these two ports as two different Units and connects them individually.

Connecting through Ethernet

Connect to the built-in EtherNet/IP port on the CPU Unit via an Ethernet switch.

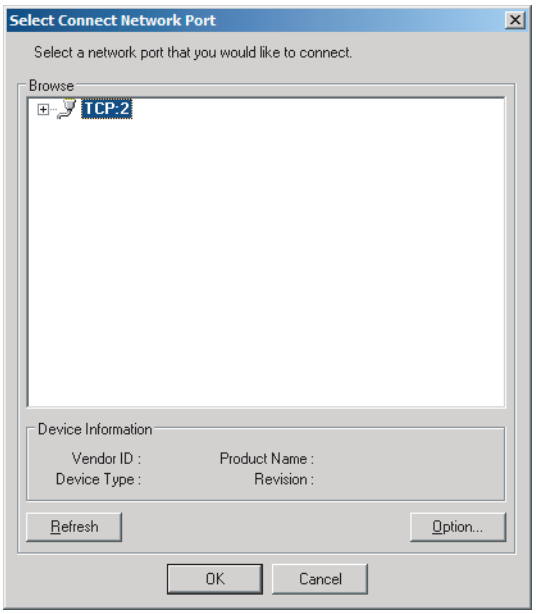
Precautions for Correct Use

The first time you connect via Ethernet with Windows XP (SP2 or higher), Windows Vista, or Windows 7, you must change the Windows firewall settings.
For the procedure, refer to *A-4 Precautions for Using the Network Configurator on Windows XP, Windows Vista, or Windows 7 or Higher* on page A-45.

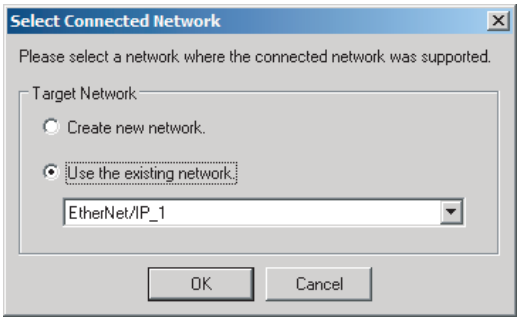
1 Select **Option** - **Select Interface** - **Ethernet I/F**.

2 Select **Network** - **Connect**.

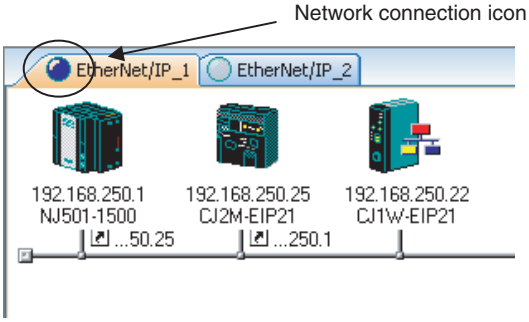
If there are multiple Ethernet interfaces on the computer, the **Select Interface** Dialog Box is displayed. Select the interface to connect, and press the **OK** Button.
The following dialog box is displayed.



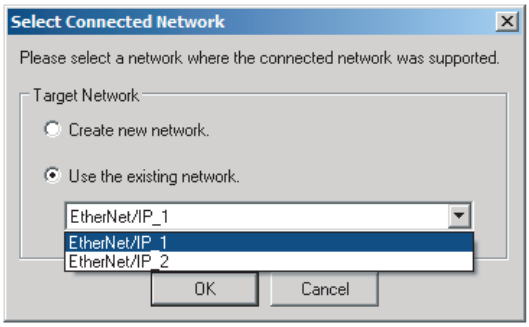
3 Click the **OK** Button.
Select the network to connect to.



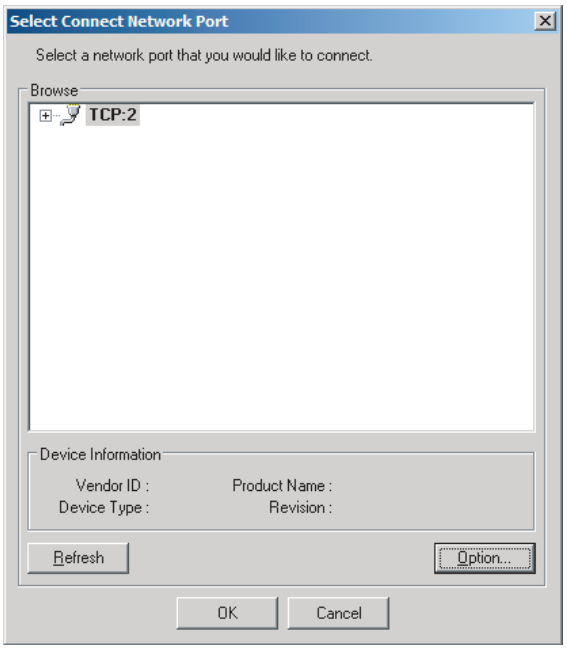
The Network Configurator will connect to the EtherNet/IP network. If the Network Configurator goes online normally, **On-line** is displayed in the status bar at the bottom of the window.
The network connection icon is displayed in blue on the Network Tab Page in which the Network Configurator is connected.



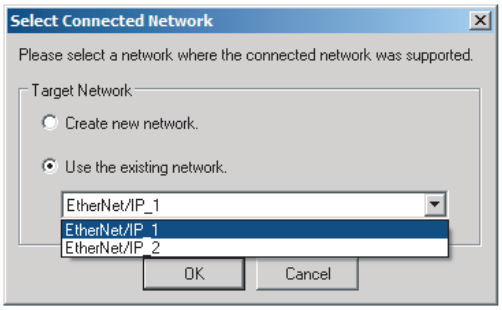
Select **Network** - **Change Connect Network** to switch the connected network.



4 The following dialog box is displayed.

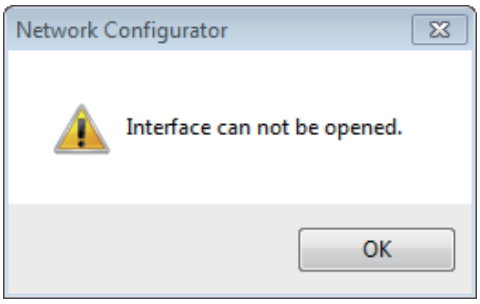


5 Click the **OK** Button.
Select the network to connect to.



Additional Information

If the following dialog box appears in the Network Configurator when you go online with an NJ/NX-series CPU Unit, refer to the following table for possible causes and corrections.



Assumed cause	Correction
The cable is not connected correctly.	Check if the cable is disconnected or loose.
Connection with the Controller is blocked due to the firewall settings.	If connection with the Controller is blocked due to the firewall settings, disable the blocking. For the firewall settings, refer to *A-4 Precautions for Using the Network Configurator on Windows XP, Windows Vista, or Windows 7 or Higher* on page A-45.
Communications with Network Configurator are blocked due to Packet Filter of the Controller.	Allow communications with Network Configurator. For details on Packet Filter settings, refer to *Packet Filter* on page 4-7.
The server function of CIP message communications is disabled.	Enable the server function of CIP message communications. Refer to *CIP Message Server* on page 4-19 for details on setting CIP message server.

Connections through CPU Unit's USB Port

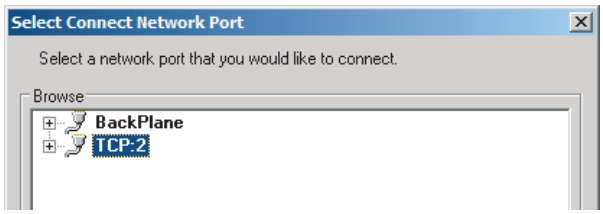
Use the following procedure to connect to the built-in EtherNet/IP port via the USB port on the CPU Unit.

Precautions for Correct Use

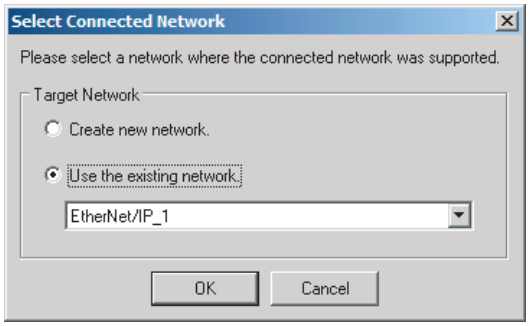
NX701 CPU Units of hardware revision A or later and CPU Units of NX102 and NX1P2 do not support connections via USB port.

1 Select the communications interface.
Select **Option** - **Select Interface** - **NJ/NX Series USB Port**.

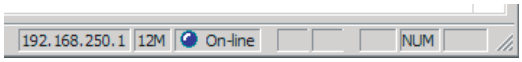
2 Select **Network** - **Connect**.
The following dialog box is displayed.



3 Select **TCP:2** and then click the **OK** Button.
The following dialog box is displayed.

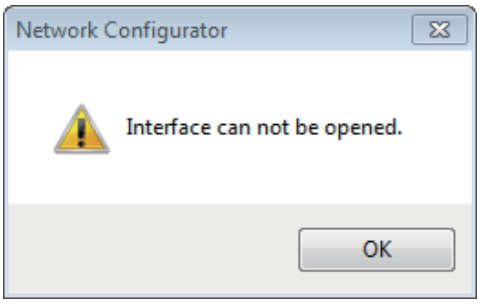


4 Select the network to connect and click the **OK** Button.
The Network Configurator will connect to the EtherNet/IP network. If the Network Configurator goes online normally, **On-line** is displayed in the status bar at the bottom of the window.



Additional Information

If the following dialog box appears in the Network Configurator when you go online with an NJ/NX-series CPU Unit, refer to the following table for possible causes and corrections.



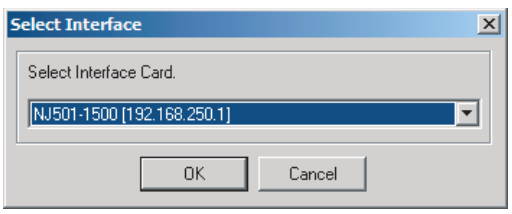
Assumed cause	Correction
The cable is not connected correctly.	Check if the cable is disconnected or loose.
Connection with the Controller is blocked due to the firewall settings.	If connection with the Controller is blocked due to the firewall settings, disable the blocking. For the firewall settings, refer to *A-4 Precautions for Using the Network Configurator on Windows XP, Windows Vista, or Windows 7 or Higher* on page A-45.
The USB driver is not installed correctly.	Install the USB driver correctly. For how to install the USB driver, refer to the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)*.
The server function of CIP message communications is disabled.	Enable the server function of CIP message communications. Refer to *CIP Message Server* on page 4-19 for details on setting CIP message server.

Direct Connection via Ethernet to Built-in EtherNet/IP Port

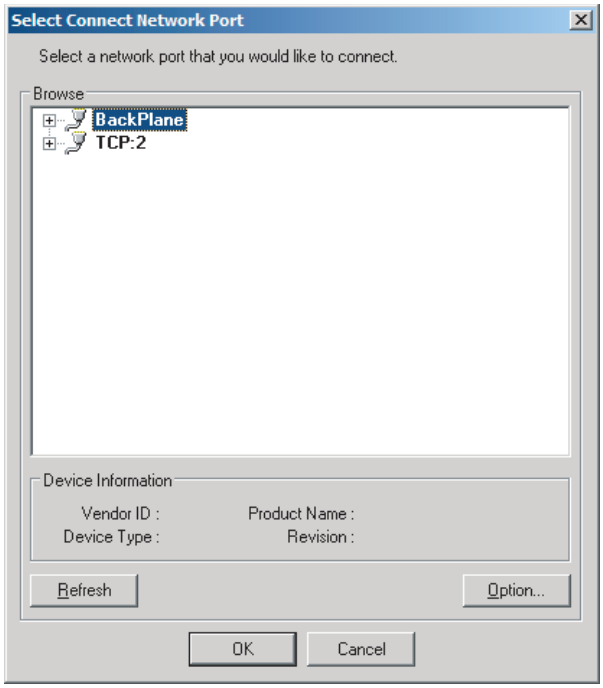
Use the following procedure to directly connect to a built-in EtherNet/IP port on an NJ/NX-series CPU Unit via Ethernet.
You can connect to the built-in EtherNet/IP port even if the IP address is not set on the computer.

1 Select the communications interface.
Select **Option** - **Select Interface** - **NJ/NX Series Ethernet Direct I/F**.

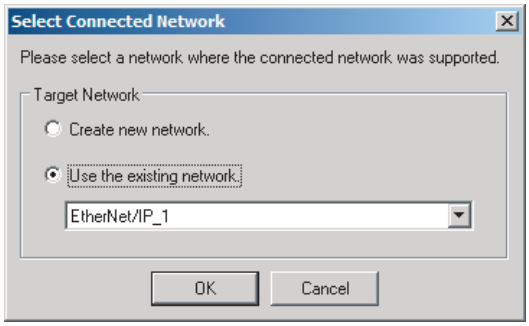
2 Select **Network** - **Connect**.
The **Select Interface** Dialog Box is displayed if there are several CPU Units that you can connect to.



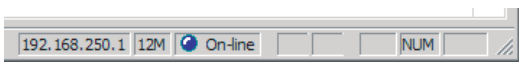
3 Select the Interface Card to connect and click the **OK** Button.
When you select one of the options listed as **CPU Unit model (IP number)**, the following dialog box is displayed.



4 Select **TCP:2** and then click the **OK** Button.
The following dialog box is displayed.

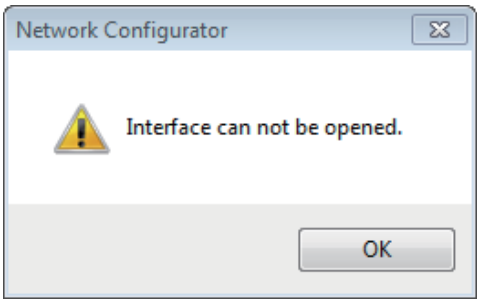


5 Select the network to connect to.
The Network Configurator will connect to the EtherNet/IP network. If the Network Configurator goes online normally, **On-line** is displayed in the status bar at the bottom of the window.



Additional Information

If the following dialog box appears in the Network Configurator when you go online with an NJ/NX-series CPU Unit, refer to the following table for possible causes and corrections.



Assumed cause	Correction
The cable is not connected correctly.	Check if the cable is disconnected or loose.
Connection with the Controller is blocked due to the firewall settings.	If connection with the Controller is blocked due to the firewall settings, disable the blocking. For the firewall settings, refer to *A-4 Precautions for Using the Network Configurator on Windows XP, Windows Vista, or Windows 7 or Higher* on page A-45.

6-2-9 Downloading Tag Data Link Parameters

To make tag data links, you must download tag data link parameters, such as tag set settings and connection settings, to all devices in the EtherNet/IP network.
When the download operation is executed, the tag data link parameters are transferred to the EtherNet/IP devices that require the settings.

The following procedure shows how to download the tag data link parameters.
For details on how to connect to the network from the Network Configurator, refer to *6-2-8 Connecting the Network Configurator to the Network* on page 6-52.

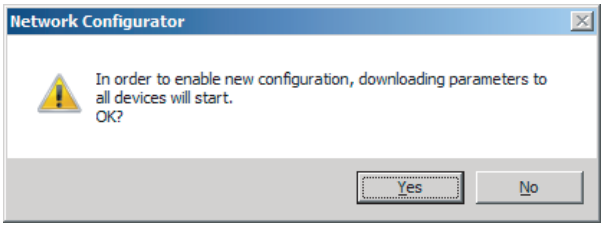
Precautions for Correct Use

- If the node addresses (IP addresses) are not set correctly, you may connect to the wrong Controller and set incorrect device parameters. Download data only after you confirm that you are connected to the correct Controller.
- If incorrect tag data link parameters are set, it may cause equipment to operate unpredictably. Even when the correct tag data link parameters are set, make sure that there will be no effect on equipment before you transfer the data.
- When network variables are used in tag settings, a connection error will result if the variables are not set in the CPU Unit. Before downloading the tag data link parameters, check to confirm that the network variables are set in the CPU Unit. Check whether the network variable, tag, and connection settings are correct on the **Connection** Tab Page and the **Tag Status** Tab Page as described in *15-2-1 The Network Configurator's Device Monitor Function* on page 15-3.
- If a communications error occurs, the output status depends on the specifications of the device being used. When a communications error occurs for a device that is used along with output devices, check the operating specifications and implement safety countermeasures.
- The built-in EtherNet/IP port is automatically restarted after the parameters are downloaded. This restart is required to enable the tag set and connection information. Before you download the parameters, make sure that restarting the port will not adversely affect the controlled system.
- Make sure that the major CIP revision of the device registered with the Network Configurator is the same as the major CIP revision of the CPU Unit that you use. If the major CIP revisions are not the same, the parameters may not be downloaded. To determine whether downloading is possible, refer to *6-2-3 Registering Devices* on page 6-21.
- Do not disconnect the Ethernet cable or reset or turn OFF the power to the EtherNet/IP Unit during the parameter download.
- Tag data links (data exchange) between relevant nodes are stopped during a download. Before you download data in RUN mode, make sure that it will not adversely affect the controlled system.
  Also implement interlocks on data processing in ladder programming that uses tag data links when the tag data links are stopped or a tag data link error occurs.
- For EtherNet/IP Units with revision 1, you can download tag data link parameters only when the CPU Unit is in PROGRAM mode.
- Even for Units with revision 2 or later, all CPU Units must be in PROGRAM mode to download the parameters if any Units with revision 1 are included in the network.

1 Connect the Network Configurator to the network.

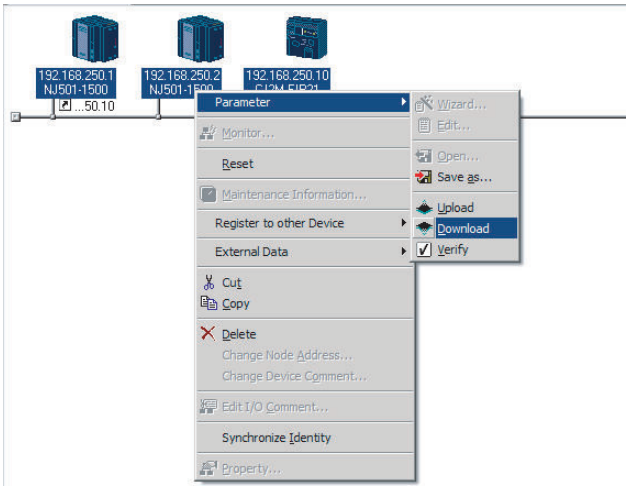
2 There are two ways to download the parameters.

- Downloading to All Devices in the Network
  Select **Network** - **Download**.
  The following dialog box is displayed.

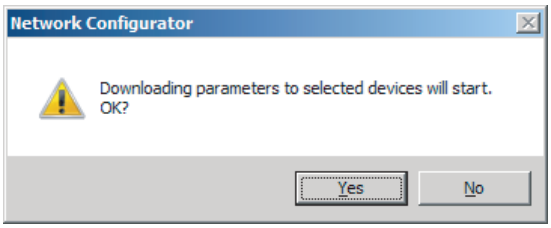


- Downloading Individually to Particular Devices
  Select the icon of the EtherNet/IP Unit to which you want to download. To select multiple nodes, hold down the Shift Key or the Ctrl Key while you click the icons. (In the following example, 2 nodes are selected: 192.168.250.1 and 192.168.250.2.)

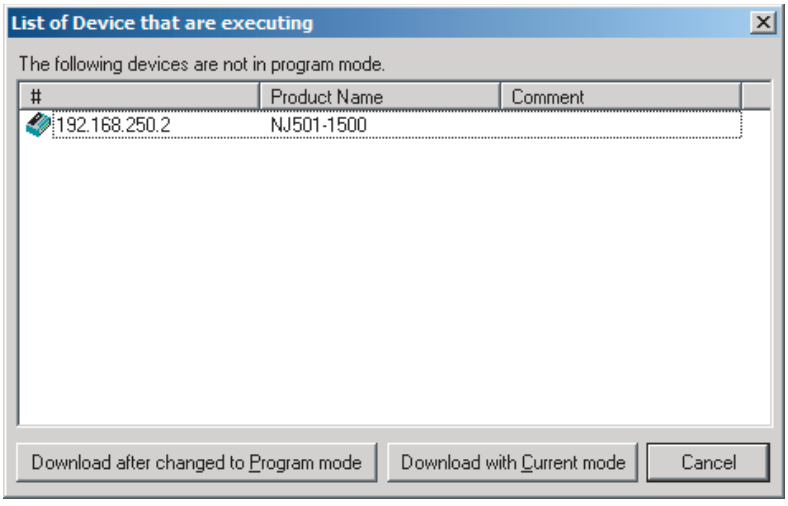
Right-click the icon to display the popup menu, and select **Parameter** - **Download**.



The following dialog box is displayed.

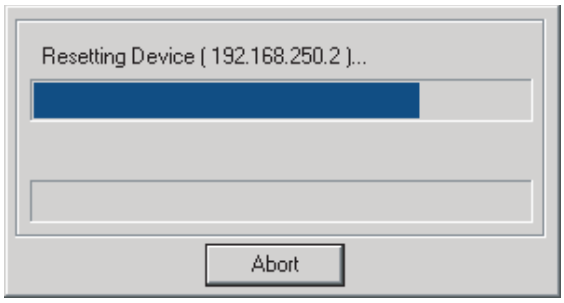


3 Click the **Yes** Button to download the tag data link parameters to the EtherNet/ IP Unit.
The following dialog box is displayed if any of the CPU Units is not in PROGRAM mode.

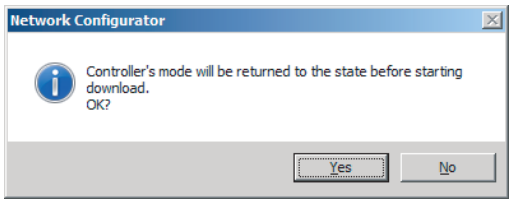


If the **Download after changed to Program mode** Button is clicked, all CPU Units are changed to PROGRAM mode and the parameters are downloaded. Confirm safety for all controlled equipment before you change the CPU Units to PROGRAM mode. You can restore the operating modes after the parameters are downloaded.
You can click the **Download with Current mode** Button to download the parameters even when one or more CPU Units is in RUN mode.
The **Download with Current mode** Button is disabled if the EtherNet/IP Unit does not support the **Download with Current mode** Button (e.g., revision 1 of CJ1W-EIP21 or CS1W-EIP21).

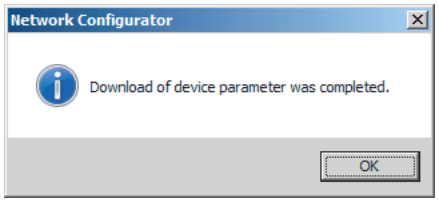
During the download, the following progress indicator is displayed to show the progress of the download.



If the operating mode of one or more CPU Units was changed to download the parameters, you can return the CPU Units to the previous operating modes. If the **No** Button is clicked, the CPU Units remain in PROGRAM mode.



4 The following dialog box is displayed to show that the download was completed.



6-2-10 Uploading Tag Data Link Parameters

You can upload tag data link parameters (such as tag set settings and connection settings) from EtherNet/IP devices in the EtherNet/IP network.
The following procedure shows how to upload the parameters. For details on how to connect to the network from the Network Configurator, refer to *6-2-8 Connecting the Network Configurator to the Network* on page 6-52.

Precautions for Correct Use

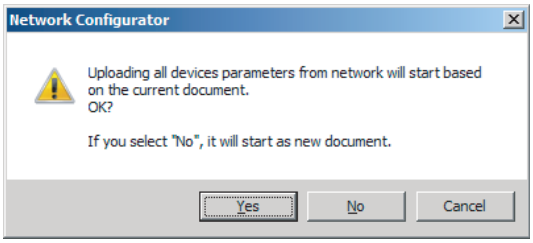
- Make sure that the major CIP revision of the device registered with the Network Configurator is the same as the major CIP revision of the NJ/NX-series CPU Unit that you use. If the major CIP revisions are not the same, the parameters may not be uploaded. To determine whether uploading is possible, refer to *6-2-3 Registering Devices* on page 6-21.

There are two ways to upload the parameters.

Uploading from All Devices in the Network

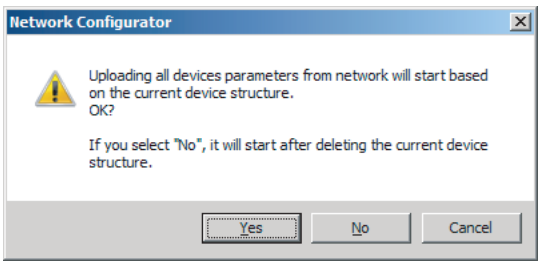
1 Connect the Network Configurator online, and then select **Upload** from the **Network** Menu.

2 The following dialog box is displayed.

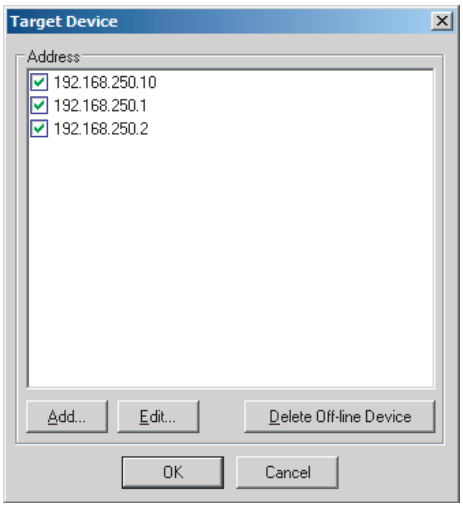


- Clicking the **Yes** Button:
  The tag data link parameters in the current project are uploaded.
- Clicking the **No** Button:
  You open a new project to upload the tag data link parameters. The current project is closed.
- Clicking the **Cancel** Button:
  The upload operation is canceled. The upload is not performed.

3 If you click the **Yes** Button in step 2, the following dialog box is displayed.

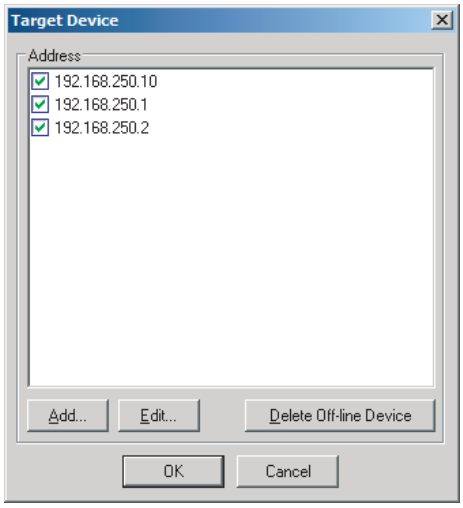


- Clicking the **Yes** Button:
  Parameters are uploaded only from the devices registered in the Network Configuration Pane. Parameters are not uploaded from devices that are not registered in the Network Configuration Pane.
- Clicking the **No** Button:
  Performing a Batch Upload over the Network
  Parameters are uploaded from all devices on the network.
  The current Network Configuration Information will be lost.
  The following dialog box will be displayed. Select the devices for which to upload parameters and click the **OK** Button.



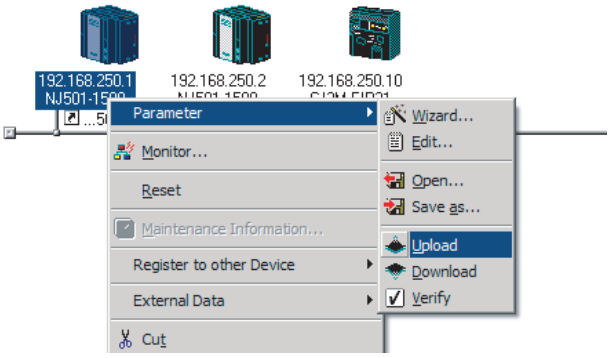
- Clicking the **Cancel** Button:
  The upload operation is canceled. The upload is not performed.

4 If you click the **No** Button in step 2, the following dialog box is displayed.
Select the devices for which to upload parameters and click the **OK** Button.

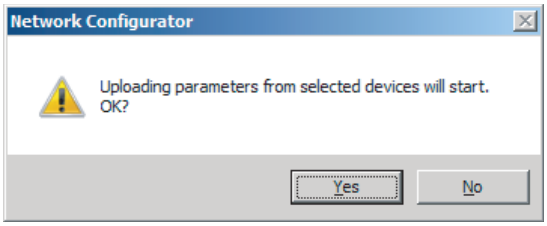


Uploading Individually from Particular Devices

1 Connect the Network Configurator to the network.
Select the icon of the EtherNet/IP Unit from which you want to upload parameters. To select multiple nodes, press and hold the Shift Key or the Ctrl Key while you select additional icons. (In the following example, 2 nodes are selected: 192.168.250.1 and 192.168.250.2.)
Right-click the icon to display the pop-up menu, and select **Parameter** - **Upload**.

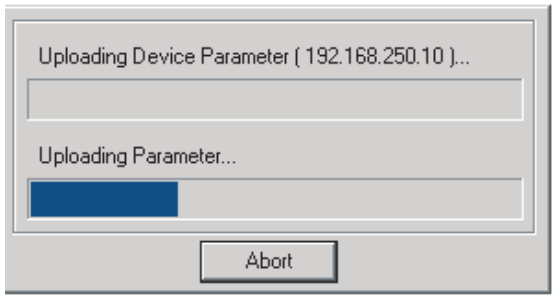


2 The following dialog box is displayed.

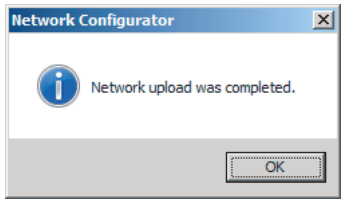


Click the **Yes** Button or the **No** Button.

3 During the upload, the following progress indicator is displayed to show the progress of the upload.



4 The following dialog box is displayed to show that the upload was completed.



6-2-11 Verifying Tag Data Link Parameters

Tag data link parameters (such as tag set settings and connection settings) can be compared with the parameters of the built-in EtherNet/IP ports in the EtherNet/IP network.

The following procedure shows how to compare the parameters. For details on how to connect to the network from the Network Configurator, refer to *6-2-8 Connecting the Network Configurator to the Network* on page 6-52.

Precautions for Correct Use

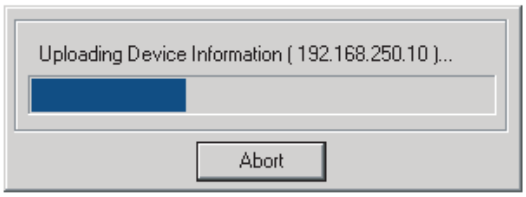
- Make sure that the major CIP revision of the device registered with the Network Configurator is the same as the major CIP revision of the NJ/NX-series CPU Unit that you use. If the major CIP revisions are not the same, the parameters may not be compared. To determine whether comparison is possible, refer to *6-2-3 Registering Devices* on page 6-21.

Verifying the Network Configuration

You can use the following procedure to compare the list of registered devices in the Network Configuration Pane with the devices connected on the EtherNet/IP network, and check the IP addresses and device types.
This function does not verify device parameters.

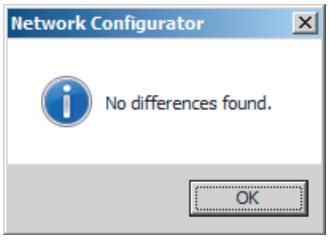
1 Connect the Network Configurator to the network.

2 Select **Network** - **Verify Structure**.
The following progress indicator is displayed to show the progress as data is read from the network and compared.

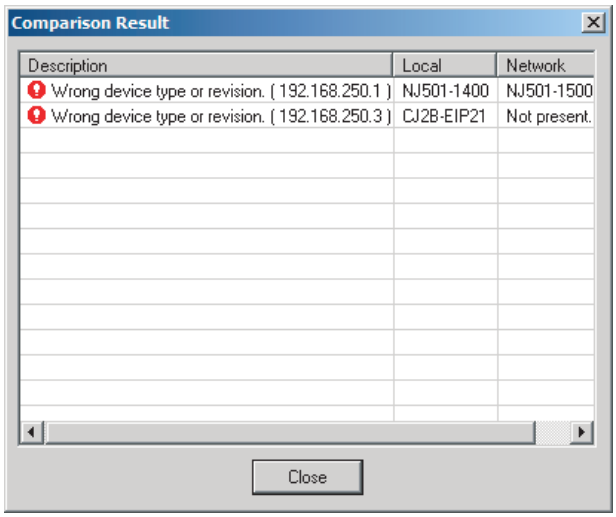


3 The result of the comparison between the network configuration file and data from the network is displayed as shown below.

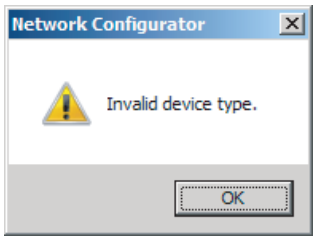
- Differences Not Found in the Comparison



- Differences Found in the Comparison



- Differences Found in the Device Type



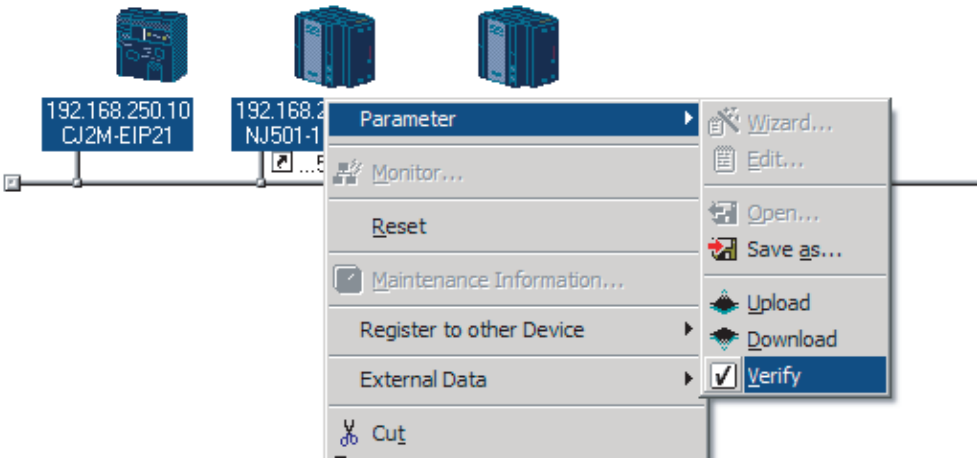
Click the **OK** Button or the **Close** Button.

Verifying the Device Parameters

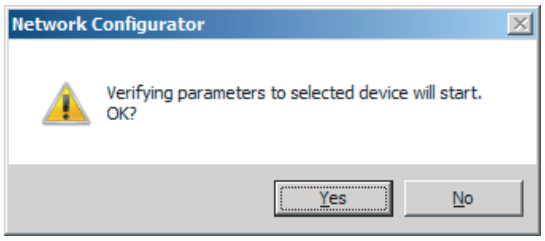
Use the following procedure to compare the device parameters for the devices selected in the Network Configuration Pane with those of the devices connected on the EtherNet/IP network.
The IP addresses, device types, and device parameters are compared.

1 Connect the Network Configurator to the network.

2 Click the icon of the built-in EtherNet/IP port to verify. To select multiple nodes, press and hold the Shift Key or the Ctrl Key while you select additional icons. (In the following example, 2 nodes are selected: 192.168.250.1 and 192.168.250.2.)
Right-click the icon to display the pop-up menu and select **Parameter** - **Verify**.



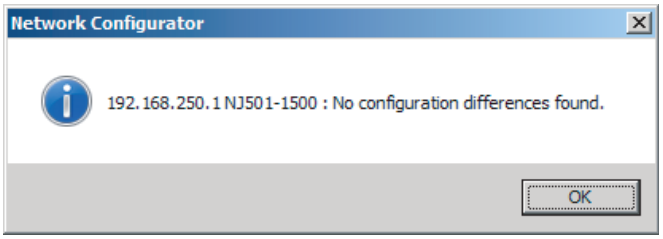
3 The following dialog box is displayed.



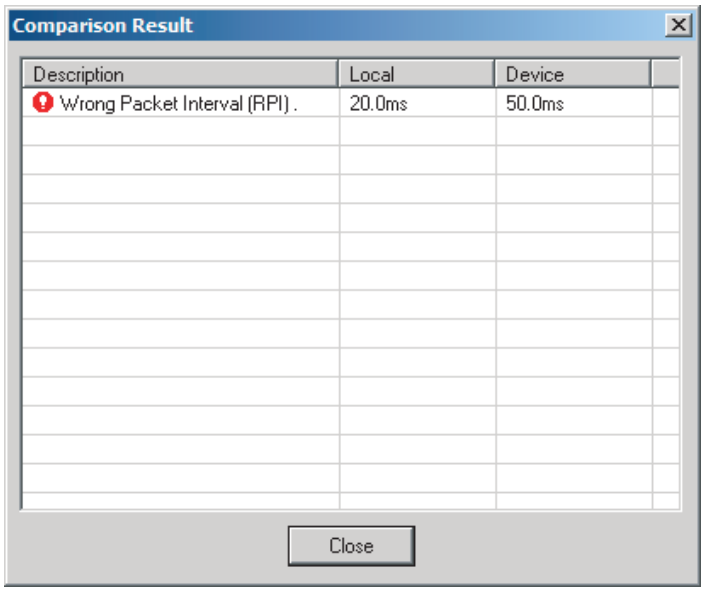
Click the **Yes** Button or the **No** Button.

4 The following dialog box is displayed.

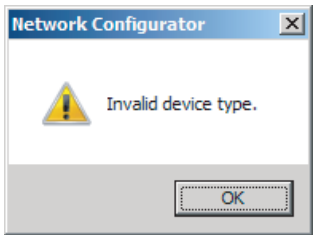
- Differences Not Found in the Comparison



- Differences Found in the Comparison

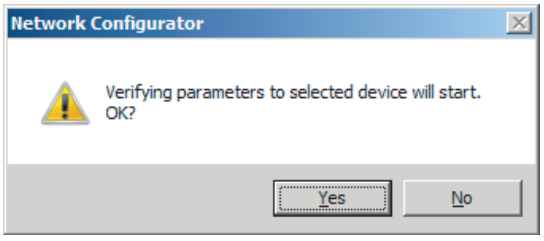


- Differences Found in the Device Type



Click the **OK** Button or the **Close** Button.

5 If multiple nodes have been selected and compared, the following message is displayed. Click the**Yes** Button.



The comparison results are displayed in order of the selected nodes.

6-2-12 Starting and Stopping Tag Data Links

This section describes the procedure for starting/stopping tag data links. For details on how to connect Network Configurator to a network, refer to *6-2-8 Connecting the Network Configurator to the Network* on page 6-52.

Automatically Starting Tag Data Links

Tag data links are automatically started immediately after the data link parameters are downloaded from the Network Configurator.

(They are automatically started after the CPU Unit's power is turned ON or the Unit is restarted.)

Additional Information

With a CPU Unit with unit version 1.04 or later that operates as the originator device, a *Tag Data Link Connection Timeout* error will occur if a connection is not established with the target device within one minute after the tag data links are started.
Even after this error occurs, reconnection processing is continued periodically until automatic recovery is performed.
If the application environment allows you to ignore this error, such as when a target device is started later than the originator device, you can change the event level to the observation level.

Starting and Stopping Tag Data Links for the Entire Network

You can start and stop tag data links for the entire network from the user program or from the Network Configurator.

Precautions for Correct Use

Use the same method (i.e., either the user program or the Network Configurator) to both start and stop tag data links.
For example, if you use the *_EIP_TDLinkStopCmd* (Tag Data Link Communications Stop Switch) system-defined variable stop tag data links, you cannot start them from the Network Configurator.

● Using Commands in the User Program

You can start and stop tag data links on a device basis by changing the values of the following system-defined variables from FALSE to TRUE in the user program. (Refer to *Section 3 System-defined Variables Related to the Built-in EtherNet/IP Port* on page 3-1.)

- NX701 CPU Unit and NX102 CPU Unit:
  You can individually start and stop tag data links for each built-in EtherNet/IP port.
  - Tag data links start/stop operation switch for built-in EtherNet/IP port 1
    _EIP1_TDLinkStartCmd (CIP Communications1 Tag Data Link Communications Start Switch)
    _EIP1_TDLinkStopCmd (CIP Communications1 Tag Data Link Communications Stop Switch)
  - Tag data links start/stop operation switch for built-in EtherNet/IP port 2
    _EIP2_TDLinkStartCmd (CIP Communications2 Tag Data Link Communications Start Switch)
    _EIP2_TDLinkStopCmd (CIP Communications2 Tag Data Link Communications Stop Switch)
- NX1P2 CPU Unit:
  _EIP1_TDLinkStartCmd (CIP Communications1 Tag Data Link Communications Start Switch)
  _EIP1_TDLinkStopCmd (CIP Communications1 Tag Data Link Communications Stop Switch)
- NJ-series CPU Unit:
  _EIP_TDLinkStartCmd (Tag Data Link Communications Start Switch)
  _EIP_TDLinkStopCmd (Tag Data Link Communications Stop Switch)

Additional Information

- Change the Tag Data Link Communications Start Switch to TRUE, while the Tag Data Link Communications Stop Switch is FALSE.
  If the Tag Data Link Communications Stop Switch is TRUE, the tag data links do not start even if the Tag Data Link Communications Start Switch is changed to TRUE.
  Furthermore, if the Tag Data Link Start Switch and the Tag Data Link Stop Switch are both TRUE, an error occurs, the Multiple Switches ON Error system-defined variable changes to TRUE, and the event is recorded in the event log.
- After you start the tag data links, do not force the Tag Data Link Communications Start Switch to change to FALSE from the user program or from the Sysmac Studio. It will change to FALSE automatically.

● Using the Network Configurator

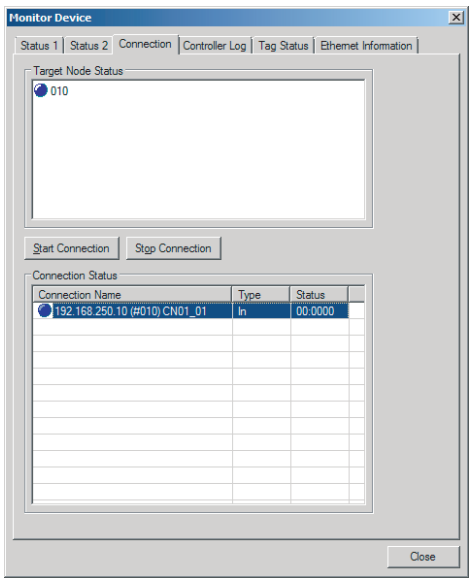
You can select **I/O Connection** - **Start** or **Stop** from the **Network** Menu to start and stop tag data links for individual devices.

Starting and Stopping Tag Data Links for Individual Devices

● Using the Network Configurator

You can start and stop tag data links on a device basis (at the originator) by selecting **Monitor** from the **Device** Menu and performing the following operation in the **Connection** Tab Page in the **Monitor Device** Dialog Box.

When using an NX701 CPU Unit or NX102 CPU Unit, you can start and stop tag data links for each of the built-in EtherNet/IP port 1 and 2 connected to the Network Configurator.



Start Connection Button:

Starts all connections for which the device is the originator.

Stop Connection Button:

Stops all connections for which the device is the originator.

6-2-13 Clearing the Device Parameters

You can clear the tag data link settings (or return them to their factory settings) that are saved in the registered EtherNet/IP device.

The following shows how to clear tag data link parameters. For details on how to connect to the network from the Network Configurator, refer to *6-2-8 Connecting the Network Configurator to the Network* on page 6-52.

Precautions for Correct Use

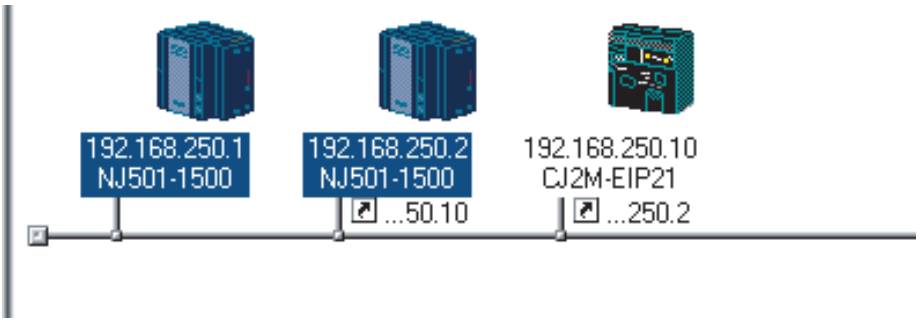
For a CPU Unit with Unit Version 1.10 or Later

- Use the Network Configurator version 3.58 or higher to perform the following procedure to clear the tag data link settings.
- If you perform the following procedure from the Network Configurator version 3.57 or lower, the tag data link settings are not cleared. Refer to Additional Information in this section for the procedure to clear the tag data link settings from the Network Configurator version 3.57 or lower.

1 Connect the Network Configurator to the network.

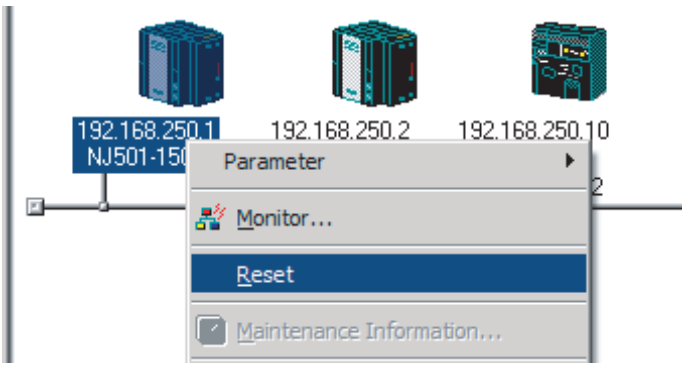
2 Select the icon of the device from which you want to clear the device parameters.

In the following example, two nodes are selected: 192.168.250.1 and 192.168.250.2. To select multiple nodes, press and hold the **Shift** Key while you select additional icons.

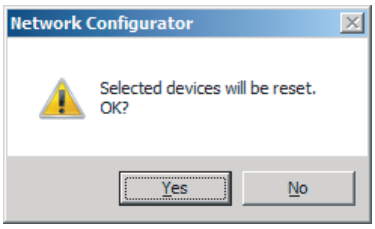


3 Select **Device** - **Reset**.

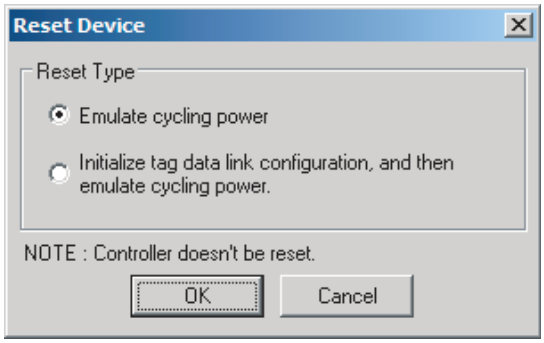
You can also right-click the icon and select **Reset** from the pop up menu.



4 The following dialog box is displayed.



- If you click the **Yes** Button:
  The following dialog box is displayed.



  Select the **Initialize tag data link configuration, and then emulate cycling power** Option, and then click the **OK** Button.

Precautions for Correct Use

The Controller is not restarted. Only the built-in EtherNet/IP port is reset.

- If you click the **No** Button:
  The tag data link settings will not be cleared and the built-in EtherNet/IP port will not be reset.

Additional Information

You can also execute the Reset service of the Identity Object for the CPU Unit to clear the tag data link settings. The procedure to execute the service from the Network Configurator is given below.

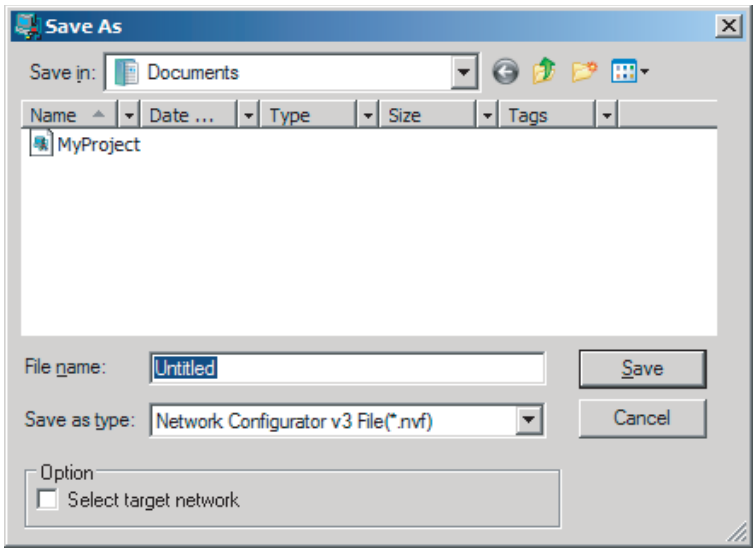
1. Connect the Network Configurator to the network.
2. Select **Tool** - **Setup Parameters** in the main window.
   Then the dialog box for the general parameter settings are displayed.
3. Specify the target device and message to send.
   - Target Node Address : Enter the IP address of the target device.
   - Service : Select **Reset**.
   - Class : Enter *01*.
   - Instance : Enter *01*.
   - Attribute : Enter *00*.
   - Data : Enter *02*[*1].
4. Click the **Send** Button.

*1. For a CPU Unit with unit version 1.09 or earlier, specify *01*.

6-2-14 Saving the Network Configuration File

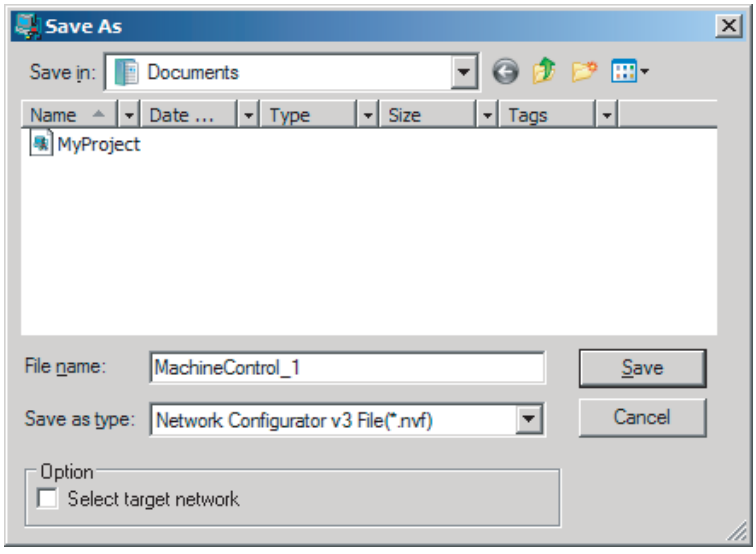
You can save device parameters set in the Network Configurator or device parameters uploaded from the network in a network configuration file.

1 Select **File** - **Save As**.
The following dialog box is displayed.



Untitled.nvf is displayed as the default file name.

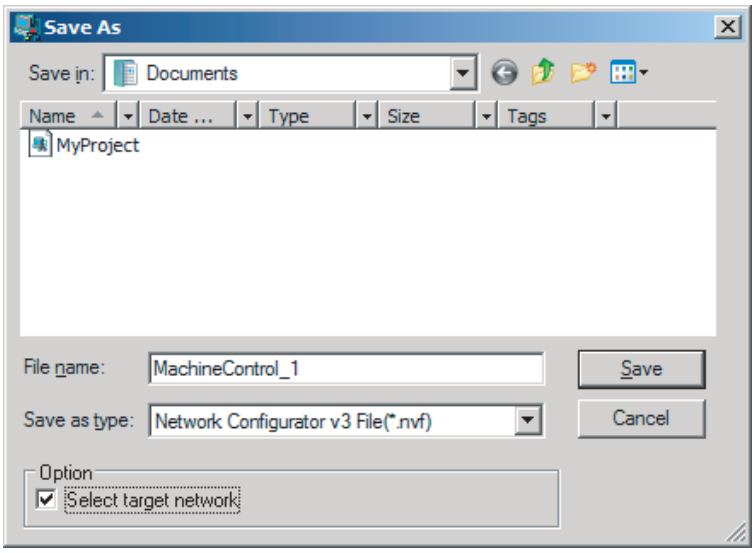
2 Input the file name, and then click the **Save** Button.



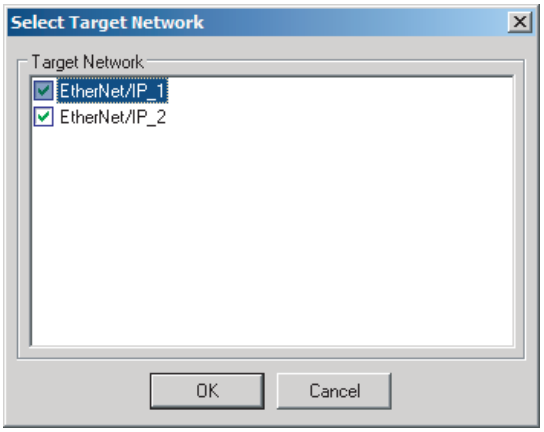
This completes the network configuration file save operation.

3 When the network configuration is changed later, you can overwrite the existing network configuration file if you select **File** - **Save**, or click the Button.

4 You can select the **Select target network** Check Box in the **Option** Area to select and save only the required network configuration files from the existing multiple files.



Select the check boxes of the networks to save and click the **OK** Button.

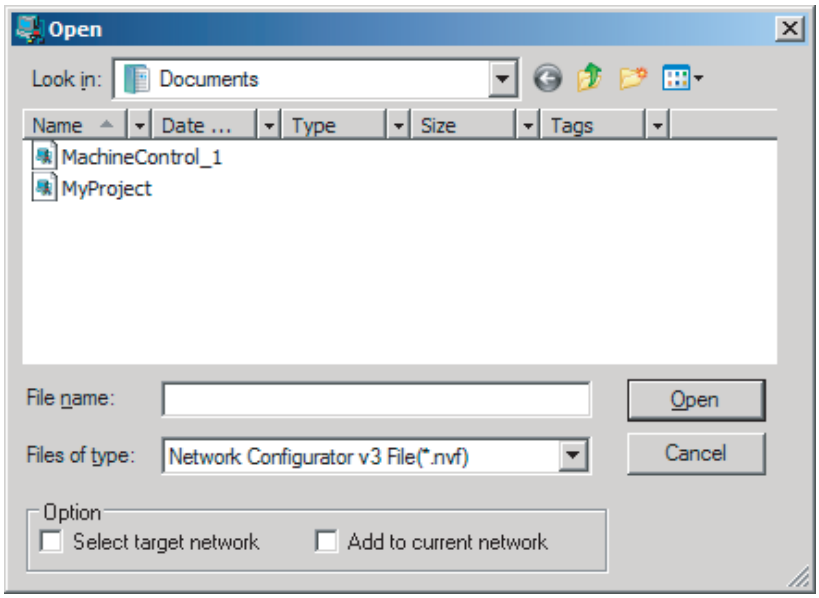


6-2-15 Reading a Network Configuration File

You can read out a previously saved network configuration file into the Network Configurator.

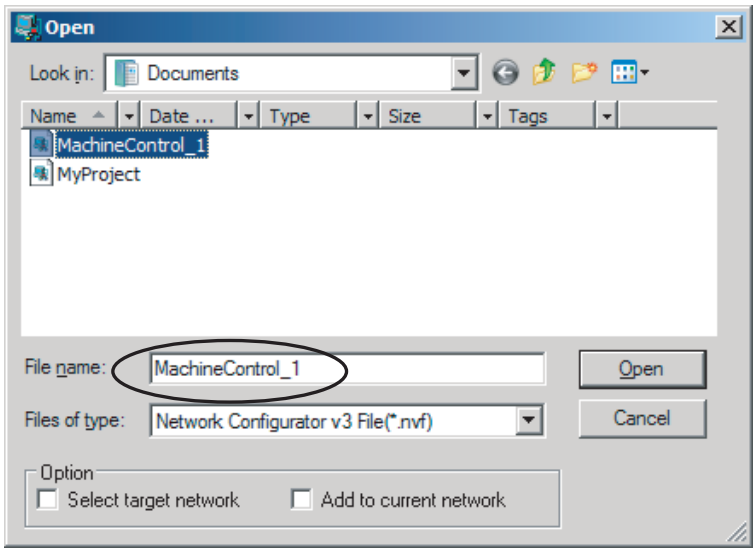
1 Select **File** - **Open**, or click the Button.

The following dialog box is displayed.



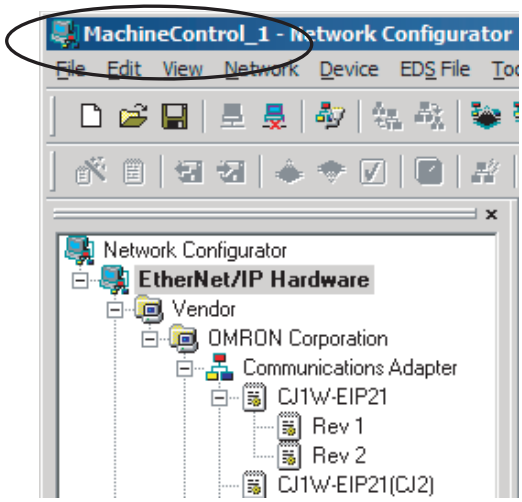
If the network configuration file that you want to read out is not displayed, change to another folder.

2 If you select the network configuration file that you want to read out, that file name is displayed in the File name Field.



3 Click the **Open** Button to read out the network configuration file.

4 The Network Configurator's Title Bar will display the name of the file that was read out.



5 Select options in the **Option** Area as necessary.
The options are listed below.

Setting	Description
Select target network	Allows you to select specific networks from the network configuration and open them.
Add to current document	Allows you to add the networks from the network configuration file that is currently open to the current configuration file.

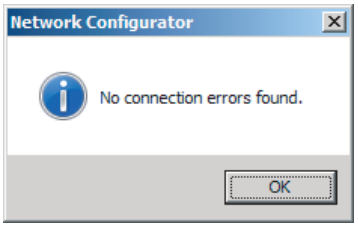
Additional Information

The save format will depend on the version of the Network Configurator. You can import configuration files (*.ncf) created with the Network Configurator for EtherNet/IP (version 2 or lower) if you select **External Data** - **Import** from the **File** Menu.

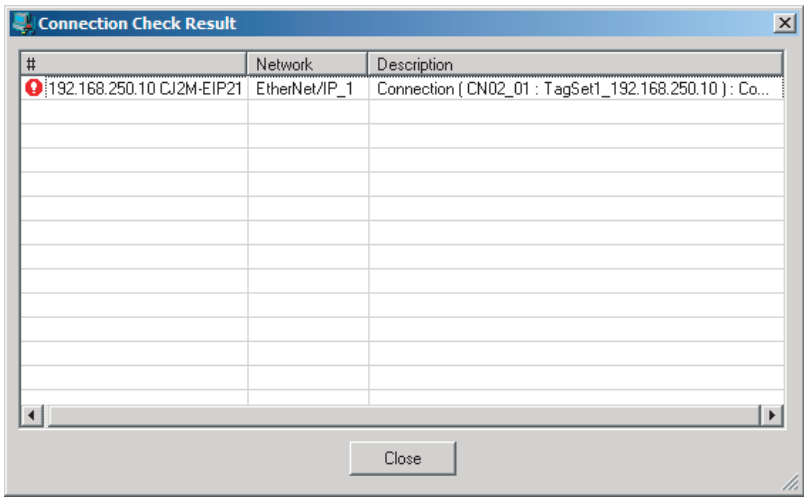
6-2-16 Checking Connections

You can check the consistency of connection parameters for network configuration files with device parameters that were set with the Network Configurator or device parameters uploaded from the network.

1 Select **Check Connection** from the **Network** Menu.
The following dialog box is displayed if parameters are normal.

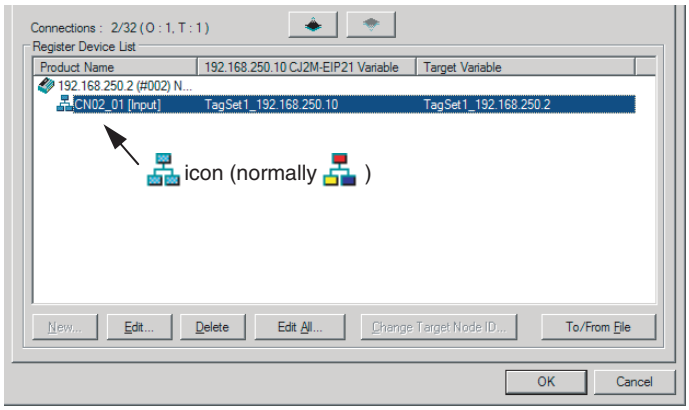


The following dialog box is displayed if there are parameter errors. Check the displayed details and review the settings.



If an inconsistency is found, open the originator's **Edit Device Parameter** Dialog Box and click the **Connection** Tab. The inconsistent connection in the **Register Device List** is displayed with a icon (instead of the normal icon).

To change the connection setting and select a different target variable, select the connection as shown below and click the **Edit** Button.



6-2-17 Changing Devices

You can change devices that are registered in a network configuration with the Network Configurator. Select **Change Device** from the **Device** Menu to display a list of the possible devices to change. You can change a device only when there is complete or upward compatibility with the device.

Device Changes

Model after change		CS1W-EIP21	CJ1W-EIP21	CJ1W-EIP21 (CJ2)	CJ2B-EIP21	CJ1W-EIP21 (NJ)	CJ2M	NJ501-□□□□ NJ301-□□□□ NJ101		NX701	NX102	NX1P2
Model before change	CIP Rev	Rev 2	Rev 3	Rev 3	Rev 3	Rev 3	Rev 3	Rev 1 *1	Rev 2 *2	Rev 2	Rev 2	Rev 2
CS1W-EIP21	Rev 2	----	Yes	Yes	Yes	△5	△3	△4/5	△4/5	No	No	No

Model after change		CS1W-EIP21	CJ1W-EIP21	CJ1W-EIP21 (CJ2)	CJ2B-EIP21	CJ1W-EIP21 (NJ)	CJ2M	NJ501-□□□□ NJ301-□□□□ NJ101		NX701	NX102	NX1P2
Model before change	CIP Rev	Rev 2	Rev 3	Rev 3	Rev 3	Rev 3	Rev 3	Rev 1 *1	Rev 2 *2	Rev 2	Rev 2	Rev 2
CJ1W-EIP21	Rev 3	Yes	----	Yes	Yes	△5	△3	△4/5	△4/5	No	No	No
CJ1W-EIP21 (CJ2)	Rev 3	△1	△1	----	Yes	△5	△3	△4/5	△4/5	△5	△4/5	△4/5
CJ2B-EIP21	Rev 3	△1	△1	Yes	----	△5	△3	△4/5	△4/5	△5	△4/5	△4/5
CJ1W-EIP21 (NJ)	Rev 3	△1/2	△1/2	△2	△2	----	△2/6	△4	△4	Yes	△4	△4
CJ2M	Rev 3	△1	△1	Yes	Yes	△5	----	△4/5	△4/5	△5	△4/5	△4/5
NJ501-□□□□ NJ301-□□□□ NJ101	Rev 1	△1/2	△1/2	△2	△2	Yes	△2/6	----	Yes	No	No	No
	Rev 2	△1/2	△1/2	△2	△2	Yes	△2/6	Yes	----	Yes	△4	△4
NX701	Rev 2	No	No	△2	△2	Yes	△2/6	No	△4	----	△4	△4
NX102	Rev 2	No	No	△2	△2	Yes	△2/6	No	Yes	Yes	----	△4
NX1P2	Rev 2	No	No	△2	△2	Yes	△2/6	No	Yes	Yes	Yes	----

*1. CPU Unit with a unit version 1.00 to 1.02

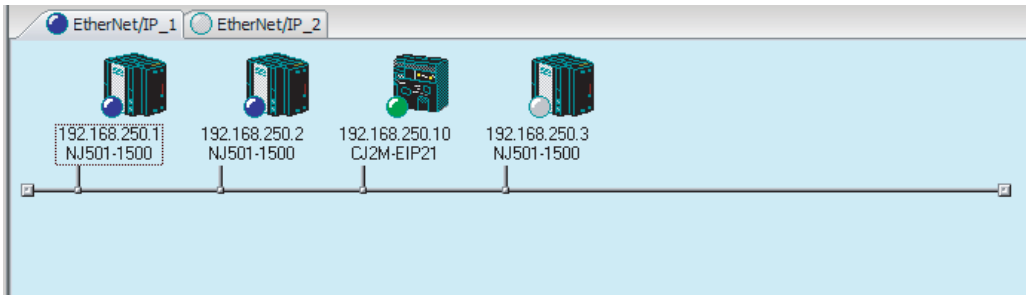
*2. CPU Unit with a unit version 1.03 or later

Yes	Can be changed.
No:	Cannot be changed.
△0	Cannot be changed if a Japanese variable is specified in the tag.
△1	Cannot be changed if a network variable is specified as a tag.
△2	Cannot be changed if the maximum size of a tag name or tag set name (size after conversion into UTF-8) exceeds 48 bytes.
△3	Cannot be changed if the following items exceed the permissible settings of the device after the change: • Number of I/O connections, number of tags, number of tag sets, and size of one tag set.
△4	Cannot be changed in any of the following cases: • The number of I/O connections, number of tags, number of tag sets, or size of one tag set exceeds the permissible settings for the device after the change. • RPI exceeds the permissible settings or is set in 0.5-ms increments (such as 10.5ms)
△5	Cannot be changed if a tag set size is an odd number of bytes.
△6	Cannot be changed if tags, tag sets, or refreshing sizes exceed the permissible settings.
△7	Cannot be changed if the maximum number of tags per tag set exceeds the permissible setting.

6-2-18 Displaying Device Status

Device status is displayed using the following icons in Maintenance Mode.

To enter Maintenance Mode, select **Large Icons - Maintenance Mode** from the **View** Menu.



Icon	Status
(white)	Offline
(gray)	Default (including no Controller Configurations and Setup)
(green)	Idle (including when the Controller is in PROGRAM mode)
(blue)	Normal communications state (including when the Controller is in RUN mode)
(yellow)	Warning status (including when there is a partial fault or non-fatal error in the Controller)
(red)	Alarm status (including when there is a major fault or fatal error in the Controller)

6-3 Ladder Programming for Tag Data Links

6-3-1 Ladder Programming for Tag Data Links

The following conditions 1 to 3 should be fulfilled if you use tag data link data for a ladder program. The additional conditions 4 and 5 should be also fulfilled if you input the Controller information of the target node.

● Conditions for enabling tag data links for the built-in EtherNet/IP port on a NJ/NX-series CPU Unit

The following conditions 1 and 2 should be both fulfilled.

No.	Condition
1	The following error status bits in the *_EIP_ErrSta* (Built-in EtherNet/IP Error) variable are FALSE. • Major fault: Bit 7 • Partial fault: Bit 6 • Minor fault: Bit 5
2	The *_EIP_EtnOnlineSta* (Online) variable*1 is TRUE.

● Condition for tag data links with connection established to the target device

The following condition 3 should be fulfilled.

No.	Condition
3	In the *_EIP_EstbTargetSta* (Normal Target Node Information) variable*2, the bit corresponding to the target node address is TRUE.

● Condition of the Controller operating mode (operating or stopped) (only for OMRON Controllers)

The following condition 4 should be fulfilled.

No.	Condition
4	In the *_EIP_TargetPLCModeSta* (Target PLC Operating Mode) variable*3, the bit corresponding to the target node address is TRUE.

● Condition of the Controller error status (fatal or non-fatal error) of the target node (only for OMRON Controllers)

The following condition 5 should be fulfilled.

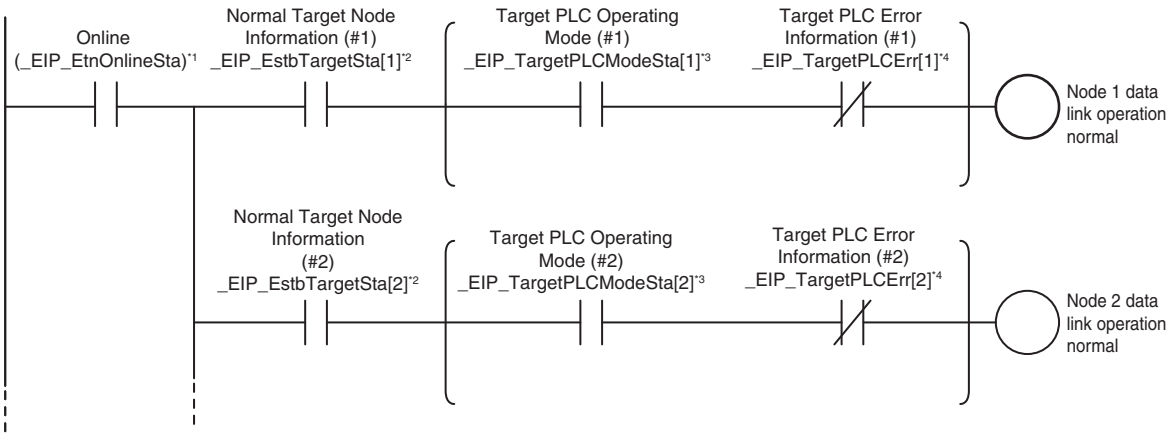
No.	Condition
5	In the *_EIP_TargetPLCErr* (Target PLC Error Information) variable*4, the bit corresponding to the target node address is FALSE. When you want to use the Target Node Controller Error Flag, the Controller status must be included in the tag sets for both the originator and target. Include the Controller status by using the Network Configurator to select the Include Option in the Edit Tag Set Dialog Box.

*1. This is a system-defined variable for NJ-series CPU Units.
For NX701 CPU Units and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_EtnOnlineSta
Built-in EtherNet/IP port 2: _EIP2_EtnOnlineSta
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_EtnOnlineSta
*2. This is a system-defined variable for NJ-series CPU Units.
For NX701 CPU Units and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_EstbTargetSta
Built-in EtherNet/IP port 2: _EIP2_EstbTargetSta
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_EstbTargetSta
*3. This is a system-defined variable for NJ-series CPU Units.
For NX701 and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_TargetPLCModeSta
Built-in EtherNet/IP port 2: _EIP2_TargetPLCModeSta
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_TargetPLCModeSta
*4. This is a system-defined variable for NJ-series CPU Units.
For NX701 and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_TargetPLCErr
Built-in EtherNet/IP port 2: _EIP2_TargetPLCErr
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_TargetPLCErr

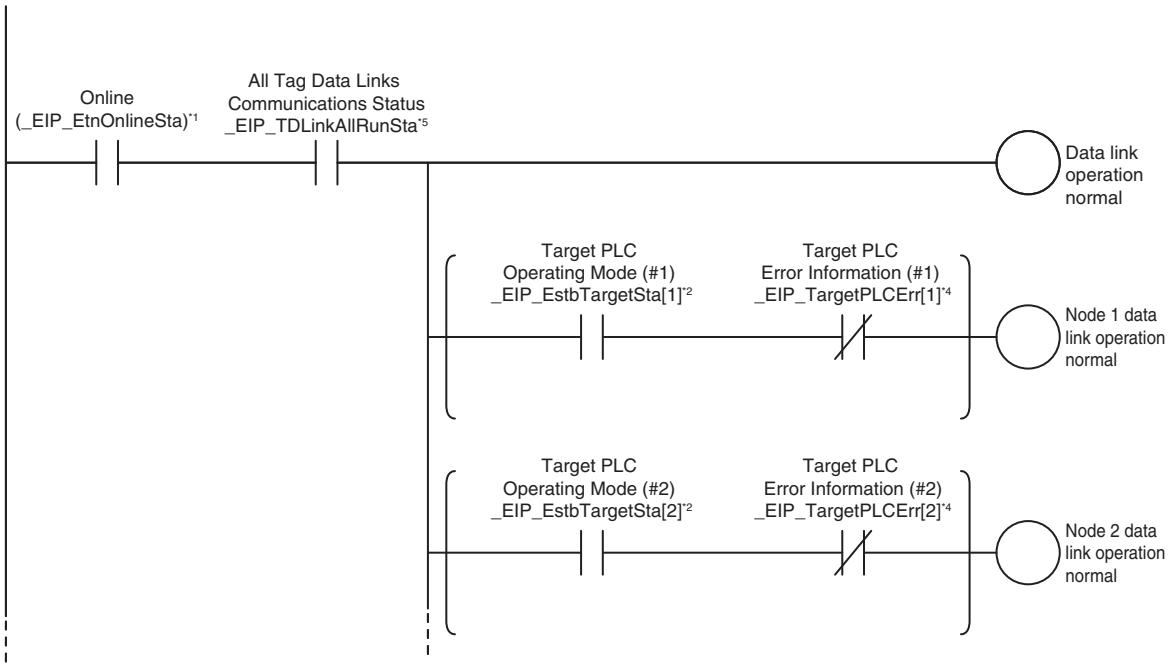
● Programming Example for Normal Operation Detection

The following program can be used to confirm that normal communications are being performed for each target node. If the Controller status is included in the tag data, the status of the Controller can also be detected.

- Normal Operation Detection Programming Example 1



- Normal Operation Detection Programming Example 2



*1. This is a system-defined variable for NJ-series CPU Units.
For NX701 and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_EtnOnlineSta
Built-in EtherNet/IP port 2: _EIP2_EtnOnlineSta
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_EtnOnlineSta

*2. This is a system-defined variable for NJ-series CPU Units.
For NX701 and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_EstbTargetSta
Built-in EtherNet/IP port 2: _EIP2_EstbTargetSta
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_EstbTargetSta

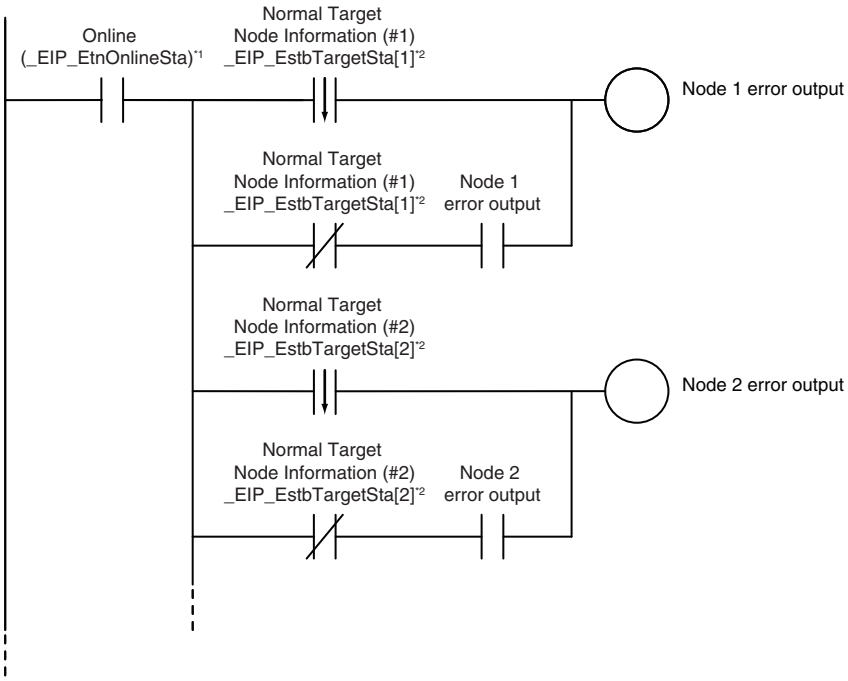
*3. This is a system-defined variable for NJ-series CPU Units.
For NX701 and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_TargetPLCModeSta
Built-in EtherNet/IP port 2: _EIP2_TargetPLCModeSta
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_TargetPLCModeSta

*4. This is a system-defined variable for NJ-series CPU Units.
For NX701 and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_TargetPLCErr
Built-in EtherNet/IP port 2: _EIP2_TargetPLCErr
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_TargetPLCErr

*5. This is a system-defined variable for NJ-series CPU Units.
For NX701 and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_TDLinkAllRunSta
Built-in EtherNet/IP port 2: _EIP2_TDLinkAllRunSta
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_TDLinkAllRunSta

● Programming Example for Error Detection

The following program can be used to check for tag data link errors for each target node. This programming is used to detect errors which may occur after the data links for all the nodes are started normally.

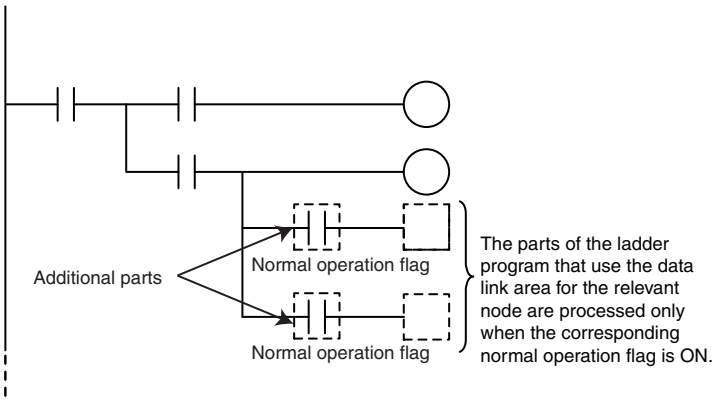


*1. This is a system-defined variable for NJ-series CPU Units.
For NX701 and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_EtnOnlineSta
Built-in EtherNet/IP port 2: _EIP2_EtnOnlineSta
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_EtnOnlineSta

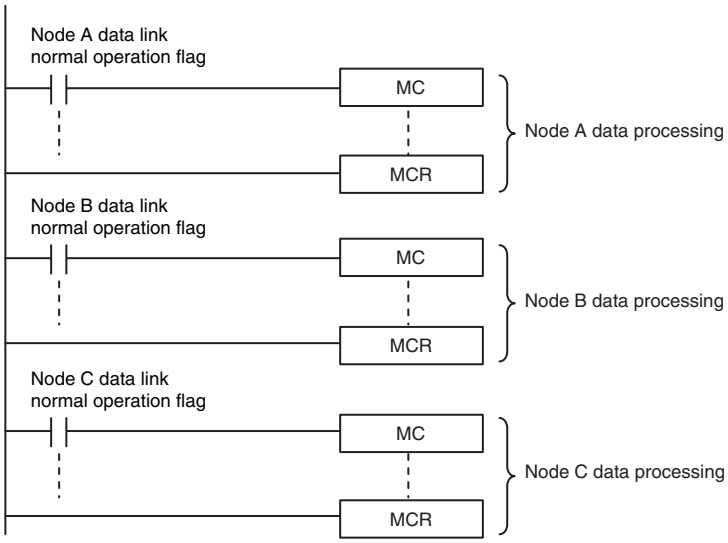
*2. This is a system-defined variable for NJ-series CPU Units.
For NX701 and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_EstbTargetSta
Built-in EtherNet/IP port 2: _EIP2_EstbTargetSta
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_EstbTargetSta

● Data Processing Programming Example

- The following shows an example where data processing is performed only when data links are operating normally.



- The following shows an example where data processing is performed only when data links are operating normally with MC and MCR instructions, or with JMP instructions.



Precautions for Correct Use

Even if an error occurs in communications with a target device, the input data from the target device will remain stored in words allocated in memory to the local node. To prevent malfunctions, write the user program so that no input processing is performed when any of the following bits of the *_EIP_ErrSta* (Built-in EtherNet/IP Error) variable is TRUE.

- Major fault: Bit 7
- Partial fault: Bit 6
- Minor fault: Bit 5

6-3-2 Status Flags Related to Tag Data Links

The status of the tag data links is reflected in the following system-defined variables.

Variable	Description
_EIP_TargetPLCModeSta[255]*1 (Target PLC Operating Mode) (Corresponds to the Controller Operating Flag in the Controller status.)	This variable shows the operating status of the target node Controller that is connected with the built-in EtherNet/IP port as the originator. The information in this area is valid only when the corresponding Normal Target Node Information is TRUE. If the value is FALSE, the Target Node Controller Operating Information indicates the previous operating status. Array[x] is TRUE: The target Controller with a node address of x is in operating status. Array[x] is FALSE: Other than the above.
_EIP_TargetNodeErr[255]*2 (Target Node Error Information) (Corresponds to the Controller Error Flag in the Controller status.)	This variable indicates that the connection for Registered Target Node Information is not established or that an error has occurred in the target the Controller. The array elements are valid only when the Registered Target Node Information is TRUE. Array[x] is TRUE: The Registered Target Node Information for a node address of x is TRUE, and the Normal Target Node Information is FALSE or the Target PLC Error Information is TRUE. Array[x] is FALSE: When the Registered Target Node Information for a node address of x is FALSE, or when the Registered Target Node Information is TRUE, the Normal Target Node Information is TRUE, and the Target PLC Error Information is FALSE.
_EIP_EstbTargetSta[255]*3 (Normal Target Node Information) (This status is not included in the Controller status.)	This variable gives a list of nodes that have normally established built-in EtherNet/IP connections. Array[x] is TRUE: The connection to the node with a node address of x is established normally. Array[x] is FALSE: A connection is not established yet, or an error has occurred.

*1. This is a system-defined variable for NJ-series CPU Units.
For NX701 and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_TargetPLCModeSta
Built-in EtherNet/IP port 2: _EIP2_TargetPLCModeSta
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_TargetPLCModeSta

*2. This is a system-defined variable for NJ-series CPU Units.
For NX701 and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_TargetNodeErr
Built-in EtherNet/IP port 2: _EIP2_TargetNodeErr
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_TargetNodeErr

*3. This is a system-defined variable for NJ-series CPU Units.
For NX701 and NX102 CPU Units, the variable varies depending on the built-in EtherNet/IP port, as shown below.
Built-in EtherNet/IP port 1: _EIP1_EstbTargetSta
Built-in EtherNet/IP port 2: _EIP2_EstbTargetSta
For NX1P2 CPU Units, the variable is as below.
Built-in EtherNet/IP port 1: _EIP1_EstbTargetSta

6-4 Tag Data Links with Other Models

The performance of tag data links depends on the CPU Unit model and EtherNet/IP Unit model as shown below.
When you use tag data links between the built-in EtherNet/IP port on an NJ/NX-series CPU Unit and another CPU Unit or EtherNet/IP Unit, configure the tag data link settings based on the Unit which has the lower level of communications performance.

● Differences in Tag Data Link Performance Specifications

Item		NX-series CPU Unit			NJ-series CPU Unit		CJ2M-CPU3□		CS1W-EIP21 CJ1W-EIP21 CJ2H-CPU6□-EIP
							Unit version		
		NX701	NX102	NX1P2	Ver. 1.00 to 1.02	1.03 or higher	2.0	2.1 or later	
Tag	Total size of all tags	184,832 words (total of 369,664 words with two ports)	9,600 words (total of 19,200 words with two ports)	9,600 words			640 words		184,832 words
	Maximum size of tag	722 words (721 words when the tag set includes the Controller status)	300 words (299 words when the tag set includes the Controller status)				20 words (19 words when the tag set includes the Controller status)	640 words (639 words when the tag set includes the Controller status)	722 words (721 words when the tag set includes the Controller status)
	Number of registrable tags	256 (total of 512 with two ports)		256*1			32		256
Tag set	Maximum size of 1 tag set	722 words (721 words when the tag set includes the Controller status)	300 words (299 words when the tag set includes the Controller status)				20 words (19 words when the tag set includes the Controller status)	640 words (639 words when the tag set includes the Controller status)	722 words (721 words when the tag set includes the Controller status)
	Number of tags per tag set	8 (7 tags when the tag set includes the Controller status) Note: Input and output variables cannot be combined in one tag set.							
	Number of registrable tag sets	256 (total of 512 with two ports)	32 (total of 40 with two ports)*2	32			32		256

Item		NX-series CPU Unit			NJ-series CPU Unit		CJ2M-CPU3□		CS1W-EIP21 CJ1W-EIP21 CJ2H-CPU6□-EIP
							Unit version		
		NX701	NX102	NX1P2	Ver. 1.00 to 1.02	1.03 or higher	2.0	2.1 or later	
Connection	Number of connections	256 (total of 512 with two ports)	32 (total of 64 with two ports)	32			32		256
	Maximum data size per connection	722 words *3 (Data concurrency is maintained at each connection.)	300 words (Refer to *6-1-7 Concurrency of Tag Data Link Data* on page 6-12 for the conditions for maintaining data concurrency on a connection basis.)				20 words (Data concurrency is maintained at each connection.)	640 words	252 or 722 words*3 (Data concurrency is maintained at each connection.)
Packet intervals (RPIs)		0.5 to 10,000 ms in 0.5-ms increments	1 to 10,000 ms in 1-ms increments	2 to 10,000 ms in 1-ms increments	10 to 10,000 ms in 1-ms increments	1 to 10,000 ms in 1-ms increments	1 to 10,000 ms in 0.5-ms increments		0.5 to 10,000 ms in 0.5-ms increments
Communications bandwidth used (pps)*4		40,000 pps*5	12,000 pps*5	3,000 pps	1,000 pps	3,000 pps	3,000 pps		6,000 pps

*1. The maximum number of tags is given for the following conditions.
- All tag sets contain eight tags.
- The maximum number of tag sets (32) is registered.

*2. When tag sets that exceed total of 40 are set, a Number of Tag Sets for Tag Data Links Exceeded (840E0000 hex) event occurs.

*3. To use data of 505 bytes or more, large forward open (an optional CIP specification) should be supported. The SYSMAC CS/CJ-series Units support large forward open, and if you use nodes from other companies, confirm that the devices also support it.

*4. Here, pps means “packets per second” and indicates the number of packets that can be processed in one second.

*5. If the two built-in EtherNet/IP ports are used simultaneously, the maximum communications data size means the maximum data size of the total of the two ports.

● Specifying Tags

When you assign a tag to a device, you can specify the device with its network variable or I/O memory address. Some CPU Units, however, may not support both of these methods. Communications with such CPU Units are possible though, regardless of whether the I/O memory address or network variable is specified for the tag assignment.

The supported tag specification methods for each CPU Unit are listed in the table below.

Yes: Supported, No: Not supported

CPU Unit	EtherNet/IP Unit	Network Configurator hardware list name	Specifying with network variable	Specifying with I/O memory address
NX-series CPU Unit	---	NX701	Yes	No
		NX102	Yes	Yes *1*2
		NX1P2□□□□	Yes	Yes*1*2
NJ-series CPU Unit	---	NJ501-□□□□ NJ301-□□□□ NJ101	Yes	Yes*1
	CJ1W-EIP21	CJ1W-EIP21 (NJ)	Yes	Yes*1
CJ2H-CPU6□-EIP	---	CJ2B-EIP21	Yes	Yes
	CJ1W-EIP21	CJ1W-EIP21 (CJ2)	Yes	Yes
CJ2H-CPU6□	CJ1W-EIP21	CJ1W-EIP21 (CJ2)	No	Yes

CPU Unit	EtherNet/IP Unit	Network Configurator hardware list name	Specifying with network variable	Specifying with I/O memory address
CJ2M-CPU3□	---	CJ2M-EIP21	Yes	Yes
	CJ1W-EIP21	CJ1W-EIP21 (CJ2)	Yes	Yes
CJ2M-CPU1□	CJ1W-EIP21	CJ1W-EIP21 (CJ2)	No	Yes
CJ1 CPU Unit	CJ1W-EIP21	CJ1W-EIP21	No	Yes
CS1 CPU Unit	CS1W-EIP21	CS1W-EIP21	No	Yes

*1. To specify an I/O memory address for tag assignment, do not specify the address directly. Instead, create a variable with an AT specification of the I/O memory address on the Sysmac Studio, and then specify the variable for the tag.

*2. For NX102 and NX1P2 CPU Units, you need to set CJ memory when you specify an I/O memory address for tag assignment. For details on CJ memory setting, refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)*.

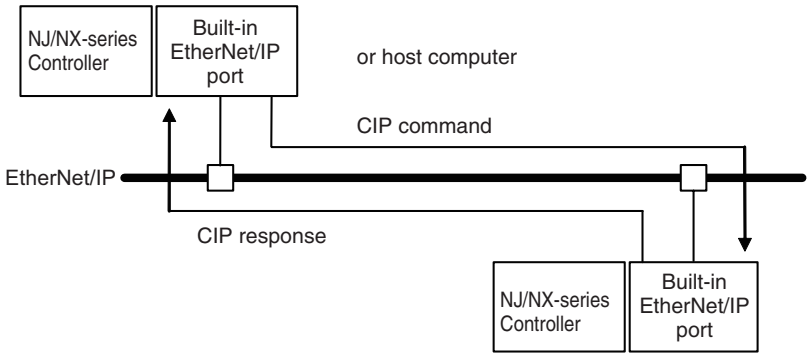
7

CIP Message Communications

7-1 Overview of the CIP Message Communications Service

7-1-1 Overview of the CIP Message Communications Service

CIP commands can be sent to devices on the EtherNet/IP network whenever they are required.
You execute CIP_SEND instructions in a program in the NJ/NX-series CPU Unit to send CIP commands, such as those to read and write data and to receive the responses.
You can use CIP messages from the client to read and write memory in the Controller with the server without adding any special programming to the user program of the Controller with the server.



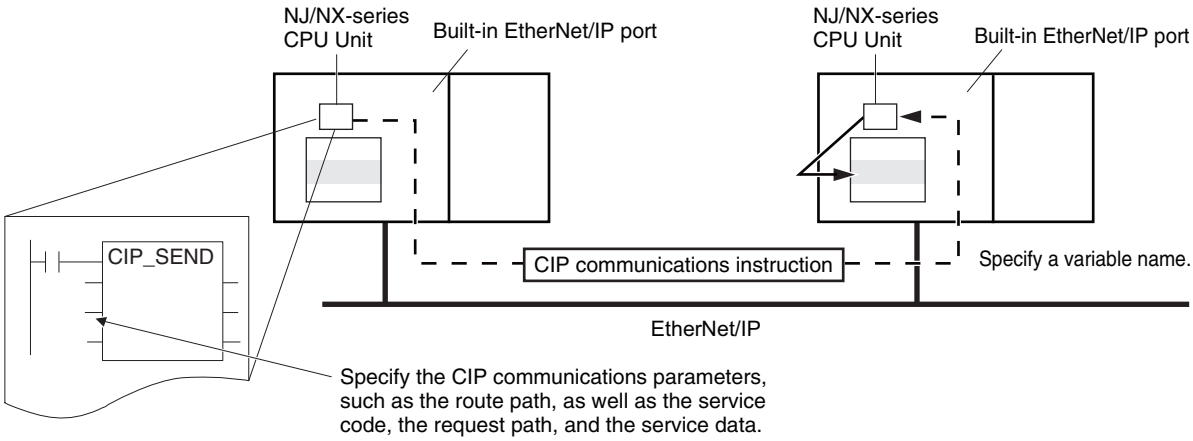
7-1-2 Message Communications Service Specifications

Item		Specification
Message type		Either of the following can be selected. CIP UCMM connectionless messages CIP class 3 connection messages
Execution method		CIPSend (Send Explicit Message Class 3) instruction or CIPUCMMSend (Send Explicit Message UCMM) instruction
Data contents		Sending required CIP commands and receiving responses
Communications parameters		Message type, timeout value, and route path specification
Maximum length per connection	Non-connection type (UCMM)	502 bytes
	Connection type (class 3)	• Using Forward_Open 502 bytes • Using Large_Forward_Open NX701 CPU Unit: 8192 bytes NX102 CPU Unit: 1994 bytes NX1P2 CPU Unit: 1994 bytes NJ-series CPU Unit: 1994 bytes

7-2 CIP Message Communications Client Function

7-2-1 Overview

The NJ/NX-series CPU Units can send a CIP message to an external device to request a service by specifying an internal object of the device which supports CIP message communications server functionality.
This is called the CIP message communications client function.
The NJ/NX-series CPU Units execute CIP communications instructions in the user program and send CIP messages. With those CIP messages, you can read and write variables of an NJ/NX-series CPU Unit on the EtherNet/IP network.



7-2-2 CIP Communications Instructions

The following CIP communications instructions are available.
For details on CIP communications instructions, refer to the *NJ/NX-series Instructions Reference Manual (Cat. No. W502)*.

Instructions	Name	Description	Communications method
CIPUCMM-Read	Read Variable UCMM Explicit	Reads the value of a variable with a Network Publish attribute from the specified remote Controller on the CIP network and stores the value in a variable at the local Controller.	CIP UCMM connectionless message
CIPUCMM-Write	Write Variable UCMM Explicit	Writes the value of a variable at the local controller to a variable with a Network Publish attribute at the specified remote Controller on the CIP network.	
CIPUCMM-Send	Send Explicit Message UCMM	Sends a specified CIP command to the specified remote Controller on the CIP network. Refer to *7-2-10 Response Codes* on page 7-35 and *7-5 CIP Object Services* on page 7-48 for information on the service codes and response codes that are used with the NJ/NX-series CPU Units.	

Instructions	Name	Description	Communications method
CIPOpen	Open CIP Class 3 Connection (Large_Forward_Open)	Opens a CIP class 3 connection (Large_Forward_Open) with the specified remote node.	CIP class 3 connection message
CIPOpenWithDataSize	Open CIP Class 3 Connection with Specified Data Size	Opens a CIP class 3 connection with the specified remote node that allows class 3 explicit messages of the specified data length or shorter to be sent and received.	
CIPRead	Read Variable Class 3 Explicit	Reads the value of a variable with a Network Publish attribute from the specified remote Controller on the CIP network and stores the value in a variable at the local Controller.	
CIPWrite	Write Variable Class 3 Explicit	Writes the value of a variable at the local controller to a variable with a Network Publish attribute at the specified remote Controller on the CIP network.	
CIPSend	Send Explicit Message Class 3	Sends a specified class 3 CIP command to the specified remote Controller on the CIP network. Refer to *7-2-10 Response Codes* on page 7-35 and *7-5 CIP Object Services* on page 7-48 for information on the service codes and response codes that are used with the NJ/NX-series CPU Units.	
CIPClose	Close CIP Class 3 Connection	Closes the CIP class 3 connection that is specified by the handle.	

Version Information

A CPU Unit with unit version 1.06 or later and Sysmac Studio version 1.07 or higher are required to use the CIPOpenWithDataSize instruction.

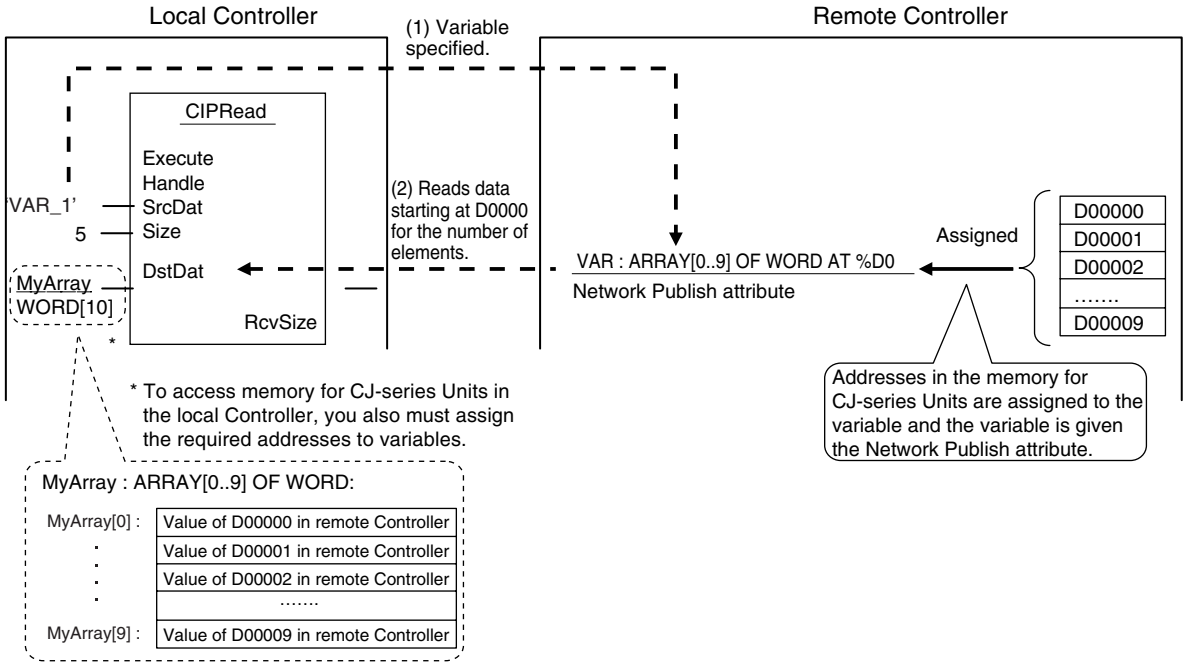
7-2-3 Using CIP Communications Instructions

CIP message communications include the following processes.
If CIP class 3 connections are used, the open and close processes are required before and after the data is sent and received.

Process	Description	Instruction
Open process (only for CIP class 3 connections)*1	Execute this process before you use a CIP message. Open processing is continued until a CIP class 3 connection is established.	CIPOpen CIPOpenWithDataSize
Sending and receiving variable data*2	This process is used to read and write data for specified variables with the Network Publish attributes.	CIPUCMMRead CIPUCMMWrite CIPRead CIPWrite
Sending CIP commands	You can set the required CIP command.	CIPUCMMSend CIPSend
Close process (only for CIP class 3 connections)	This process closes the connection.	CIPClose

*1. The maximum number of connection handles that you can obtain simultaneously through the opening process is 32. Even if a connection is disconnected for a timeout, the handle is not released. Execute the CIPClose instruction to close the connection.

*2. Addresses in memory for CJ-series Units (e.g., D0000) cannot be specified directly. To access memory for CJ-series Units, access a variable with an AT specification. (Accessing is possible only for NJ-series CPU Units.)



Precautions for Correct Use

You can execute up to 32 CIP communications instructions at the same time regardless of the instruction types.
Use exclusive control in the user program so that the number of CIP communications instructions executed at the same time does not exceed the above number.

7-2-4 Route Path

The route path indicates the path from the local CPU Unit to the remote Controller on the network. Routing for CIP communications instructions is performed based on the route path.

Route Path Notation

The EPATH data type is used to give route paths. The basic format is shown below.

Network_type_number\Destination_address

● NX701 CPU Unit and NX102 CPU Unit

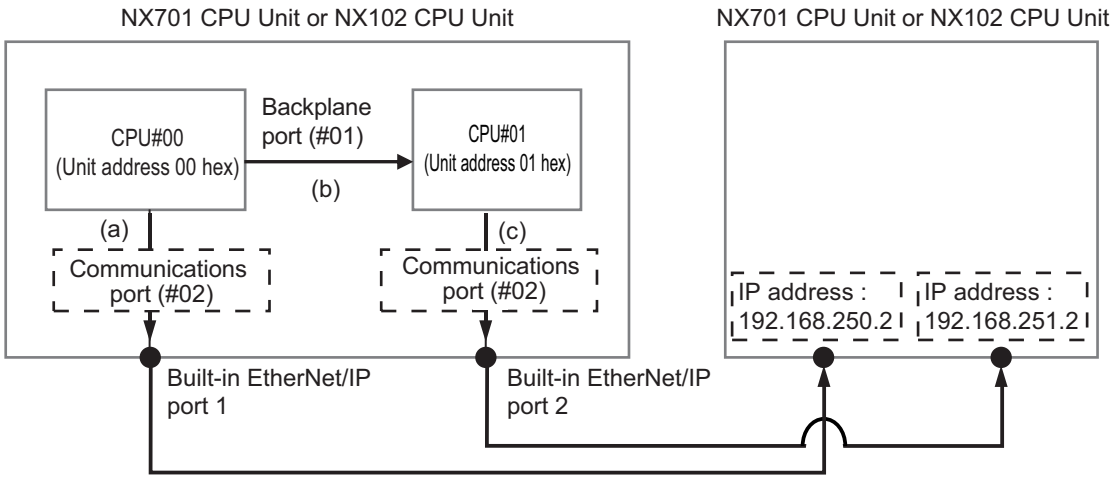
Two internal CPU Units are provided (each with a unique unit address) to control the two built-in EtherNet/IP ports.

- For the built-in EtherNet/IP port 1: CPU Unit with a unit address of 00 hex (CPU #00)
- For the built-in EtherNet/IP port 2: CPU Unit with a unit address of 01 hex (CPU #01)

The *RoutePath* input variable for the CIP communications instructions is used to distinguish the two CPU Units (CPU #00 and CPU #01) and send the CIP communications instructions.

Route path for sending a CIP communications instruction

- The CIP communications instruction is issued from CPU #00. (a)
- The output from the built-in EtherNet/IP port 2 is routed from CPU #00 via CPU #01. (b) to (c)



Route	Route notation	Route path specifications	
		Network type number (hexadecimal)	Destination address (hexadecimal)
Output from the built-in EtherNet/IP port 1	(a)	#02 (communications port)	IP address
Output from the built-in EtherNet/IP port 2	(b)	#01 (backplane port)	#01 (unit address of the CPU Unit) (CPU #01 for built-in EtherNet/IP port 2 communications)
	(c)	#02 (communications port)	IP address

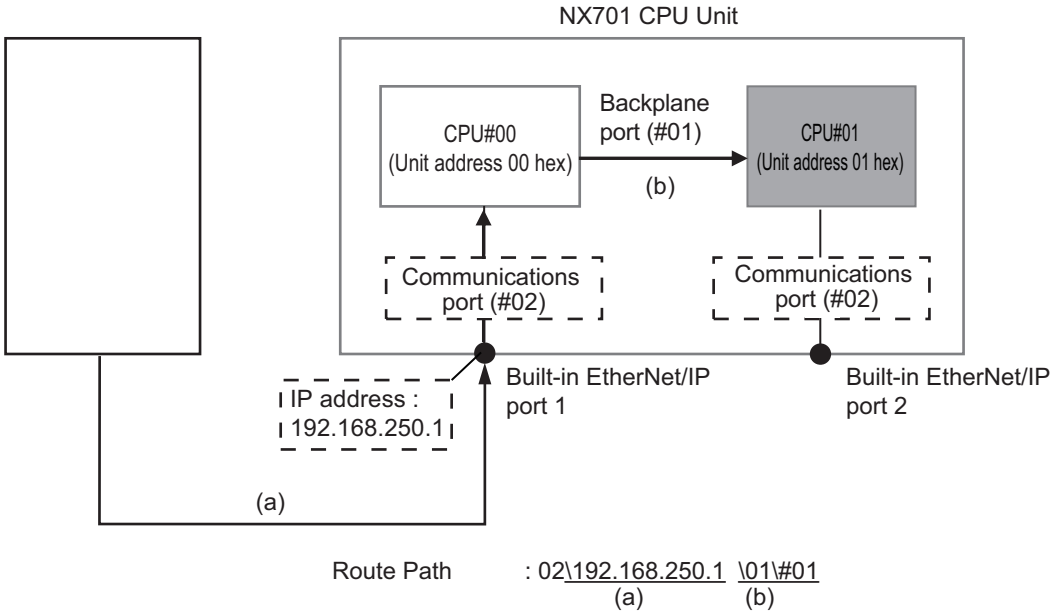
Route Path

Output from built-in EtherNet/IP port 1 : 02\192.168.250.2
(a)

Output from built-in EtherNet/IP port 2 : 01\#01 \02\192.168.251.2
(b) (c)

- The CPU Units (CPU#00 and CPU#01), which control the respective built-in EtherNet/IP ports, can be accessed via the backplane port regardless of whether the input is routed via the EtherNet/IP port 1 or 2.

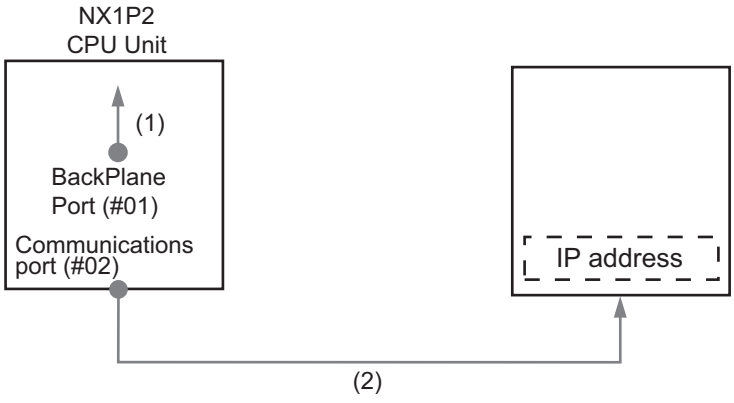
Example: Inputting an Ethernet Link object (class ID: F6 hex) to the built-in EtherNet/IP port 1 of the remote NX701 CPU Unit, and reading out the settings and status of the built-in EtherNet/IP port 2.



● NX1P2 CPU Unit

As shown in the table below, the network type number and the destination address are determined depending on whether the output is routed (1) to a Unit on the CPU Rack or (2) from a communications port on a Communications Unit.

Route	Network type number (hexadecimal)	Destination address (hexadecimal)
(1) Output to a Unit on the CPU Rack	#01 (backplane port)	Unit address of the destination Unit (Refer to Additional Information below.)
(2) Output from a communications port on a Communications Unit	#02 (built-in EtherNet/IP port)	IP address



1. When Routing the Output to a Unit on the CPU Rack
   Route the output to the backplane port for the network with the CPU Rack, with the Unit address of the destination Unit specified as the destination address.
2. When Routing the Output from a Communications Port on a Communications Unit
   Route the output to an EtherNet/IP port, with the IP address specified as the destination node address.

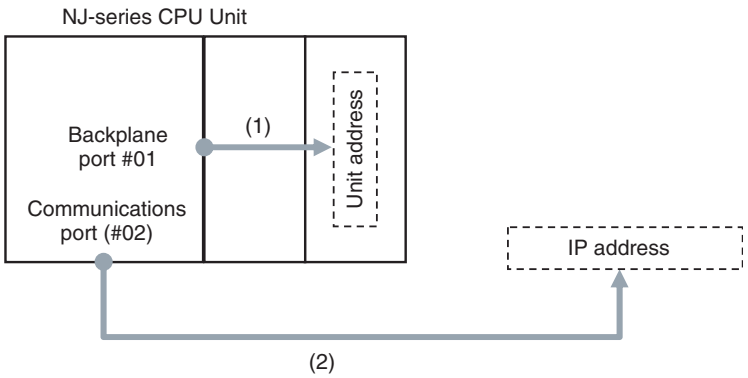
Additional Information

Unit Addresses
Unit addresses are used to identify each of devices connected to a single node on a network. Unit addresses are set as shown below.
- CPU Unit: 00 hex

● NJ-series CPU Unit

As shown in the table below, the network type number and the destination address are determined depending on whether the output is routed (1) to a Unit on the CPU Rack or (2) from a communications port on a Communications Unit.

Route	Network type number (hexadecimal)	Destination address (hexadecimal)
(1) Output to a Unit on the CPU Rack	#01 (backplane port)	Unit address of the destination Unit (Refer to Additional Information below.)
(2) Output from a communications port on a Communications Unit	#02 (built-in EtherNet/IP port)	IP address



1. When Routing the Output to a Unit on the CPU Rack
   Route the output to the backplane port for the network with the CPU Rack, with the Unit address of the destination Unit specified as the destination address.
2. When Routing the Output from a Communications Port on a Communications Unit
   Route the output to an EtherNet/IP port, with the IP address specified as the destination node address.

Additional Information

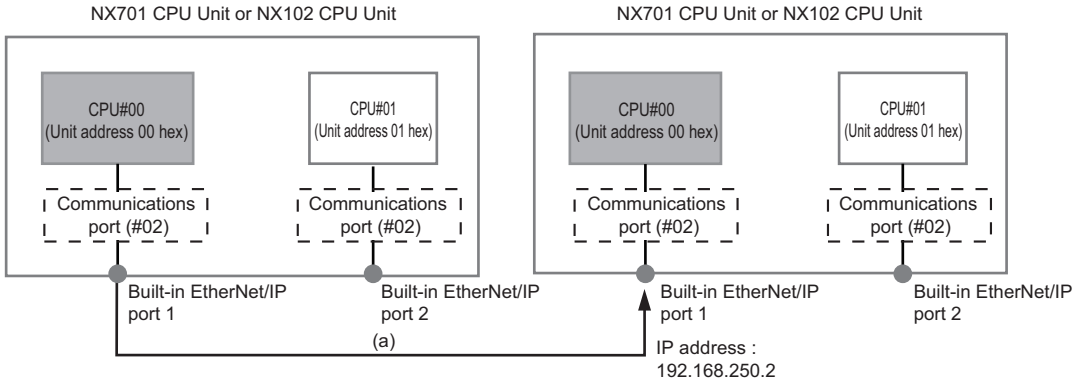
Unit Addresses
Unit addresses are used to identify each of devices connected to a single node on a network. Unit addresses are set as shown below.
- CPU Unit: 00 hex, 01 hex
- CPU Bus Units (EtherNet/IP Units): Unit number + 10 hex

Route Path Notation Examples

● NX701 CPU Unit and NX102 CPU Unit

The route path notation is different for communications using the built-in EtherNet/IP port 1 (CPU#00) and for communications using the built-in EtherNet/IP port 2 (CPU#01).
This section provides examples of route paths.
This example explains communications via an NX-series CPU Unit.

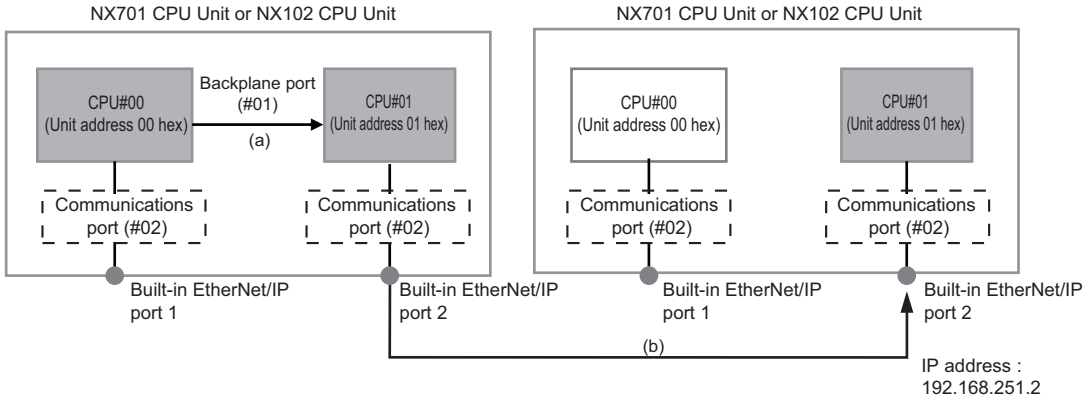
1. Using the built-in EtherNet/IP port 1 (local CPU #00)
   (Local CPU #00 to destination CPU #00)



   a) Local CPU #00 to destination IP address
      - Network type number: "02" (Output to the communications port)
      - Destination address: Specify the destination IP address

      Route Path : 02\192.168.250.2

2. Using the built-in EtherNet/IP port 2 (local CPU #01)
   (Local CPU #00 to destination CPU #01 via local CPU #01)



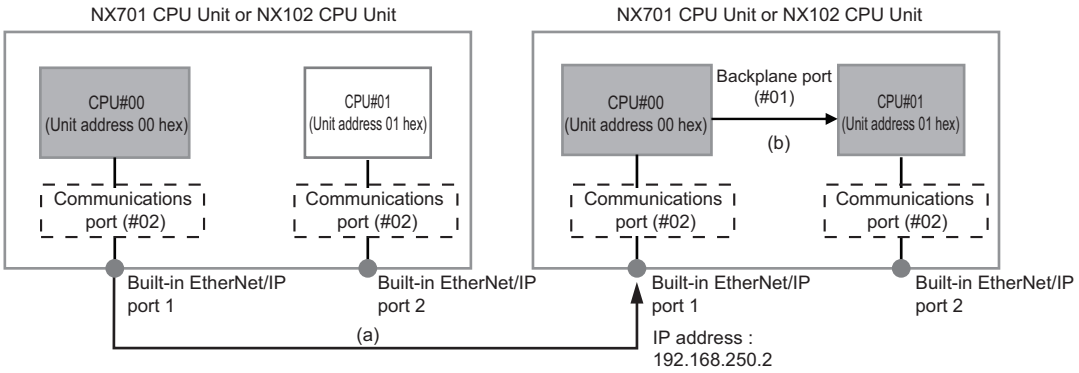
   a) Local backplane to local CPU #01
      - Network type number: "01" (Output to Backplane port)
      - Destination address: "#01" (CPU#01) Note: This is in order to output using the sender IP address of the built-in EtherNet/IP port 2.

   b) Local CPU #01 to destination IP address

- Network type number: "02" (Output to the communications port)
- Destination address: Specify the destination IP address

Route Path : 01\#01 \02\192.168.251.2
(a) (b)

3. Communicating with the destination built-in EtherNet/IP port 2 (destination CPU #01) via the destination built-in EtherNet/IP port 1 (destination CPU #00)
(Local CPU #00 to destination CPU #01 via destination CPU #00)

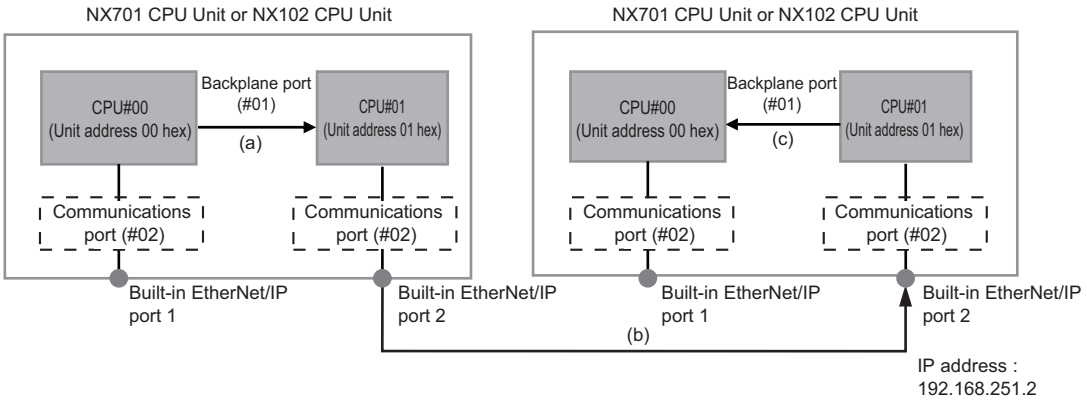


a) Local CPU #00 to destination IP address
   - Network type number: "02" (Output to the communications port)
   - Destination address: Specify the destination IP address

b) Destination backplane to destination CPU #01
   - Network type number: "01" (Output to Backplane port)
   - Destination address: "#01" (CPU#01)

Route Path : 02\192.168.250.2 \01\#01
(a) (b)

4. Communicating with the destination built-in EtherNet/IP port 1 (destination CPU #00) via the destination built-in EtherNet/IP port 2 (destination CPU #01)
(Local CPU #00 to destination CPU #00 via destination CPU #01)



a) Local backplane to local CPU #01
   - Network type number: "01" (Output to Backplane port)

- Destination address: "#01" (CPU#01) Note: This is in order to output using the sender IP address of the built-in EtherNet/IP port 2.

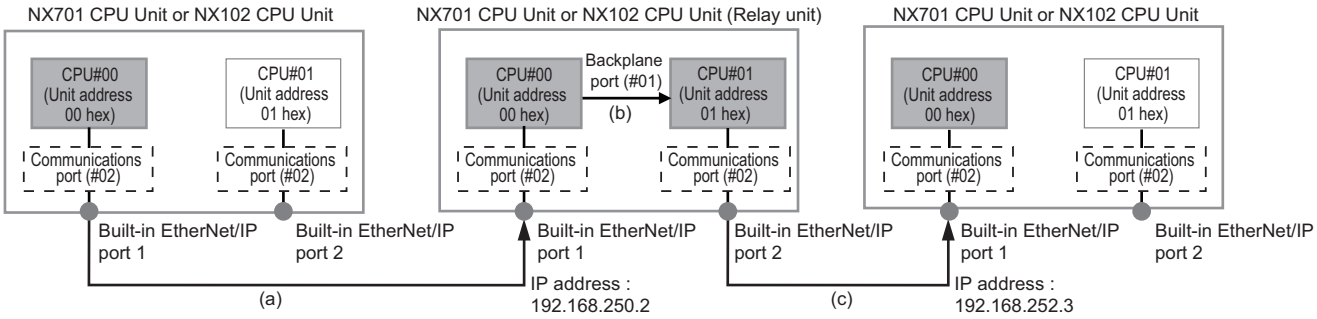
b) Local CPU #01 to destination IP address
- Network type number: "02" (Output to the communications port)
- Destination address: Specify the destination IP address

Route Path : 01\#01 \02\192.168.251.2
(a) (b)

c) Destination CPU #01 to destination CPU #00
- Network type number: "01" (Output to Backplane port)
- Destination address: "#00" (CPU#00)

Route Path : 01\#01 \02\192.168.251.2\01\#00
(a) (b) (c)

5. Using an NX701 CPU or an NX102 CPU Unit as a relay Unit (the built-in EtherNet/IP port 1 to the built-in EtherNet/IP port 2)
(Local CPU #00 to destination CPU #00 via relay CPU #00 and relay CPU #01)



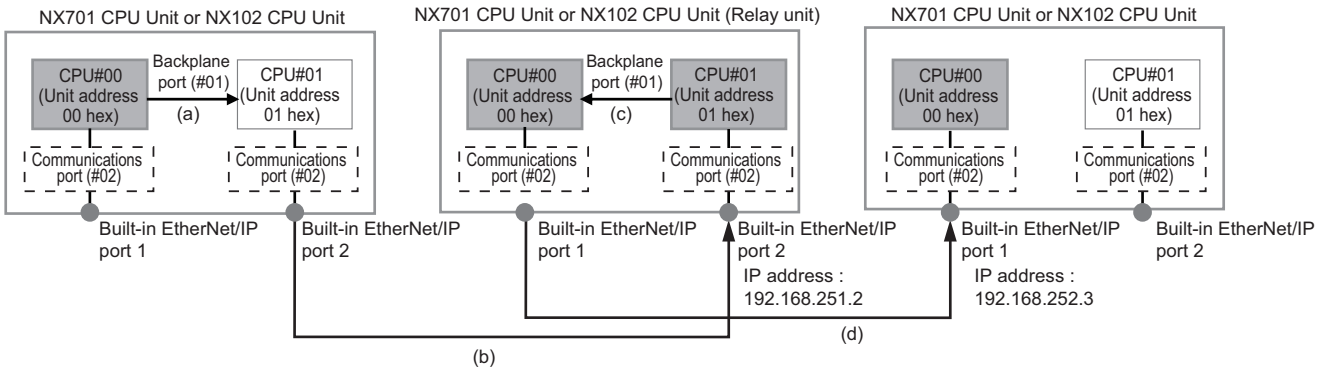
a) Local CPU #00 to relay IP address
- Network type number: "02" (Output to the communications port)
- Destination address: Specify the relay IP address

b) Relay backplane to relay CPU #01
- Network type number: "01" (Output to Backplane port)
- Destination address: "#01" (CPU#01) Note: This is in order to output using the sender IP address of the built-in EtherNet/IP port 2.

c) Relay CPU #01 to destination IP address
- Network type number: "02" (Output to the communications port)
- Destination address: Specify the destination IP address

Route Path : 02\192.168.250.2\01\#01\02\192.168.252.3
(a) (b) (c)

6. Using an NX701 CPU or an NX102 CPU Unit as a relay Unit (the built-in EtherNet/IP port 2 to the built-in EtherNet/IP port 1)
(Local CPU #00 to destination CPU #00 via local CPU #01, relay CPU #01, and relay CPU #00)



a) Local backplane to local CPU #01
- Network type number: "01" (Output to Backplane port)
- Destination address: "#01" (CPU#01) Note: This is in order to output using the sender IP address of the built-in EtherNet/IP port 2.

b) Local CPU #01 to destination IP address
- Network type number: "02" (Output to the communications port)
- Destination address: Specify the destination IP address

c) Relay backplane to relay CPU #00
- Network type number: "01" (Output to Backplane port)
- Destination address: "#00" (CPU#00) Note: This is in order to output using the sender IP address of the built-in EtherNet/IP port 1.

d) Relay CPU #00 to destination IP address
- Network type number: "02" (Output to the communications port)
- Destination address: Specify the destination IP address

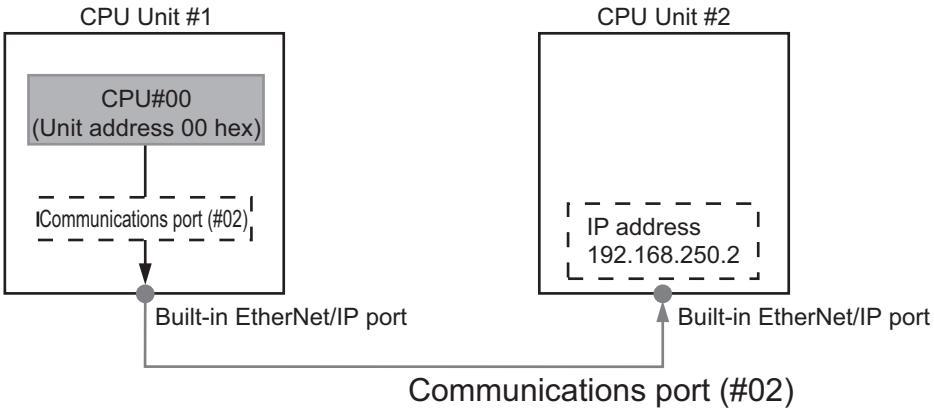
Route Path : 01\#01\02\192.168.251.2\01\#00\02\192.168.252.3
(a) (b) (c) (d)

● NX1P2 CPU Unit

This section provides examples of route paths.

1. Communicating between Built-in EtherNet/IP Ports

Example: Communicating between the built-in EtherNet/IP ports on CPU Unit 1 and CPU Unit 2



- Network type number: "#02" (Output the command via the built-in EtherNet/IP port)
- Destination address: Specify the destination IP address

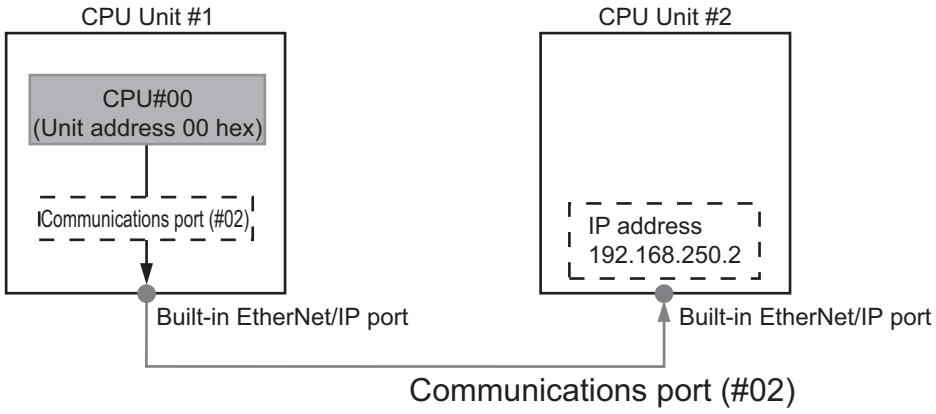
- Route path: 02\192.168.250.2

● NJ-series CPU Unit

The notation of the route path is different for communications on the built-in EtherNet/IP port and for communications on an EtherNet/IP Unit.
This section provides examples of route paths.

1. Communicating between Built-in EtherNet/IP Ports

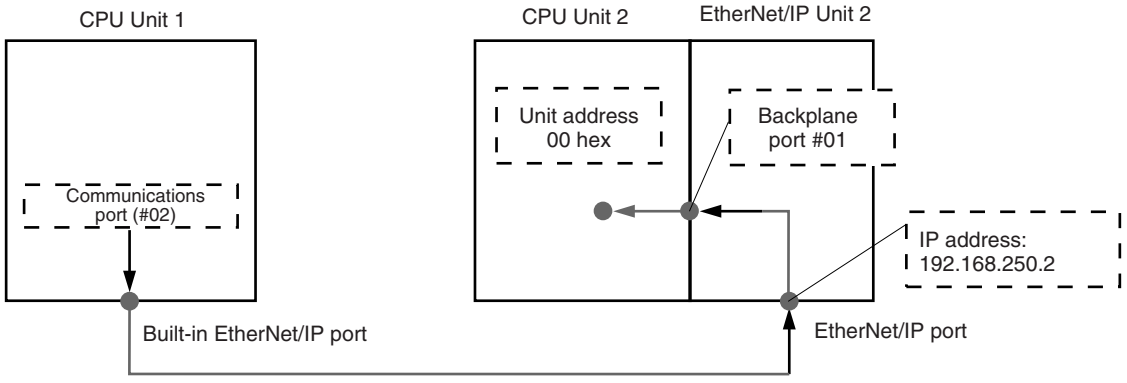
   Example: Communicating between the built-in EtherNet/IP ports on CPU Unit 1 and CPU Unit 2



- Network type number: "#02" (Output the command via the EtherNet/IP port)
- Destination address: Specify the destination IP address
- Route path: 02\192.168.250.2

2. Communicating from a Built-in EtherNet/IP Port to an EtherNet/IP Unit

   Example: Communicating from the built-in EtherNet/IP port on CPU Unit 1 to CPU Unit 2 via the EtherNet/IP Unit mounted to CPU Unit 2



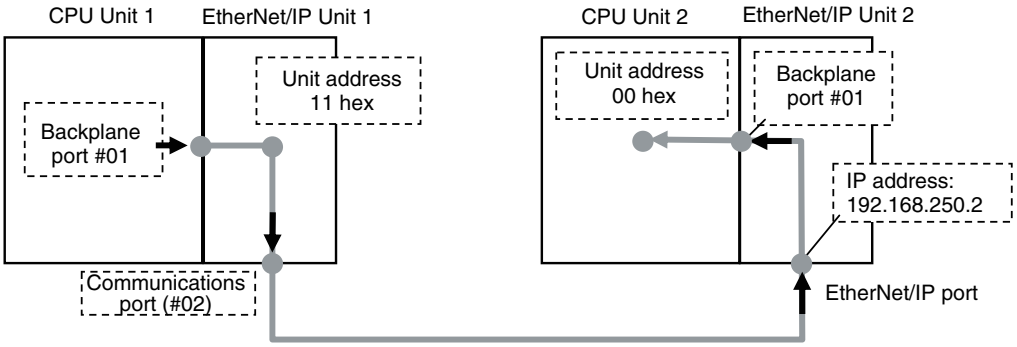
a) CPU Unit 1 to EtherNet/IP Unit 2
   - Network type number: "#02" (Output the command via the EtherNet/IP port)
   - Destination address: Specify the destination IP address

b) EtherNet/IP Unit 2 to CPU Unit 2
   - Network type number: "#01" (Output the command via the internal backplane port)
   - Destination address: "#00" (Unit address of the CPU Unit)

Route path : 02\192.168.250.2\01\#00
(1) (2)

3. Communicating between EtherNet/IP Units

Example: Communicating via EtherNet/IP Units mounted to CPU Unit 1 and CPU Unit 2



a) CPU Unit 1 to EtherNet/IP Unit 1
- Network type number: "#01" (Output the command via the internal backplane port)
- Destination address: "#11" (Unit address of EtherNet/IP Unit (Unit number: 1+10 hex))

b) EtherNet/IP Unit 1 to EtherNet/IP Unit 2
- Network type number: "#02" (Output the command via the EtherNet/IP port)
- Destination address: Specify the destination IP address

c) EtherNet/IP Unit 2 to CPU Unit 2
- Network type number: "#01" (Output the command via the internal backplane port)
- Destination address: "#00" (Unit address of the CPU Unit)

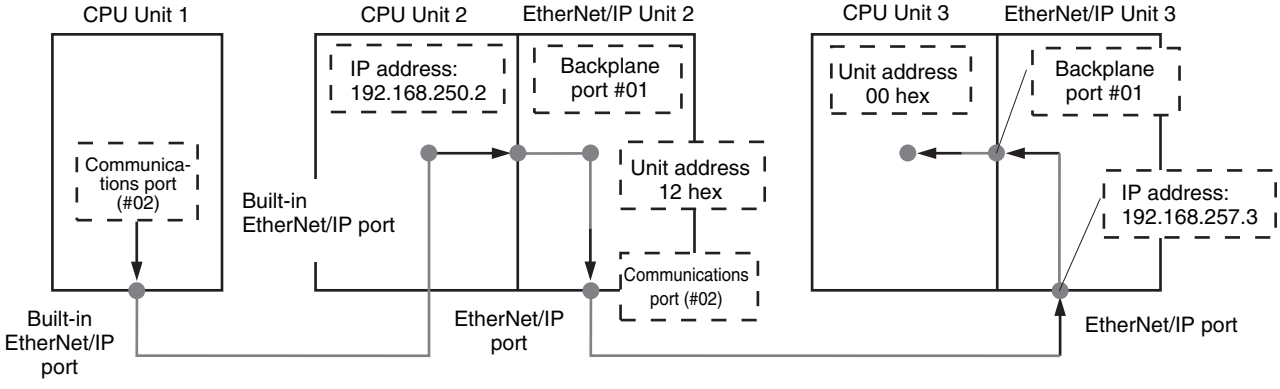
Route path : 01\#11\02\192.168.250.2\01\#00
(1) (2) (3)

Version Information

You can use the CJ1W-EIP21 EtherNet/IP Unit mounted to an NJ-series Controller with a CPU Unit with unit version 1.01 or later and Sysmac Studio version 1.02 or higher.

4. Accessing via a Relay Node

Example: Communicating from CPU Unit 1 to CPU Unit 3 via CPU Unit 2



a) CPU Unit 1 to CPU Unit 2

- Network type number: "#02" (Output the command via the EtherNet/IP port)
- Destination address: Specify the destination IP address

b) CPU Unit 2 to EtherNet/IP Unit 2
- Network type number: "#01" (Output the command via the internal backplane port)
- Destination address: "#12 hex" (Unit address of the EtherNet/IP Unit (Unit number: 2+10 hex =12 hex))

c) EtherNet/IP Unit 2 to EtherNet/IP Unit 3
- Network type number: "#02" (Output the command via the EtherNet/IP port)
- Destination address: Specify the destination IP address

d) EtherNet/IP Unit 3 to CPU Unit 3
- Network type number: "#01" (Output the command via the internal backplane port)
- Destination address: "#00" (Unit address of the CPU Unit)

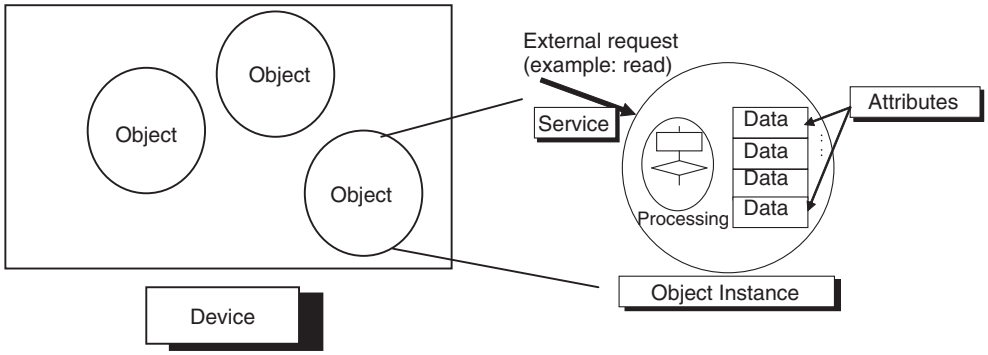
Route path : 02\192.168.250.2\01\#12\02\192.168.257.3\01\#00
(1) (2) (3) (4)

7-2-5 Request Path (IOI)

A request path indicates an object of a device on the network.
A CIP communications instruction uses the request path to access an object of a device.

Overview of Request Path

In the CIP world, each device is modeled as a collection of objects. An Object abstractly represents the specific configuration elements of a device.

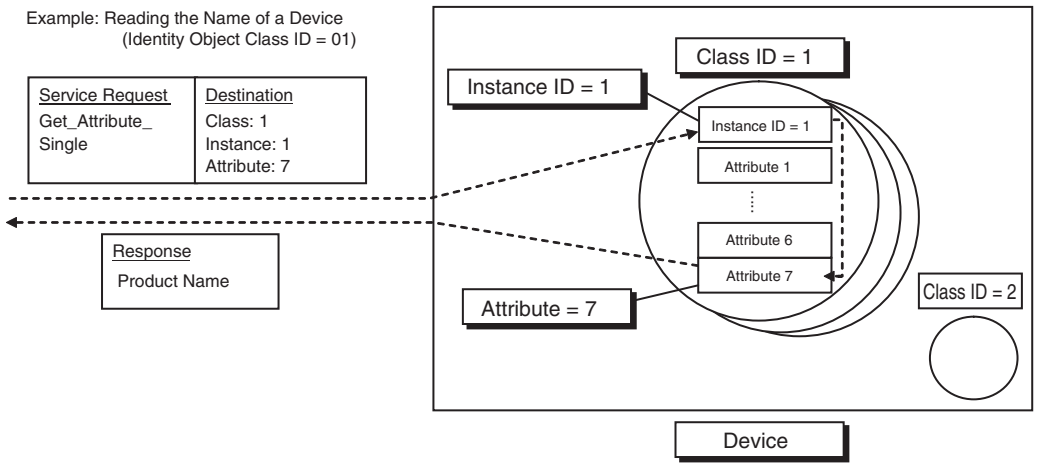


In the CIP Common Specification, Object, Class, Instance, Attribute, and Service are defined as follows: (Source: CIP Common Specification)

Term	Definition
Object	An abstract representation of a particular component within a device.
Class	A set of objects that all represent the same kind of system component.
Instance	A specific and real (physical) occurrence of an object.
Attribute	A description of an externally visible characteristic or feature of an object.
Service	A request from an external object (e.g., to read data).

You use the Class ID Instance ID and Attribute ID to access an object.
You specify these three IDs to designate an object in a device.

When you make a request from an external device for a service, you must specify the Class ID Instance ID and Attribute ID. (The Instance ID and Attribute ID are not required for some services.)



These are called *IOI* (Internal Object Identifier) because they identify the Class ID, Instance ID, and Attribute ID within the device.
Refer to *7-5 CIP Object Services* on page 7-48 for the class ID, instance ID, attribute ID, and service code for each object.

Providing the Structure Variables to Input Request Paths

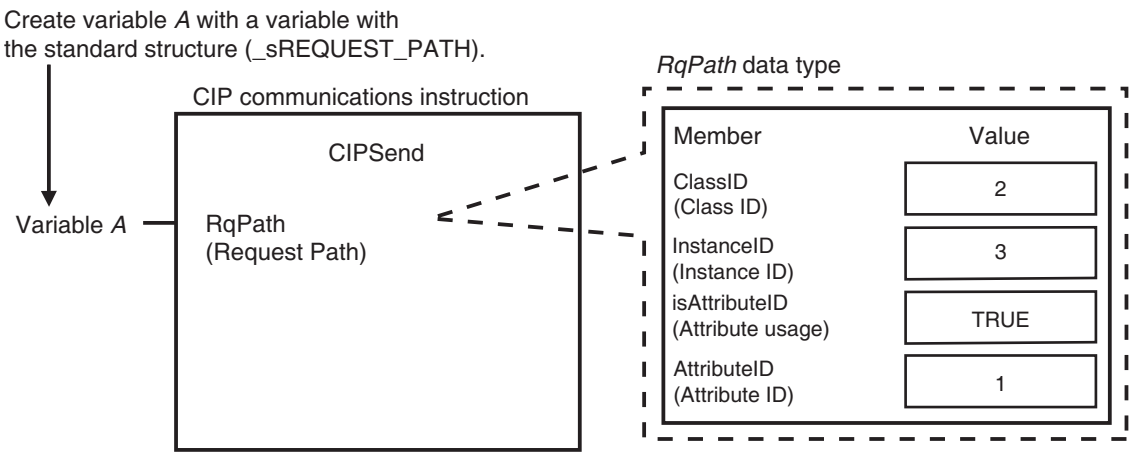
For a CIP communications instruction, you prepare a variable to store the request path. In this variable, you specify the object to access with the user program.
A structure in which the Class ID, Instance ID, and Attribute ID are specified is provided for the data type of a variable for a request path.
There are two types of structures: standard structure (_sREQUEST_PATH) and extension structure (_sREQUEST_PATH_EX). When you use an extension structure, it is possible to specify the size according to the size of values of the Class ID, Instance ID, and Attribute ID of the object that you access. When you use a standard structure, the size is always set to 16 bits.

Version Information

A CPU Unit with unit version 1.11 or later and Sysmac Studio version 1.15 or higher are required to specify extension structure (_sREQUEST_PATH_EX).

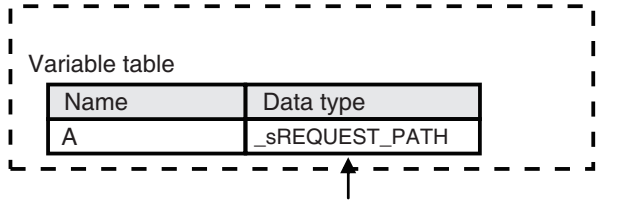
● When a Standard Structure Variable Is Used

Example: Using a standard structure variable to input values into *RqPath* (Request Path) for the CIPSend instruction



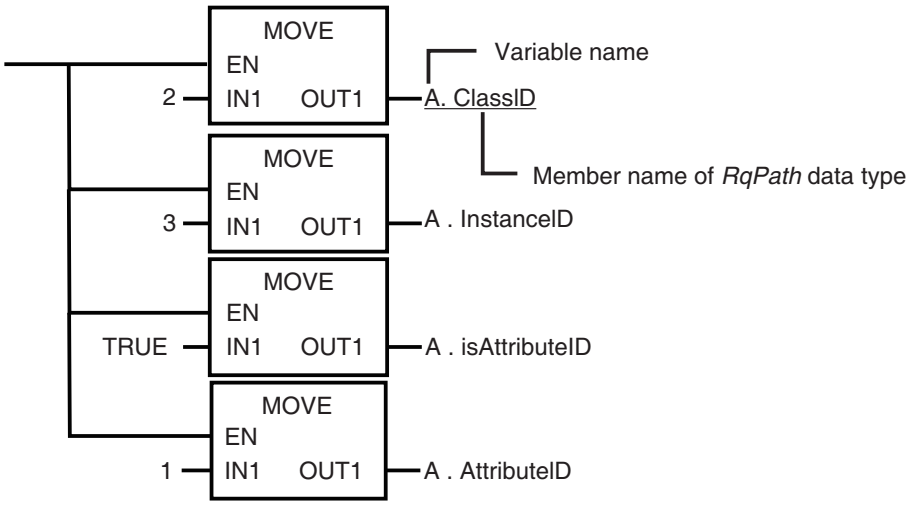
1 Create a standard structure variable.

To use a standard structure variable to input values into *RqPath* (Request Path) for a CIP communications instruction, first you need to create a standard structure user-defined variable. When you create a variable in a variable table, select the pre-registered standard structure (_sREQUEST_PATH) for a CIP communications instruction.



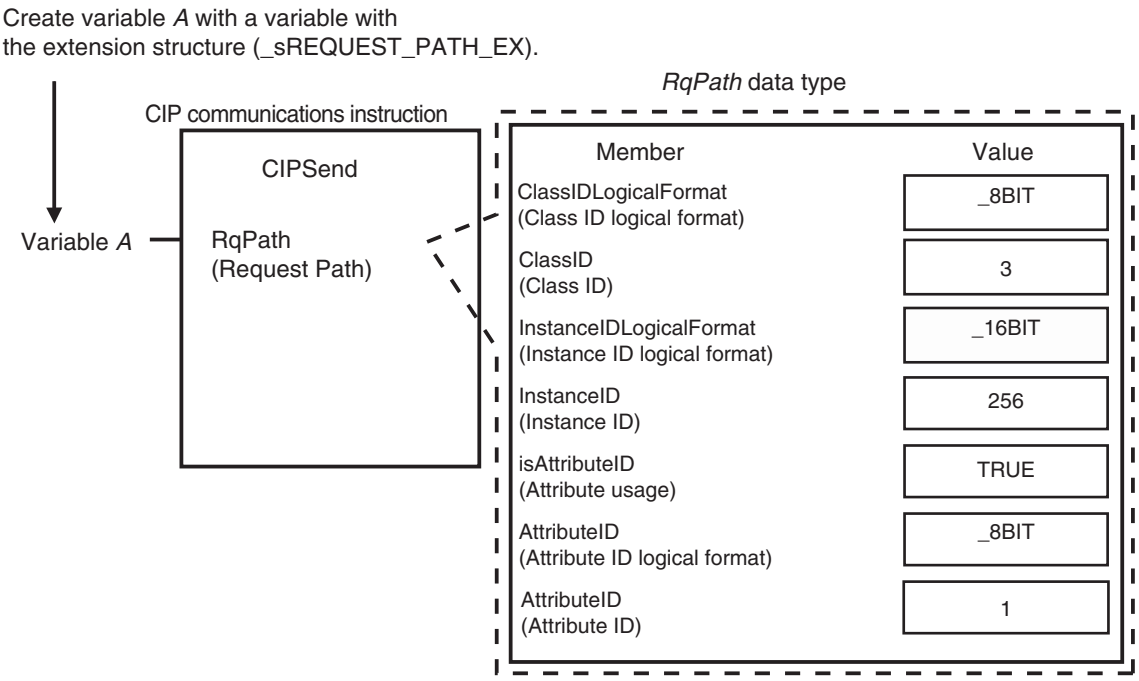
2 Input a value for each standard structure variable member.

Input the following values into the communications parameters that were registered as members of the standard structure variable.



● When an Extension Structure Variable Is Used

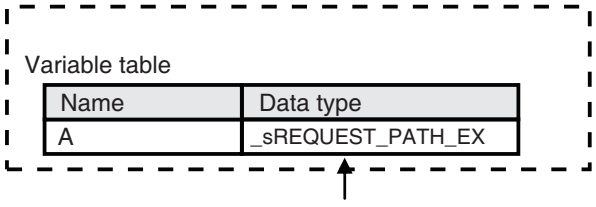
Example: Using an extension structure variable to input values into *RqPath* (Request Path) for the CIPSend instruction



1 Create an extension structure variable.

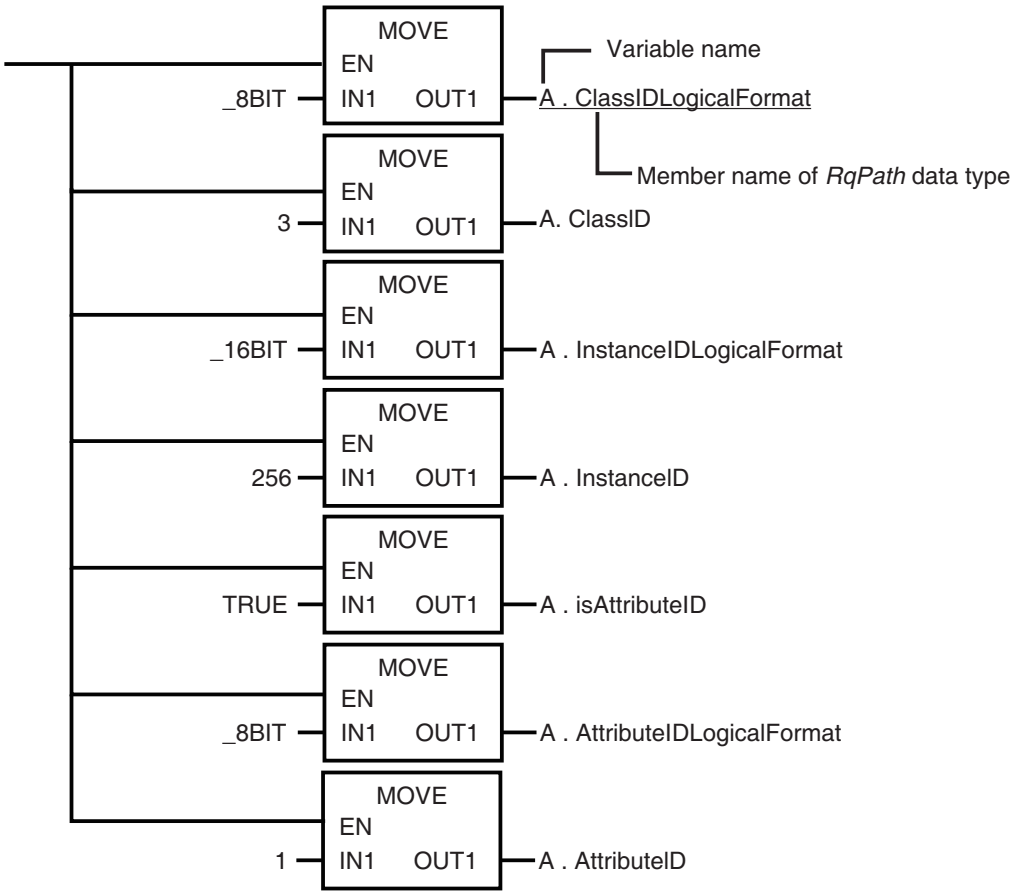
To use an extension structure variable to input values into *RqPath* (Request Path) for a CIP communications instruction, first you need to create an extension structure user-defined variable.

When you create a variable in a variable table, select the pre-registered extension structure (_sREQUEST_PATH_EX) for a CIP communications instruction.



2 Input a value for each extension structure variable member.

Input the following values into the communications parameters that were registered as members of the extension structure variable.



7-2-6 Service Data and Response Data

CIP communications instructions send and receive data that is stored in array variables.

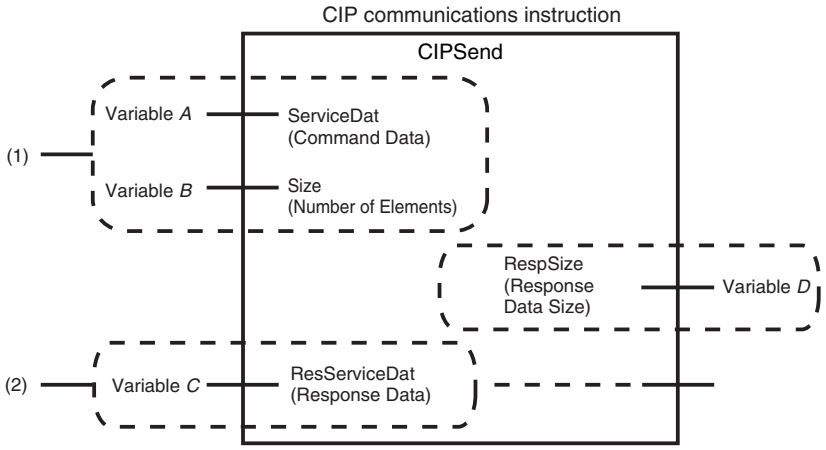
Preparing Array Variables to Input and Output Service Data and Response Data

This section describes the array variables for storing service data and response data that CIP communications instructions send and receive.

● Creating Array Variables

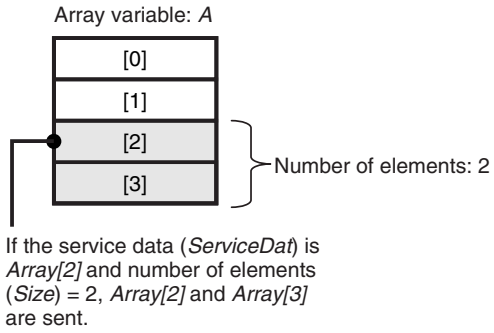
To input a value into the array variable of a CIP communications instruction, you must create a variable with the same configuration as the array variable in advance.

Example: Creating a Variable to Input Data to the CIPSend Instruction Array Variables



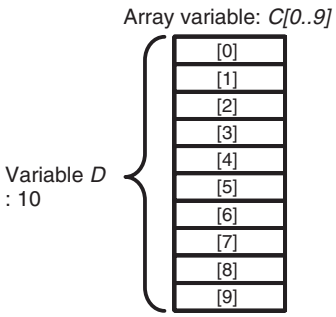
(1) Input the service data to send

The data to send is stored in array variable *A*.
If only certain elements are specified in array variable *A*, specify the number of elements in variable *B*.



(2) Store received response data

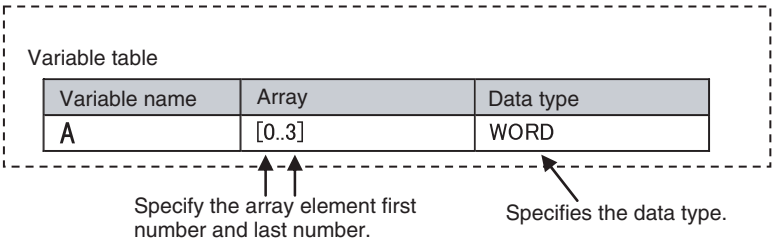
The data that is received is stored in variable *C*.
The byte size of the data that was actually received is stored in variable *D*.



Use the following procedure to create a variable in the variable table.

Specify the element first number, the element last number, and the data type.

Example: UINT Array



● CIP Communications Instructions That Use Array Variables

Instruction	Structure variable name		
	Input variable	In-out variable	Output variable
CIPRead	---	---	DstDat (Read Data)
CIPWrite	SrcDat (Write Data)	---	---
CIPSend	ServiceDat (Command Data)	ResServiceDat (Response Data)	---

7-2-7 Sample Programming for CIP Connectionless (UCMM) Message Communications

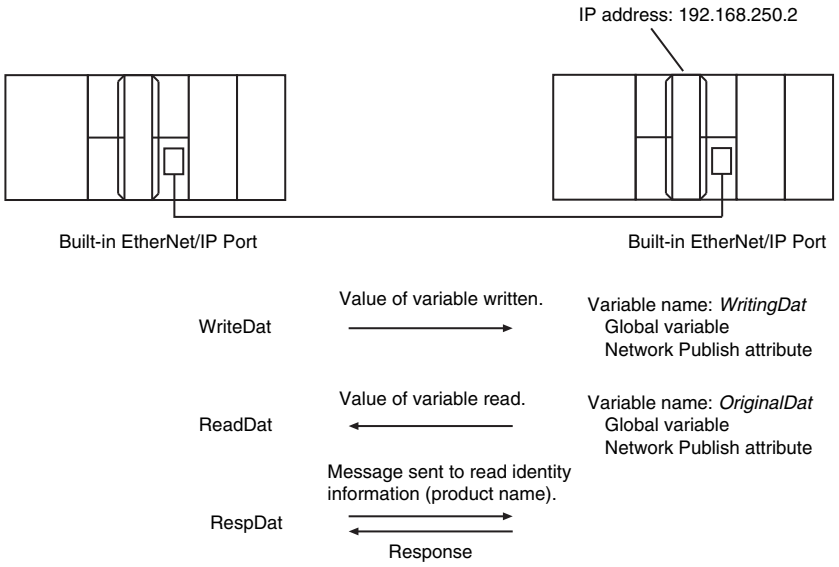
This sample uses CIP UCMM messages to write a variable, read a variable, and send a message. The Controllers are connected to an EtherNet/IP network. The IP address of the remote node is 192.168.250.2.
The following procedure is used.

1 The CIPUCMMWrite instruction is used to write the value of a variable at a remote node. The variable name at the remote node is *WritingDat* and the contents of the *WriteDat* is written to it. *WritingDat* must be defined as a global variable at the remote node and the Network Publish attribute must be set.

2 The CIPUCMMRead instruction is used to read the value of a variable at a remote node. The value of the variable *OriginalDat* at the other node is read and the read value is stored in the *ReadDat* variable. *OriginalDat* must be defined as a global variable at the remote node and the Network Publish attribute must be set.

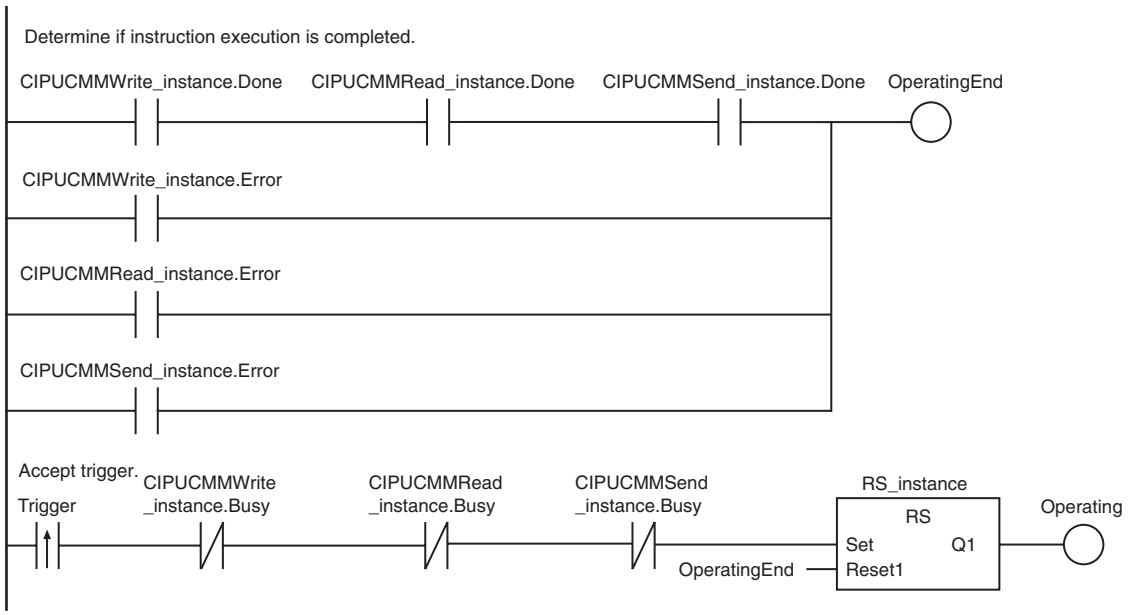
3 The CIPUCMMSend instruction is used to send an explicit message to a remote node. The contents of the message is to read identity information (product name). The class ID, instance ID, attribute ID, and service code are as follows. The response data is stored in the *RespDat* variable.

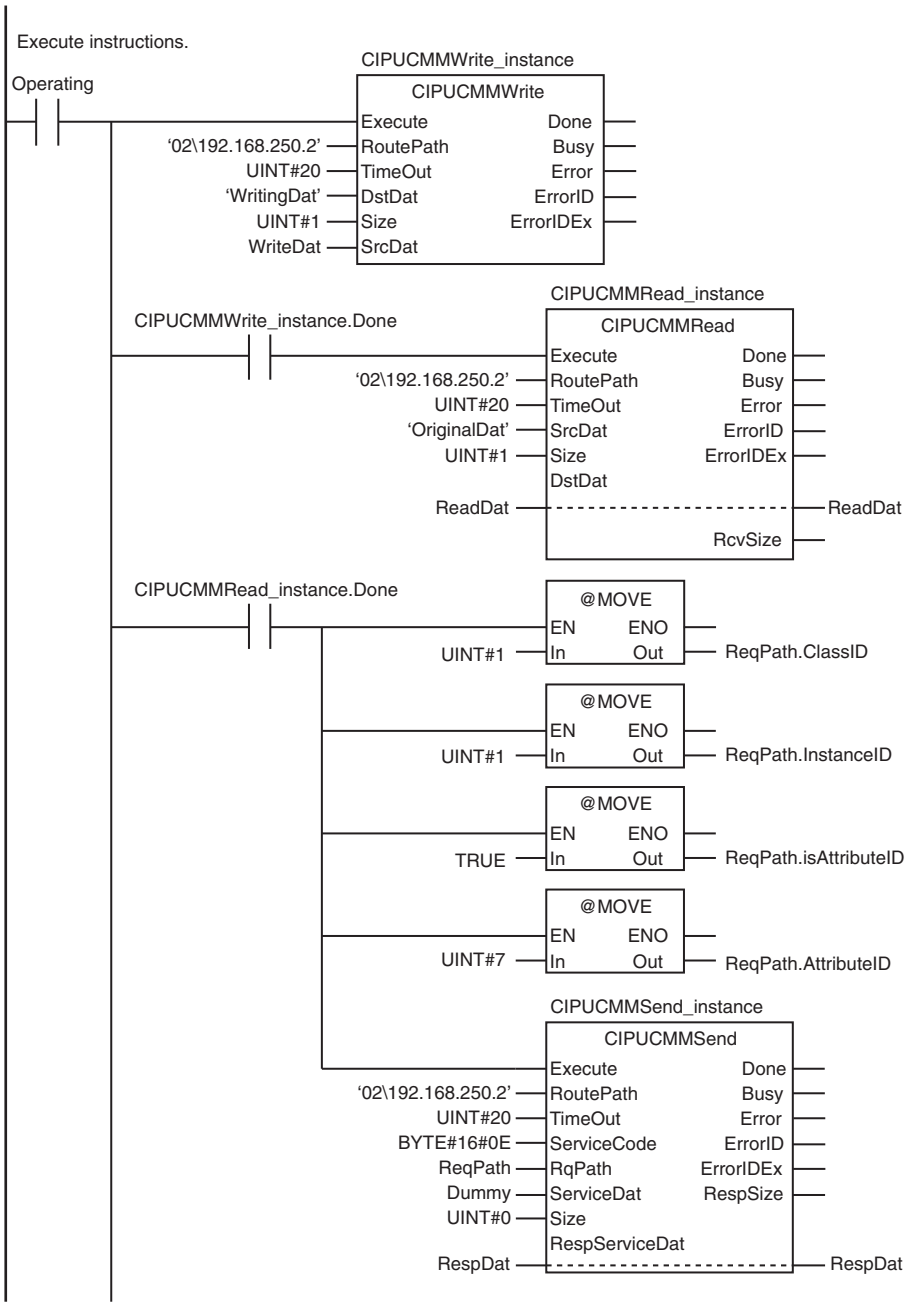
Item	Value
Class ID	1
Instance ID	1
Attribute ID	7
Service Code	16#0E

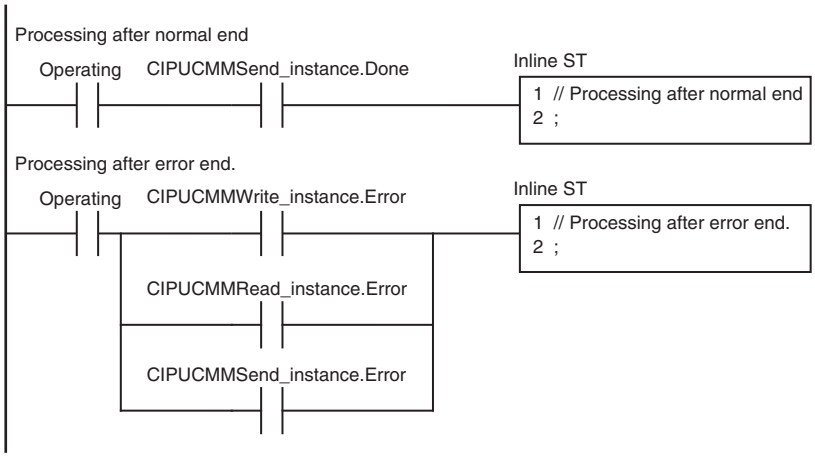


LD

Variable	Data type	Initial value	Comment
OperatingEnd	BOOL	False	Processing completed
Trigger	BOOL	False	Execution condition
Operating	BOOL	False	Processing
WriteDat	INT	1234	Write data
ReadDat	INT	0	Read data
ReqPath	_sREQUEST_PATH	(ClassID:=0, InstanceID:=0, isAttributeID:=False, AttributeID:=0)	Request path
RespDat	ARRAY[0..10] OF BYTE	[11(16#0)]	Response data
Dummy	BYTE	16#0	Dummy
RS_instance	RS		
CIPUCMMWrite_instance	CIPUCMMWrite		
CIPUCMMRead_instance	CIPUCMMRead		
CIPUCMMSend_instance	CIPUCMMSend		




Execute instructions.
Operating
CIPUCMMWrite_instance
CIPUCMMWrite
Execute
Done
'02\192.168.250.2'
RoutePath
Busy
UINT#20
TimeOut
Error
'WritingDat'
DstDat
ErrorID
UINT#1
Size
ErrorIDEx
WriteDat
SrcDat
CIPUCMMRead_instance
CIPUCMMWrite_instance.Done
CIPUCMMRead
Execute
Done
'02\192.168.250.2'
RoutePath
Busy
UINT#20
TimeOut
Error
'OriginalDat'
SrcDat
ErrorID
UINT#1
Size
ErrorIDEx
DstDat
ReadDat
ReadDat
RcvSize
CIPUCMMRead_instance.Done
@MOVE
EN
ENO
UINT#1
In
Out
ReqPath.ClassID
@MOVE
EN
ENO
UINT#1
In
Out
ReqPath.InstanceID
@MOVE
EN
ENO
TRUE
In
Out
ReqPath.isAttributeID
@MOVE
EN
ENO
UINT#7
In
Out
ReqPath.AttributeID
CIPUCMMSend_instance
CIPUCMMSend
Execute
Done
'02\192.168.250.2'
RoutePath
Busy
UINT#20
TimeOut
Error
BYTE#16#0E
ServiceCode
ErrorID
ReqPath
RqPath
ErrorIDEx
Dummy
ServiceDat
RespSize
UINT#0
Size
RespServiceDat
RespDat
RespDat


Processing after normal end
Operating
CIPUCMMSend_instance.Done
Inline ST
1 // Processing after normal end
2 ;
Processing after error end.
Operating
CIPUCMMWrite_instance.Error
Inline ST
1 // Processing after error end.
2 ;
CIPUCMMRead_instance.Error
CIPUCMMSend_instance.Error

ST

Internal variables	Variable	Data type	Initial value	Comment
	Trigger	BOOL	False	Execution condition
	DoUCMMTrigger	BOOL	False	Processing
	Stage	INT	0	Status change
	WriteDat	INT	1234	Write data
	ReadDat	INT	0	Read data
	ReqPath	_sREQUEST_PATH	(ClassID:=0, InstanceID:=0, isAttributeID:=False, AttributeID:=0)	Request path
	RespDat	ARRAY[0..10] OF BYTE	[11(16#0)]	Response data
	Dummy	BYTE	16#0	Dummy
	CIPUCMMWrite_instance	CIPUCMMWrite		
	CIPUCMMRead_instance	CIPUCMMRead		
	CIPUCMMSend_instance	CIPUCMMSend		

External variable	Variable	Data type	Constant	Comment
	_EIP_EtnOnlineSta*1	BOOL	☑	Online

*1. For an NX701 CPU Unit and an NX102 CPU Unit, replace the variable with _EIP1_EtnOnlineSta (Port1 Online) or _EIP2_EtnOnlineSta (Port2 Online), depending on the built-in EtherNet/IP port which is used.
For an NX1P2 CPU Unit, replace the variable with _EIP1_EtnOnlineSta (Port1 Online).

```
// Start sequence when Trigger changes to TRUE
IF ((Trigger=TRUE) AND (DoUCMMTrigger=FALSE) AND (_EIP_EtnOnlineSta=TRU
E))
THEN
DoUCMMTrigger            :=TRUE;
Stage                    :=INT#1;
CIPUCMMWrite_instance(
Execute              :=FALSE,                    // Initialize instan
ce
SrcDat               :=WriteDat);                // Dummy
CIPUCMMRead_instance(                                // Initialize in
stance
Execute              :=FALSE,                    // Dummy
DstDat               :=ReadDat);                 // Dummy
CIPUCMMSend_instance(
Execute              :=FALSE,                    // Initialize instan
ce
ServiceDat           := Dummy,                   // Dummy
RespServiceDat       :=RespDat);                 // Dummy
END_IF;
```

```
            IF (DoUCMMTrigger=TRUE) THEN
            CASE Stage OF
            1 :                                                 // Request writi
ng value of variable
            CIPUCMMWrite_instance(
            Execute          :=TRUE,
            RoutePath        :='02\192.168.250.2',         // Route path
            TimeOut          :=UINT#20,                    // Timeout time
            DstDat           :='WritingDat',               // Destination variable
name
            Size             :=UINT#1,                     // Number of elements to
 write
            SrcDat           :=WriteDat);                  // Write data

            IF (CIPUCMMWrite_instance.Done=TRUE) THEN
            Stage            :=INT#2;                      // Normal end
            ELSIF (CIPUCMMWrite_instance.Error=TRUE) THEN
            Stage                :=INT#10;                     // Error end
            END_IF;
            2 :                                                 // Request readi
ng value of variable
            CIPUCMMRead_instance(
            Execute          :=TRUE,
            RoutePath        :='02\192.168.250.2',         // Route path
            TimeOut          :=UINT#20,                    // Timeout time
            SrcDat           :='OriginalDat',              // Source variable name
            Size             :=UINT#1,                     // Number of elements to
 read
            DstDat           :=ReadDat);                   // Read data

            IF (CIPUCMMRead_instance.Done=TRUE) THEN
            Stage            :=INT#3;                      // Normal end
            ELSIF (CIPUCMMRead_instance.Error=TRUE) THEN
            Stage            :=INT#40;                     // Error end
            END_IF;

            3 :                                                 // Send message
            ReqPath.ClassID      :=UINT#01;
            ReqPath.InstanceID   :=UINT#01;
            ReqPath.isAttributeID:=TRUE;
            ReqPath.AttributeID  :=UINT#07;
            CIPUCMMSend_instance(
            Execute          :=TRUE,
            RoutePath        :='02\192.168.250.2',         // Route path
            TimeOut          :=UINT#20,                    // Timeout time
            ServiceCode      :=BYTE#16#0E,                 // Service code
```

```
            RqPath          :=ReqPath,                  // Request path
            ServiceDat      :=Dummy,                    // Service data
            Size            :=UINT#0,                   // Number of elements
            RespServiceDat  :=RespDat);                 // Response data

            IF (CIPUCMMSend_instance.Done=TRUE) THEN
            Stage           :=INT#0;                    // Normal end
            ELSIF (CIPUCMMSend_instance.Error=TRUE) THEN
            Stage           :=INT#30;                   // Error end
            END_IF;

            0:                                                  // Processing af
ter normal end
            DoUCMMTrigger       :=FALSE;
            Trigger             :=FALSE;

            ELSE                                                // Processing af
ter error end
            DoUCMMTrigger       :=FALSE;
            Trigger             :=FALSE;
            END_CASE;
            END_IF;
```

7-2-8 Sample Programming for CIP Connection (Class 3) Message Communications

This sample uses CIP class 3 messages to write a variable, read a variable, and send a message. The Controllers are connected to an EtherNet/IP network. The IP address of the remote node is 192.168.250.2.
The following procedure is used.

1 The CIPOpen is used to open a class 3 connection (Large_Forward_Open). The timeout time is 2 s.

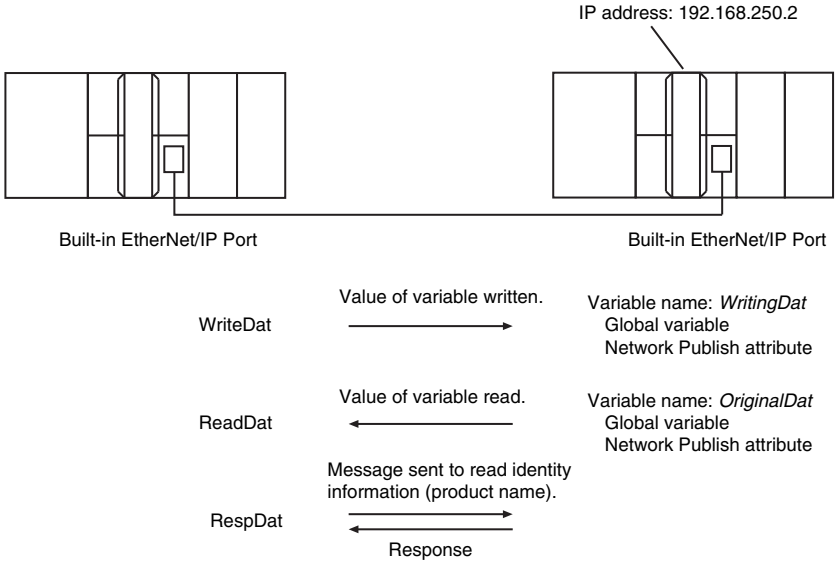
2 The CIPWrite instruction is used to write the value of a variable at a remote node. The variable name at the remote node is *WritingDat* and the contents of the *WriteDat* is written to it. *WritingDat* must be defined as a global variable at the remote node and the Network Publish attribute must be set.

3 The CIPRead instruction is used to read the value of a variable at a remote node. The value of the variable *OriginalDat* at the other node is read and the read value is stored in the *ReadDat* variable. *OriginalDat* must be defined as a global variable at the remote node and the Network Publish attribute must be set.

4 The CIPSend instruction is used to send an explicit message to a remote node. The contents of the message is to read identity information (product name). The class ID, instance ID, attribute ID, and service code are as follows. The response data is stored in the RespDat variable.

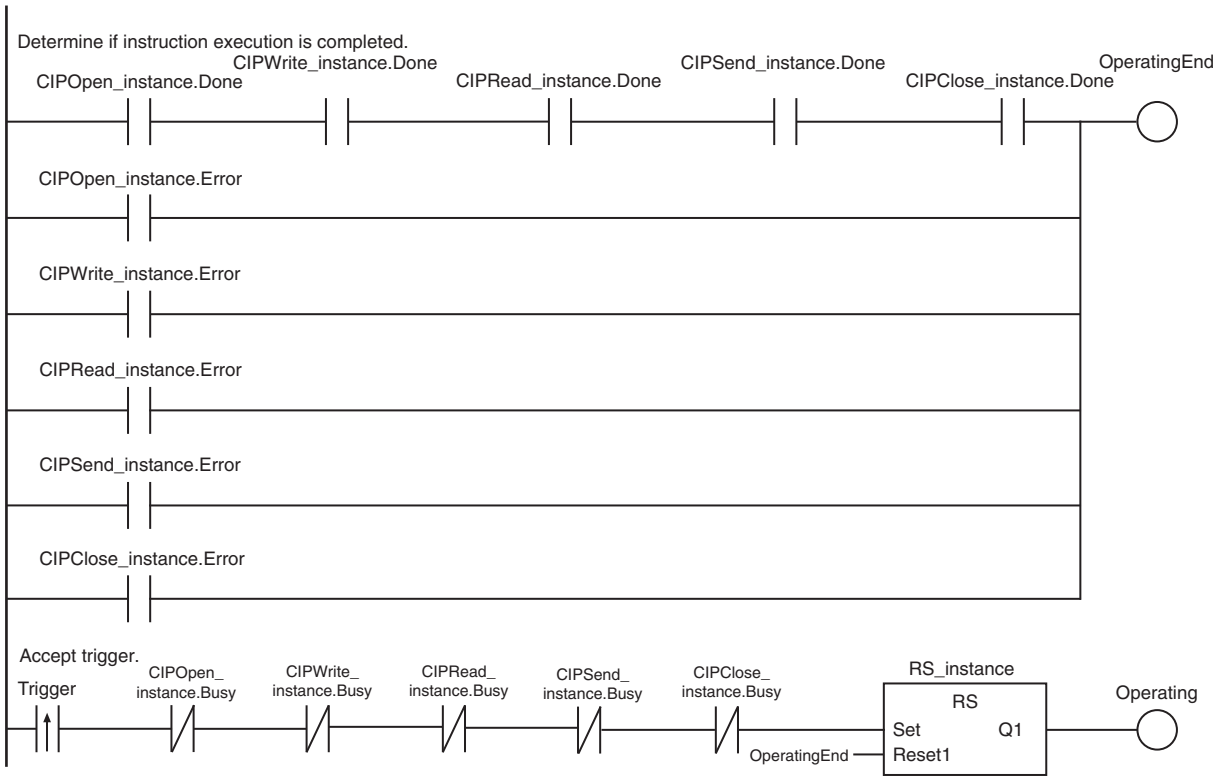
Item	Value
Class ID	1
Instance ID	1
Attribute ID	7
Service Code	16#0E

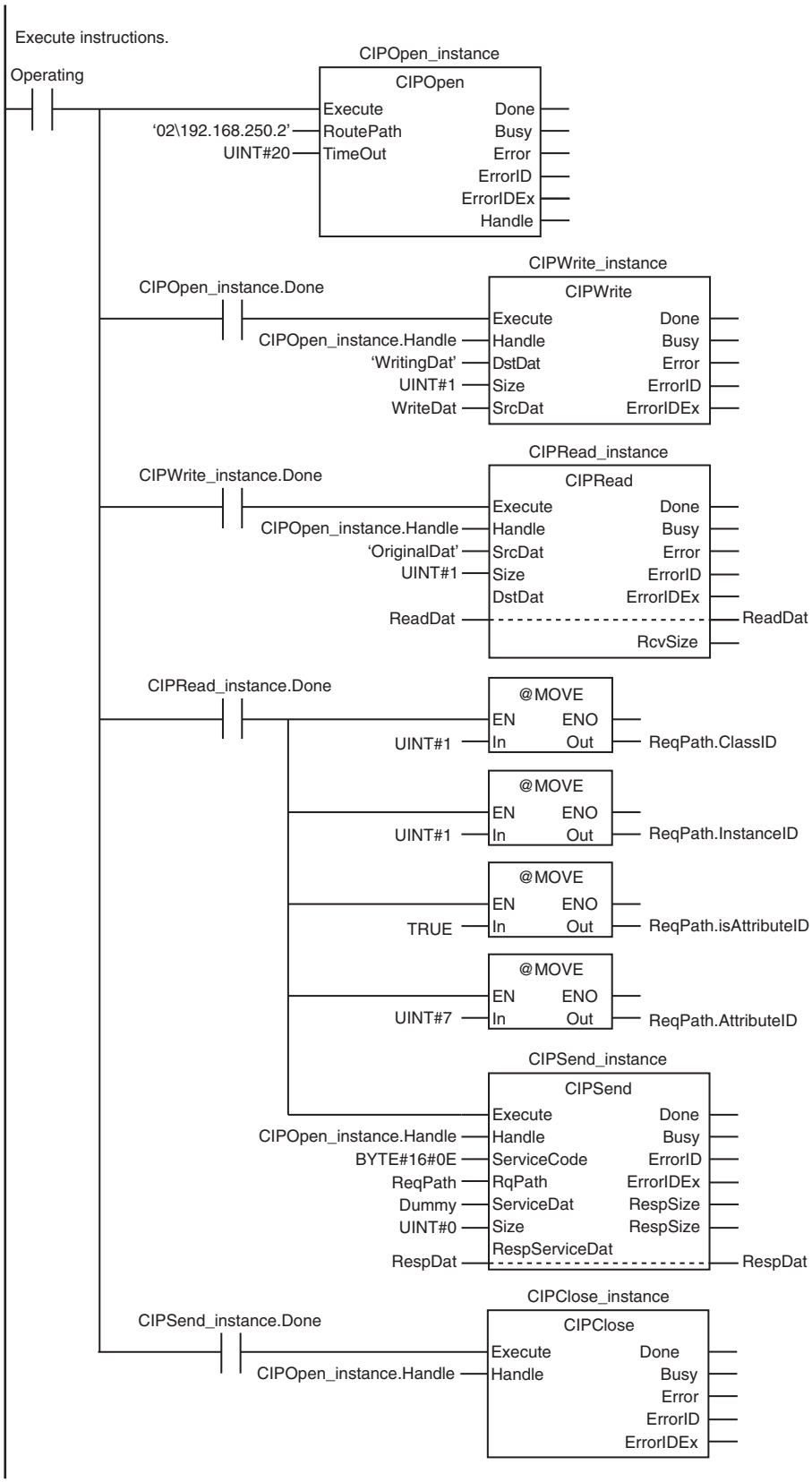
5 The CIPClose instruction is used to close the class 3 connection.

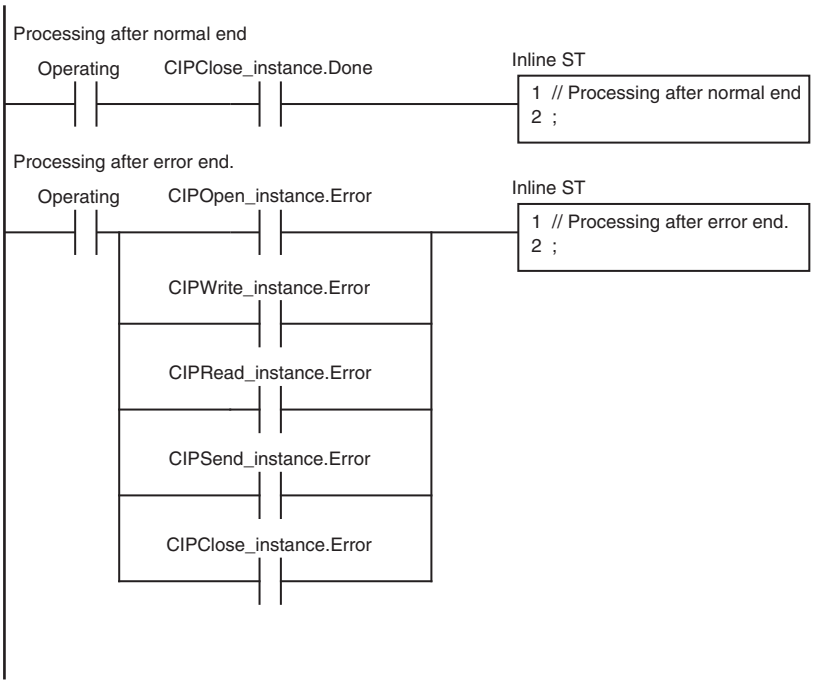


LD

Variable	Data type	Initial value	Comment
OperatingEnd	BOOL	False	Processing completed
Trigger	BOOL	False	Execution condition
Operating	BOOL	False	Processing
WriteDat	INT	1234	Write data
ReadDat	INT	0	Read data
ReqPath	_sREQUEST_PATH	(ClassID:=0, InstanceID:=0, isAttributeID:=False, AttributeID:=0)	Request path
RespDat	ARRAY[0..10] OF BYTE	[11(16#0)]	Response data
Dummy	BYTE	16#0	Dummy
RS_instance	RS		
CIPOpen_instance	CIPOpen		
CIPWrite_instance	CIPWrite		
CIPRead_instance	CIPRead		
CIPSend_instance	CIPSend		
CIPClose_instance	CIPClose		


Determine if instruction execution is completed.
CIPOpen_instance.Done
CIPWrite_instance.Done
CIPRead_instance.Done
CIPSend_instance.Done
CIPClose_instance.Done
OperatingEnd
CIPOpen_instance.Error
CIPWrite_instance.Error
CIPRead_instance.Error
CIPSend_instance.Error
CIPClose_instance.Error
Accept trigger.
Trigger
CIPOpen_ instance.Busy
CIPWrite_ instance.Busy
CIPRead_ instance.Busy
CIPSend_ instance.Busy
CIPClose_ instance.Busy
RS_instance
RS
Set
Q1
Reset1
OperatingEnd
Operating


Execute instructions.
Operating
CIPOpen_instance
CIPOpen
Execute
RoutePath
TimeOut
Done
Busy
Error
ErrorID
ErrorIDEx
Handle
'02\192.168.250.2'
UINT#20
CIPWrite_instance
CIPWrite
CIPOpen_instance.Done
CIPOpen_instance.Handle
'WritingDat'
UINT#1
WriteDat
Execute
Handle
DstDat
Size
SrcDat
Done
Busy
Error
ErrorID
ErrorIDEx
CIPRead_instance
CIPRead
CIPWrite_instance.Done
CIPOpen_instance.Handle
'OriginalDat'
UINT#1
ReadDat
Execute
Handle
SrcDat
Size
DstDat
Done
Busy
Error
ErrorID
ErrorIDEx
ReadDat
RcvSize
CIPRead_instance.Done
@MOVE
EN
In
ENO
Out
UINT#1
ReqPath.ClassID
UINT#1
ReqPath.InstanceID
TRUE
ReqPath.isAttributeID
UINT#7
ReqPath.AttributeID
CIPSend_instance
CIPSend
CIPOpen_instance.Handle
BYTE#16#0E
ReqPath
Dummy
UINT#0
RespDat
Execute
Handle
ServiceCode
RqPath
ServiceDat
Size
RespServiceDat
Done
Busy
ErrorID
ErrorIDEx
RespSize
RespSize
RespDat
CIPClose_instance
CIPClose
CIPSend_instance.Done
CIPOpen_instance.Handle
Execute
Handle
Done
Busy
Error
ErrorID
ErrorIDEx



ST

Internal variables	Variable	Data type	Initial value	Comment
	Trigger	BOOL	False	Execution condition
	DoCIPTrigger	BOOL	False	Processing
	Stage	INT	0	Status change
	WriteDat	INT	1234	Write data
	ReadDat	INT	0	Read data
	ReqPath	_sREQUEST_PATH	(ClassID:=0, InstanceID:=0, isAttributeID:=False, AttributeID:=0)	Request path
	RespDat	ARRAY[0..10] OF BYTE	[11(16#0)]	Response data
	Dummy	BYTE	16#0	Dummy
	CIPOpen_instance	CIPOpen		
	CIPWrite_instance	CIPWrite		
	CIPRead_instance	CIPRead		
	CIPSend_instance	CIPSend		
	CIPClose_instance	CIPClose		

External variable	Variable	Data type	Constant	Comment
	_EIP_EtnOnlineSta*1	BOOL	☑	Online

*1. For an NX701 CPU Unit and an NX102 CPU Unit, replace the variable with _EIP1_EtnOnlineSta (Port1 Online) or _EIP2_EtnOnlineSta (Port2 Online), depending on the built-in EtherNet/IP port which is used.

For an NX1P2 CPU Unit, replace the variable with _EIP1_EtnOnlineSta (Port1 Online).

```
// Start sequence when Trigger changes to TRUE
IF ((Trigger=TRUE) AND (DoCIPTrigger=FALSE) AND (_EIP_EtnOnlineSta=TRUE))THEN
    DoCIPTrigger            :=TRUE;
    Stage                   :=INT#1;
    CIPOpen_instance(Execute:=FALSE);                    // Initialize instance
    CIPWrite_instance(
        Execute             :=FALSE,                     // Initialize instance
        SrcDat              :=WriteDat);                 // Dummy
    CIPRead_instance(                                    // Initialize instance
        Execute             :=FALSE,                     // Dummy
        DstDat              :=ReadDat);                  // Dummy
    CIPSend_instance(
        Execute             :=FALSE,                     // Initialize instance
        ServiceDat          := Dummy,                    // Dummy
        RespServiceDat      :=RespDat);                  // Dummy
    CIPClose_instance(Execute:=FALSE);                   // Initialize instance
END_IF;

IF (DoCIPTrigger=TRUE) THEN
    CASE Stage OF
    1 :                                          // Open CIP Class 3 Connection (
Large_Forward_Open)
        CIPOpen_instance(
            Execute         :=TRUE,
            TimeOut         :=UINT#20,                   // Timeout time: 2.0 s
            RoutePath       :='02\192.168.250.2');       // Route path

        IF (CIPOpen_instance.Done=TRUE) THEN
            Stage           :=INT#2;                     // Normal end
        ELSIF (CIPOpen_instance.Error=TRUE) THEN
            Stage           :=INT#10;                    // Error end
        END_IF;

    2 :                                                  // Request writing value
 of variable
        CIPWrite_instance(
            Execute         :=TRUE,
            Handle          :=CIPOpen_instance.Handle,   // Handle
            DstDat          :='WritingDat',              // Destination variable
name
            Size            :=UINT#1,                    // Number of elements to
 write
            SrcDat          :=WriteDat);                 // Write data

        IF (CIPWrite_instance.Done=TRUE) THEN
```

```
            Stage           :=INT#3;                    // Normal end
        ELSIF (CIPWrite_instance.Error=TRUE) THEN
            Stage           :=INT#20;                   // Error end
        END_IF;

    3 :                                                 // Request reading value
of variable
        CIPRead_instance(
            Execute         :=TRUE,
            Handle          :=CIPOpen_instance.Handle,  // Handle
            SrcDat          :='OriginalDat',            // Source variable name
            Size            :=UINT#1,                   // Number of elements to
read
            DstDat          :=ReadDat);                 // Read data

        IF (CIPRead_instance.Done=TRUE) THEN
            Stage           :=INT#4;                    // Normal end
        ELSIF (CIPRead_instance.Error=TRUE) THEN
            Stage           :=INT#30;                   // Error end
        END_IF;

    4 :                                                 // Send message
        ReqPath.ClassID     :=UINT#01;
        ReqPath.InstanceID  :=UINT#01;
        ReqPath.isAttributeID:=TRUE;
        ReqPath.AttributeID :=UINT#07;
        CIPSend_instance(
            Execute         :=TRUE,
            Handle          :=CIPOpen_instance.Handle,  // Handle
            ServiceCode     :=BYTE#16#0E,               // Service code
            RqPath          :=ReqPath,                  // Request path
            ServiceDat      :=Dummy,                    // Service data
            Size            :=UINT#0,                   // Number of elements
            RespServiceDat  :=RespDat);                 // Response data

        IF (CIPSend_instance.Done=TRUE) THEN
            Stage           :=INT#5;                    // Normal end
        ELSIF (CIPSend_instance.Error=TRUE) THEN
            Stage           :=INT#40;                   // Error end
        END_IF;

    5 :                                                 // Request closing CIP c
lass 3 connection
        CIPClose_instance(
            Execute         :=TRUE,
            Handle          :=CIPOpen_instance.Handle); // Handle
```

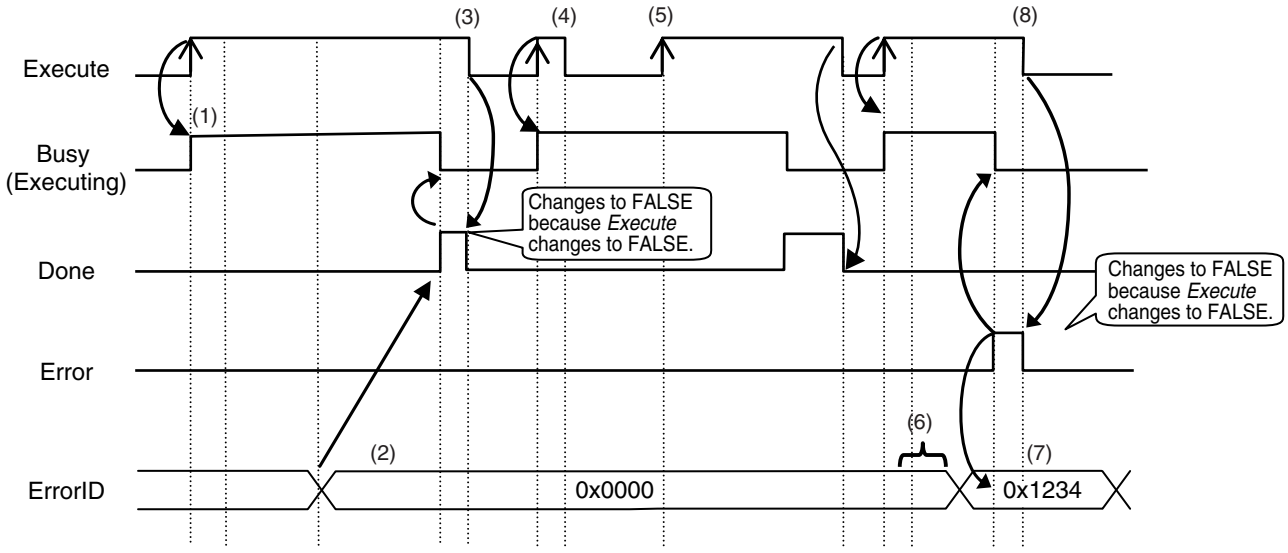
```
        IF (CIPClose_instance.Done=TRUE) THEN
            Stage            :=INT#0;
        ELSIF (CIPClose_instance.Error=TRUE) THEN
            Stage            :=INT#50;
        END_IF;

    0:                                                  // Processing after norm
al end
        DoCIPTrigger         :=FALSE;
        Trigger              :=FALSE;

    ELSE                                                // Processing after erro
r end
        DoCIPTrigger         :=FALSE;
        Trigger              :=FALSE;
    END_CASE;
END_IF;
```

7-2-9 Operation Timing

Output Variable Operation and Timing

You can monitor the values of the output variables to determine the status throughout instruction execution.

The following timing chart shows the operation of the output variables.



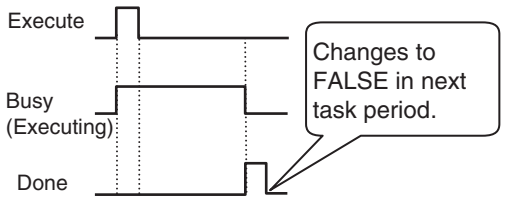
1. When *Execute* changes to TRUE, the instruction is executed and *Busy* changes to TRUE.
2. After the results of instruction execution are stored in the output variables, *Done* changes to TRUE and *Busy* changes to FALSE.
3. When *Execute* changes to FALSE, *Done* returns to FALSE.

4. When *Execute* changes to TRUE again, *Busy* changes to TRUE.
5. *Execute* is ignored if it changes to TRUE during instruction execution (i.e., when *Busy* is TRUE).
6. If an error occurs, several retries are attempted internally. The error code in *ErrorID* is not updated during the retries.
7. When a communications error occurs, *Error* changes to TRUE and the value of *ErrorID* is stored. Also, *Busy* and *Done* change to FALSE.
8. When *Execute* changes to FALSE, *Error* changes to FALSE.

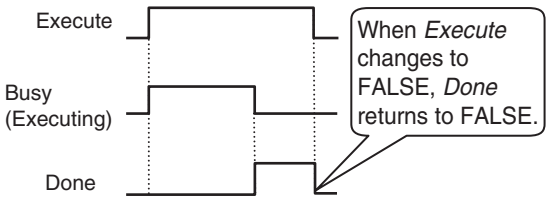
Precautions for Correct Use

If *Execute* changes back to FALSE before *Done* changes to TRUE, *Done* stays TRUE for only one task period. (Example 1)
If you want to see if *Done* is TRUE at any time, make sure to keep *Execute* TRUE until you confirm that *Done* is TRUE.
If *Execute* is TRUE until *Done* changes to TRUE, *Done* stays TRUE until *Execute* changes to FALSE. (Example 2)

Example 1



Example 2

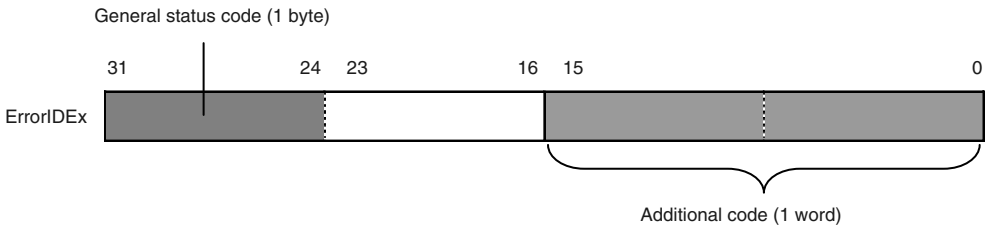


7-2-10 Response Codes

This section describes the response codes stored in the *ErrorIDEx* output variable if an error occurs during execution of a CIP message communications instruction.

General Status Codes

As response codes, general codes are stored in the *ErrorIDEx* output variable (DWORD data) after execution of a CIP communications instruction is completed.
If an additional code is added, the additional code is also stored.



General status code (hex)	Status name	Description of status
00	Success	Service was successfully performed by the object specified.
01	Connection failure	A connection related to service failed along the connection path.

General status code (hex)	Status name	Description of status
02	Resource unavailable	Resources needed for the object to perform the requested service were unavailable.
03	Invalid parameter value	See Status Code 20 hex.
04	Path segment error	The path segment identifier or the segment syntax was not understood by the processing node. Path processing stops when a path segment error occurs.
05	Path destination unknown	The path is referencing an object class, instance, or structure element that is not known or is not contained in the processing node. Path processing stops when a Path Destination Unknown Error occurs.
06	Partial transfer	Only part of the expected data was transferred.
07	Connection lost	The message connection was lost.
08	Service not supported	The requested service was not supported or was not defined for this object class/instance.
09	Invalid attribute value	Invalid attribute data was detected.
0A	Attribute list error	An attribute in the Get_Attribute_List or Set_Attribute_List response has a non-zero status.
0B	Already in requested mode/state	The object is already in the mode/state being requested by the service.
0C	Object state conflict	The object cannot perform the requested service in its current mode/state.
0D	Object already exists	The requested instance of object to be created already exists.
0E	Attribute not settable	A request to modify a non-modifiable attribute was received.
0F	Privilege violation	A permission/privilege check failed.
10	Device state conflict	The device's current mode/state prohibits the execution of the requested service.
11	Reply data too large	The data to be transmitted in the response buffer is larger than the allocated response buffer.
12	Fragmentation of a primitive value	The service specified an operation that is going to fragment a primitive data value, i.e. half a REAL data type.
13	Not enough data	The requested service did not supply enough data to perform the specified operation.
14	Attribute not supported	The attribute specified in the request is not supported.
15	Too much data	The service supplied more data than was expected.
16	Object does not exist	An object that does not exist was specified for the requested service.
17	Service fragmentation sequence not in progress	The fragmentation sequence for this service is not currently active for this data.
18	No stored attribute data	The attribute data of this object was not saved prior to the requested service.
19	Store operation failure	The attribute data of this object was not saved due to a failure during the attempt.
1A	Routing failure (request packet too large)	The service request packet was too large for transmission on a network in the path to the destination. The routing device was forced to abort the service.
1B	Routing failure (response packet too large)	The service response packet was too large for transmission on a network in the path from the destination. The routing device was forced to abort the service.

General status code (hex)	Status name	Description of status
1C	Missing attribute list entry data	The service did not supply an attribute in a list of attributes that was needed by the service to perform the requested behavior.
1D	Invalid attribute value list	The service is returning the list of attributes supplied with status information for those attributes that were invalid.
1E	Embedded service error	An embedded service resulted in an error.
1F	Vendor specific error	A vendor-specific error occurred. The Additional Code Field of the error response defines the error. This is a general error code that is used only for errors that do not correspond to any of the error codes in this table and are not in an object class definition.
20	Invalid parameter	A parameter for the requested service is invalid. This code is used when a parameter does not meet the requirements of the specification and/or the requirements defined in an application object specification.
21	Write-once value or medium already written	An attempt was made to write to a write-once medium (e.g. WORM drive or PROM) that was previously written or cannot be changed.
22	Invalid Reply Received	An invalid reply was received. (For example, the reply service code does not match the request service code. Or, the reply message is shorter than the minimum expected reply size.) This status code is used for other causes of invalid replies.
23-24		Reserved by CIP for future extensions.
25	Key Failure in path	The key segment that was included as the first segment in the path does not match the destination module. The object specific status must indicate which part of the key check failed.
26	Path Size Invalid	The size of the path that was sent with the service request is either too large or too small for the request to be routed to an object.
27	Unexpected attribute in list	An attempt was made to set an attribute that is not able to be set at this time.
28	Invalid Member ID	The member ID specified in the request does not exist in the specified class, instance, and attribute.
29	Member not settable	A request to modify a non-modifiable member was received.
2A	Group 2 only server general failure	This error code is reported only by group 2 only servers with 4K or less of code space and only in place of *Service not supported, Attribute not supported, or Attribute not settable*.
2B-CF		Reserved by CIP for future extensions.
D0-FF	Reserved for Object Class and service errors	This range of error codes is to be used to indicate object class-specific errors. This code range is used only when none of the error codes in this table accurately reflect the error that occurred. The additional code field is used to describe the general error code in more detail.

● Examples of Additional Status When General Status Is 01 Hex (Status of Connection Manager Object)

General Status (hex)	Additional Status (hex)	Description
01	0100	Connection in use or duplicate forward open.
01	0103	Transport class and trigger combination not supported.
01	0106	Ownership conflict.

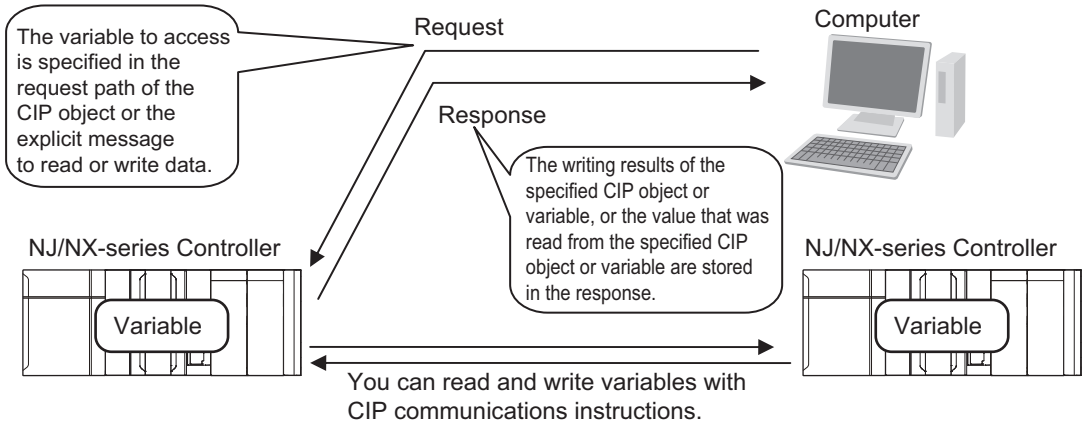
General Status (hex)	Additional Status (hex)	Description
01	0107	Connection not found at target application.
01	0108	Invalid connection type. There is a problem with either the connection type or priority of the connection.
01	0109	Invalid connection size.
01	0110	Device not configured.
01	0111	RPI not supported. May also indicate problem with connection time-out multiplier, or production inhibit time.
01	0113	Connection Manager cannot support any more connections.
01	0114	Either the vendor ID or the product code in the key segment does not match the device.
01	0115	Device type in the key segment does not match the device.
01	0116	*Major Revision* or *Minor Revision* in the key segment.
01	0117	Invalid connection point.
01	0118	Invalid configuration format.
01	0119	Connection request failed because there is no controlling connection currently open.
01	011A	Target application cannot support any more connections.
01	011B	RPI is smaller than the production inhibit time.
01	0203	Connection cannot be closed because the connection has timed out.
01	0204	Unconnected_Send service timed out while waiting for a response.
01	0205	Parameter error in Unconnected_Send service.
01	0206	Message too large for unconnected message service.
01	0207	Unconnected acknowledgment without reply.
01	0301	No buffer memory available.
01	0302	Network bandwidth not available for data.
01	0303	No tag filters available.
01	0304	Not configured to send real-time data.
01	0311	Port that was specified in port segment is not available.
01	0312	Link address that was specified in port segment is not available.
01	0315	Invalid segment type or segment value in path.
01	0316	Path and connection were not equal when closing the connection.
01	0317	The segment is not present. Or, the encoded value in the network segment is invalid.
01	0318	Link address to self is invalid.
01	0319	Resources on secondary are unavailable.
01	031A	Connection is already established.
01	031B	Direct connection is already established.
01	031C	Others
01	031D	Redundant connection mismatch.
01	031E	There are no more reception resources available on the sending module.
01	031F	No connection resources exist for the target path.
01	0320-07FF	Vendor specific.

7-3 CIP Communication Server Function

The CIP Communication Server function is exclusively available for the NJ/NX-series Controllers. This function executes services for a specified self-contained object in the CPU Unit after receiving the CIP messages from external devices.

This section provides information on CIP messages structure along with information about how to use CIP messages in a program that runs on a computer or by other means and uses the CIP Communication Server function to perform the following: -Writing CIP objects and the values of variables to the NJ/NX-series Controller, -Reading CIP objects and the values of variables from the NJ/NX-series Controller.

To read and write CIP objects or the values of variables between NJ/NX-series Controllers, use the CIP communications instructions.
Refer to *7-2 CIP Message Communications Client Function* on page 7-4 for information on how to use CIP communications instructions for CIP message communications.



Precautions for Correct Use

- To allow the Controller to receive CIP messages, select the **Use** Option for the CIP message server of the built-in EtherNet/IP port. If the **Do not use** Option for the CIP message server is selected, the Controller cannot receive CIP messages. For the details on the settings, refer to *CIP Message Server* on page 4-19.
- If the **Use** Option is selected for Packet Filter of the built-in EtherNet/IP port, make sure to permit packets to be used for CIP messages. If they are not permitted, the CIP message cannot be received. For the details on the settings, refer to *Packet Filter* on page 4-7.
- If the **Do not use** Option for the CIP message server is selected, EtherNet/IP communications cannot be used. This causes the following restrictions on the functionality of connected devices, tools, and Controllers.

Category	Restrictions
Connected device	• The programmable terminal NS-series cannot be connected.
Tools	• Sysmac Studio cannot go online through *Remote connection via USB*. • Tag data link setting using Sysmac Studio is not possible. • CX-Compolet and SYSMAC Gateway cannot be connected. • CNC Operator cannot be connected. • Network Configurator cannot be connected. Or, devices cannot be displayed. • CX-Configurator FDT (communication DTM OMRON EtherNet/IP) cannot be connected. • Sysmac Conrtoller Log Upload Tool cannot be connected to the Controller through *Remote connection via USB*.
Controller features	• The tag data link function cannot be used. • CIP Safety communications cannot be used in a configuration where an NX-SL5□□□ Unit is connected to an NX102 CPU Unit. • The server function for CIP messages (UCMM, Class 3) in the built-in EtherNet/IP port cannot be used.

Additional Information

- Selecting the **Do not use** Option for the CIP message server closes the TCP/UDP ports used for EtherNet/IP communications. This improves security of communications over the network.
- Even if the **Do not use** Option for the CIP message server is selected, the TCP/UDP message services can be used. You can also use the client function (CIP communications instructions) of CIP message communications.

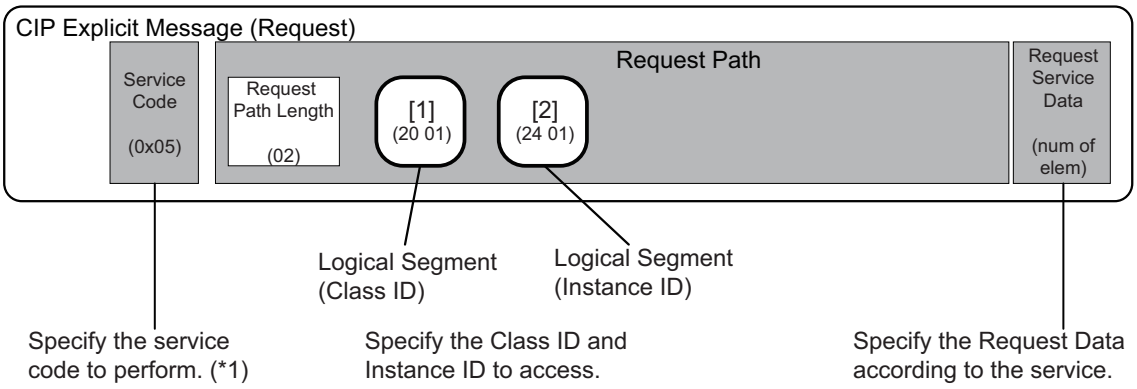
Version Information

The CIP message server settings can be used with the following unit versions of the CPU unit.
- NJ-series, NX102, NX1P2 CPU Unit: Version 1.49 or later
- NX701 CPU Unit: Version 1.29 or later

7-3-1 CIP Message Structure for Accessing CIP Objects

This section shows how to specify messages to access CIP objects.
The CIP objects to be accessed are expressed by connecting the segments
defined in the CIP Common Specifications in the request path field in a CIP explicit message.

Example: Performing the Reset service (0x05) to the Instance (01 hex) of the Identity object (class: 01 hex)



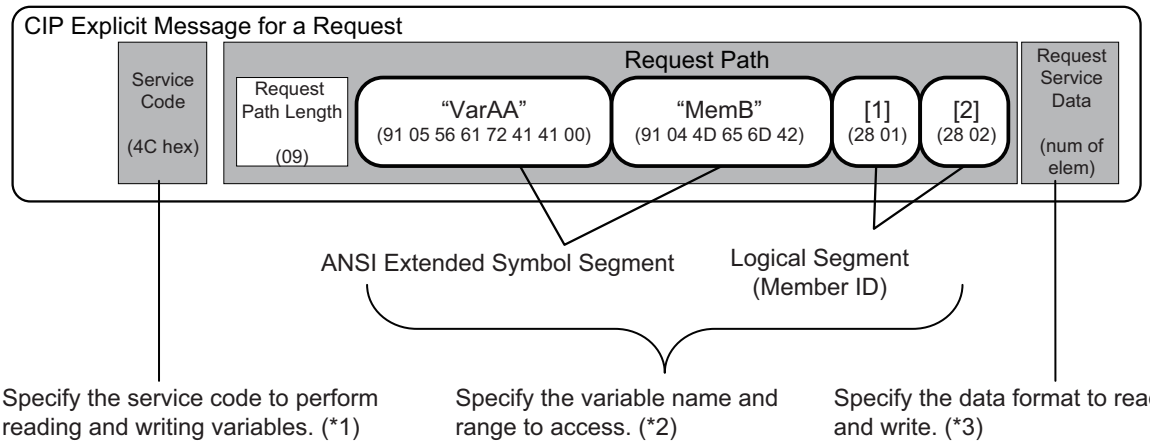
*1. Refer to *7-5 CIP Object Services* on page 7-48 for information about the service codes.

7-3-2 CIP Message Structure for Accessing Variables

This section shows how to specify messages to access variables.
The variables to access are given by connecting the segments that are defined in the CIP Common specifications so that explicit message can be set in the request path field.

The following elements are combined to make the specification.
Specifying the variable to access: The elements are stored in the CIP segments and then joined to make the message.

Example: Reading the Present Value of One Member of the VarAA.MemB[1.2] Structure Variable
Example for Using the CIP Read Data Service for a Variable Object



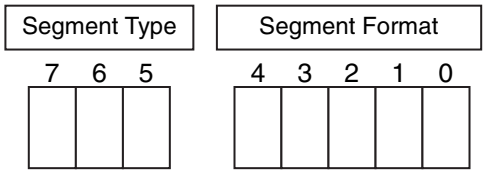
*1. Refer to *7-6 Read and Write Services for Variables* on page 7-84 for information about the service codes.
*2. Refer to *7-4-5 Specifying Variable Names in Request Paths* on page 7-44 for information about how to specify variables names.
*3. Refer to *7-7 Variable Data Types* on page 7-88 for details about how to specify data formats.

7-4 Specifying Request Path

The CIP object, variable name, structure member name, and array index are specified for the request path.
In CIP, the EPATH data type is used for the request path.
With this method, the request path is divided into segments and a value is assigned to each segment.
The request path notation shows the path to the final destination when the data segments are joined together.
Each segment includes the segment type information and the segment data.

Segment 1	Segment 2	Segment 3	Segment 4	· · · ·

The first byte gives the interpretation method for the segment. It consists of two parts; a 3-bit segment type and a 5-bit segment format.



The segment type specifications are defined as follows in the CIP specifications.

Segment Type			Meaning
7	6	5	
0	0	0	Port Segment
0	0	1	Logical Segment
0	1	0	Network Segment
0	1	1	Symbolic Segment
1	0	0	Data Segment
1	0	1	Data Type
1	1	0	Data Type
1	1	1	Reserved

The specifications for the segment format are different for each segment type. Use the segment format to request a service from a particular object of a particular device.
Logical segments and data segments, which are needed to specify variables in CIP message communications, are described below.

7-4-1 Examples of CIP Object Specifications

Logical Segments are joined to form the request path that specifies the object to access.

Logical Segment (Class ID)	Logical Segment (Instance ID)	Logical Segment (Attribute ID)
Specify the Class ID.	Specify the Instance ID.	Specify the Attribute ID.

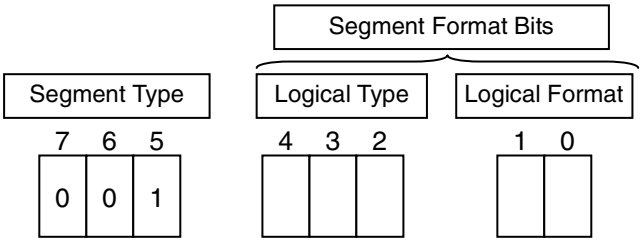
7-4-2 Examples of Variable Specifications

Segments are joined to form the request path that specifies the variable to access.

Data Segment (ANSI Extended Symbol Segment)	Logical Segment (Member ID)
Specify the variable name and the member name.	Specify the array index.

7-4-3 Logical Segment

A logical segment is used to give the range of the CIP Object or variable (array) in the request path.



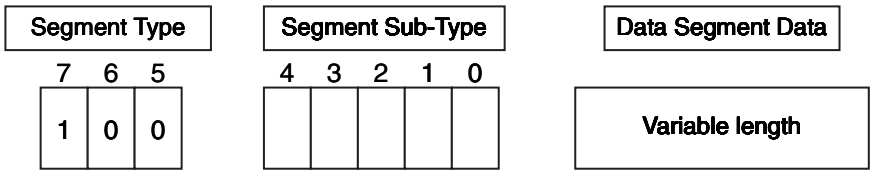
Logical Type			Meaning
4	3	2	
0	0	0	Class ID
0	0	1	Instance ID
0	1	0	Member ID
0	1	1	Connection Point
1	0	0	Attribute ID
1	0	1	Special (Do not use the logical addressing definition for the Logical Format.)
1	1	0	Service ID (Do not use the logical addressing definition for the Logical Format.)
1	1	1	Reserved

Logical Format		Meaning
1	0	
0	0	8 bit logical address
0	1	16 bit logical address
1	0	32 bit logical address
1	1	Reserved

An 8-bit or 16-bit logical address can be used for the class ID and attribute ID.
An 8-bit,16-bit, or 32-bit logical address can be used for the instance ID.

7-4-4 Data Segment

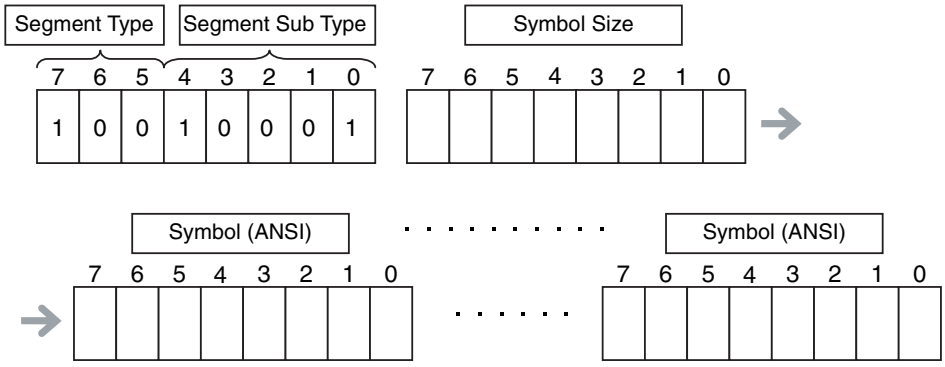
A data segment is used to give the specified variable name in the request path.



Segment Sub-Type					Meaning
4	3	2	1	0	
0	0	0	0	0	Simple Data Segment
1	0	0	0	1	ANSI Extended Symbol Segment

A data segment is mainly used for an ANSI extended symbol segment.
This segment sub-type is used to read and write the values of variables.

ANSI Extended Symbol Segment



7-4-5 Specifying Variable Names in Request Paths

Variable Names

A variable name is specified as a symbolic segment (ANSI extended symbol segment).
Variable Name Specification Format

BYTE	91 hex	ANSI Extended Symbol Segment
BYTE	Length in BYTE	Length of variable name in bytes
Array of octet	: Variable_name :	Variable name encoded in UTF-8
Octet	(pad)	00 hex. One byte is padded if the variable name length is an odd number of bytes.

Variable Names
Variable names are encoded in UTF-8.

Structure Member Names

Structure member names are specified in the same way as variable names.
Store UTF-8 character codes in the ANSI extended symbol segment.

Array Indices

Specify the array index in a logical segment that is set as a member ID.
You can specify an array index ([x]) in a variable name.

(Specification Method 1: 8-bit Index)

BYTE	28 hex	Logical Segment (Member ID)
USINT	Index	Array index from 0 to 255

(Specification Method 2: 16-bit Index)

BYTE	29 hex		Logical Segment (Member ID)
octet	00 hex		Pad
UINT	Index	(L)	Array index from 0 to 65,535
		(H)	

Range Specifications with the Num of Element Field

There is a Num of Element field in the request data for the variable read and variable write services.
You can use these services to access the specified range of an array with the following specifications.

- Specify the first element in the range of elements to access in the array variable as the variable to read or write.
- Specify the number of elements to access in the Num of Element field.

Specification Examples

This example shows how to specify VarAA.MemB[1.2] for the following structure variable.

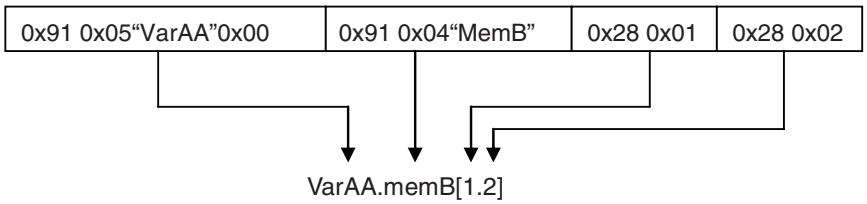
```
struct
{
    UINT    MemA;
    BOOL    MemB[10][10];
} VarAA;
```

Variable Name Specification Format

BYTE	91 hex	ANSI Extended Symbol Segment
BYTE	05 hex	Length of variable name in bytes
Array of octet	'V'	Variable name
	'a'	
	'r'	
	'A'	
	'A'	
Octet	00 hex	Pad
BYTE	91 hex	ANSI Extended Symbol Segment
BYTE	04 hex	Length of variable name in bytes
Array of octet	'M'	Variable name
	'e'	
	'm'	
	'B'	
BYTE	28 hex	Logical Segment (Member ID)
USINT	01 hex	Array index for the first element
BYTE	28 hex	Logical Segment (Member ID)
USINT	02 hex	Array index for the second element

The variable name that is specified in the symbolic segment (ANSI extended symbol segment) must be converted to a text string to pass it to the communications thread. The following conversion rules apply.

Specification Example for Structure Members and Array Elements



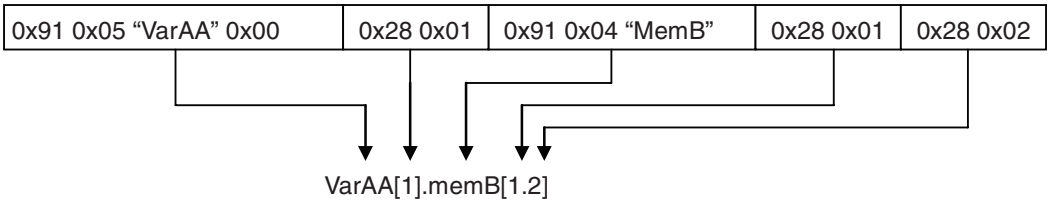
This example shows how to specify VarAA[1].MemB[1.2] for the following structure variable.

```
struct
{
    UINT    MemA;
    BOOL    MemB[10][10];
} VarAA[3]
```

Variable Name Specification Format

BYTE	91 hex	ANSI Extended Symbol Segmer
BYTE	05 hex	Length of variable name in byte
Array of octet	'V'	Variable name
	'a'	
	'r'	
	'A'	
	'A'	
Octet	00 hex	Pad
BYTE	28 hex	Logical Segment (Member ID)
USINT	01 hex	Array index
BYTE	91 hex	ANSI Extended Symbol Segmer
BYTE	04 hex	Length of variable name in byte
Array of octet	'M'	Variable name
	'e'	
	'm'	
	'B'	
BYTE	28 hex	Logical Segment (Member ID)
USINT	01 hex	Array index for the first element
BYTE	28 hex	Logical Segment (Member ID)
USINT	02 hex	Array index for the second elem

Specification Example for Structure Array



7-5 CIP Object Services

This section shows services that specify the CIP object in the Request Path and access the CIP message server function of the NJ/NX-series Controllers.

7-5-1 CIP Objects Sent to the Built-in EtherNet/IP Port

The following CIP objects can be sent to an EtherNet/IP port.

Object name	Function	Reference
Identity object	• Reads ID information from the CPU Unit. • Resets the built-in EtherNet/IP port.	page 7-48
NX Configuration object	• Reads and Writes NX object. • Restarts the NX Unit and initializes the Unit operation settings. • Saves the parameters of the NX Unit and switches the write mode. • Obtains the current errors of the Controller and NX Unit, and obtains and clears an event log. • Obtains the user-defined errors of the Controller.	page 7-52
TCP/IP Interface object	• Writes and reads TCP/IP settings.	page 7-73
Ethernet link object	• Reads Ethernet settings. • Reads Ethernet status.	page 7-76
Controller object	• Gets the Controller status. • Changes the operating mode of the Controller.	page 7-82

7-5-2 Identity Object (Class ID: 01 hex)

This object reads the ID information of the CPU Unit and resets the built-in EtherNet/IP port.
When using an NX701 CPU Unit or an NX102 CPU Unit, use the route path to specify the port number (1 or 2) of the built-in EtherNet/IP port to access.

Service Codes

Specify the service to execute with the service code.

Service code	Parameter name	Description	Supported services	
			Classes	Instances
01 hex	Get_Attribute_All	Reads the values of the attributes.	Supported	Supported
0E hex	Get_Attribute_Single	Reads the value of the specified attribute.	Supported	Supported

Service code	Parameter name	Description	Supported services	
			Classes	Instances
05 hex	Reset	Resets the built-in EtherNet/IP port. This parameter is used to reset the built-in EtherNet/IP port when you change the IP address or other parameter settings and want to apply them. Input one of the following values for the *ServiceDat* input variable to the CIPSend instruction to specify the reset method. 00 hex: Resets the built-in EtherNet/IP port. 02 hex*1: Clears the saved tag data link settings and resets the built-in EtherNet/IP port.	Not supported	Supported

*1. The value is 01 hex for a CPU Unit with unit version 1.09 or earlier.

Class ID

Specify 01 hex.

Instance ID

Specify 00 or 01 hex.

Attribute ID

The attribute ID specifies the information to read.

● Class Attribute ID

The class attribute ID specifies the attribute of the entire object.

Attribute ID	Parameter name	Description	Attribute	Read data	
				Data type	Value
01 hex	Revision	Revision of the object	Read	UINT	0001 hex
02 hex	Max Instance	The maximum instance number	Read	UINT	0001 hex

● Instance Attribute ID

The instance attribute ID specifies the attribute of the instance.

Attribute ID	Parameter name	Description	Attribute	Read data	
				Data type	Value
01 hex	Vendor ID	Vendor ID	Read	UINT	002F hex
02 hex	Device Type	Device type	Read	UINT	000C hex
03 hex	Product Code	Product code	Read	UINT	Refer to (1) Product Codes for Each Model, below.

Attribute ID	Parameter name	Description	Attribute	Read data	
				Data type	Value
04 hex	Revision	Device revision	Read	Struct	---
	Major Revision	Major revision	Read	USINT	Refer to (2) Major and Minor CIP Revisions, below.
	Minor Revision	Minor revision	Read	USINT	
05 hex	Status	Status of the built-in EtherNet/IP port	Read	WORD	Refer to (3) Status Details of the Built-in EtherNet/IP Port, below.
06 hex	Serial Number	Serial number	Read	UDINT	Set value
07 hex	Product Name	Product name	Read	STRING	Set value

1. Product Codes for Each Model

Model	Product Code
NX701-□□□□	067D hex
NX102-1200	0BBB hex
NX102-1100	0BBA hex
NX102-1000	0BB9 hex
NX102-9000	0BB8 hex
NX102-1220	0BBF hex
NX102-1120	0BBE hex
NX102-1020	0BBD hex
NX102-9020	0BBC hex
NX1P2-□□□□	068B hex
NJ501-13□□	0665 hex
NJ501-14□□	0666 hex
NJ501-15□□	0667 hex
NJ501-43□□	066E hex
NJ501-44□□	066F hex
NJ501-45□□	0670 hex
NJ501-5300	068C hex
NJ501-R300	069C hex
NJ501-R400	069D hex
NJ501-R500	069E hex
NJ501-R320	069F hex
NJ501-R420	06A0 hex
NJ501-R520	06A1 hex
NJ301-11□□	066B hex
NJ301-12□□	066C hex
NJ101-□□□□	0680 hex

2. Major and Minor CIP Revisions

Unit version	CIP revisions	
	Major revision	Minor revision
Unit version 1.00	01 hex	01 hex
Unit version 1.01 or 1.02		03 hex

Unit version	CIP revisions	
	Major revision	Minor revision
Unit version 1.03 to 1.08	02 hex	01 hex
Unit version 1.09		02 hex
Unit version 1.10		03 hex
Unit version 1.11 or 1.12		04 hex
Unit version 1.13 to 1.20		05 hex
Unit version 1.21 or later		06 hex

3. Status Details of the Built-in EtherNet/IP Port

Bit	Name	Description
0	Owned	Indicates when the built-in EtherNet/IP port has an open connection as the target of a tag data link.
1	Reserved	Always FALSE.
2	Configured	Tag data link settings exist.
3	Reserved	Always FALSE.
4 to 7	Extended Device Status	Indicates the status of the built-in EtherNet/IP port.*1
8	Minor Recoverable Fault	TRUE when any of the following errors occurs. • IP Rout Table Setting Error • DNS Server Connection Failed • Tag Data Link Setting Error • Tag Data Link Timeout • Tag Data Link Connection Timeout • FTP Server Setting Error • NTP Client Setting Error • SNMP Setting Error • NTP Server Connection Failed • Tag Name Resolution Error
9	Minor Unrecoverable Fault	TRUE when the following error occurs. • Identity Error
10	Major Recoverable Fault	TRUE when any of the following errors occurs. • IP Address Duplication Error • BOOTP Server Connection Error • Basic Ethernet Setting Error • IP Address Setting Error
11	Major Unrecoverable Fault	TRUE when any of the following errors occurs. • Communications Controller Error • MAC Address Error
12 to 15	Reserved	Always FALSE.

*1. Bits 7 to 4 indicate the status of the built-in EtherNet/IP port.

b7	b6	b5	b4	
0	1	0	1	A major fault occurred.
0	0	1	0	A timeout occurred in one or more target connections.
0	0	1	1	Indicates that there are no tag data link settings.
0	1	1	0	Indicates that one or more connections are performing communications normally.
0	1	1	1	Other than the above.

Request Paths (IOIs) to Specify Objects

When you specify an object, specify the request path (IOI) for each service code as given below.

Service code		Class ID	Instance ID	Attribute ID
01 hex	Get_Attribute_All	01 hex	• Specifying a service for a class : 00 hex • Specifying a service for an instance : Always 01 hex	Not required.
0E hex	Get_Attribute_Single			• Reading a class attribute : 01 or 02 hex • Reading an instance attribute : 01 to 07 hex
05 hex	Reset		Always 01 hex	Not required.

7-5-3 NX Configuration Object (Class ID: 74 hex)

This object is used to control the NX Unit such as reading and writing an NX object, restarting the NX Unit, obtaining an event log and current errors, and clearing. This can only be used for the NX102 CPU Units.

Service Codes

Specify the service to execute with the service code.

Service Code	Parameter name	Description	Supported services		Reference
			Classes	Instances	
33 hex	Read NX object	Reads the NX object of the specified NX Unit.	Not supported.	Supported.	page 7-53
34 hex	Write NX object	Writes the NX object of the specified NX Unit.	Not supported.	Supported.	page 7-54
35 hex	Restart NX unit	Restarts the specified NX Units.	Not supported.	Supported.	page 7-56
36 hex	Save parameter	Saves the parameters of the specified NX Unit.	Not supported.	Supported.	page 7-57
37 hex	Switch parameter write mode	Switches the parameter write mode of the specified NX Units.	Not supported.	Supported.	page 7-58
38 hex	Read total power on time	Reads the total power on time of the specified NX Unit.	Not supported.	Supported.	page 7-59
3A hex	Get current error	Obtains the current errors of the Controller or specified NX Unit.	Not supported.	Supported.	page 7-60
3B hex	Get event log	Obtains the event log of the Controller or specified NX Unit.	Not supported.	Supported.	page 7-64

Service Code	Parameter name	Description	Supported services		Reference
			Classes	Instances	
3C hex	Clear event log	Clears the event log of the Controller or specified NX Unit.	Not supported.	Supported.	page 7-68
3D hex	Initialize unit operation parameter	Initializes the Unit operation settings (NX object) of the specified NX Unit.	Not supported.	Supported.	page 7-69
3E hex	Get current user error	Obtains the user-defined errors of the Controller.	Not supported.	Supported.	page 7-70

Class ID

Specify 74 hex.

Instance ID

Specify 01 hex.

Read NX object (Service Code: 33 hex)

Read the NX object of the specified NX Unit.

● Request Data Format

Parameter name	Data type	Description
Service	USINT	Read NX object service: 33 hex
Request Path Size	USINT	Size of Request Path: 02 hex
Request Path	Padded EPATH	Request path: 2074 2401 hex Class ID: 74 hex Instance ID: 01 hex
Unit No	UINT	Unit number 0001 to 0020 hex: NX Unit 0000, 0021 hex or above: Not supported
Index	UINT	NX object index
Sub index	USINT	NX object sub index
Control Field	USINT	Complete access specification 00 hex: Not specified

● Response Data Format

When the processing is successful

Parameter name	Data type	Description
Reply Service	USINT	Read NX object service response: B3 hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Code indicating normal: 00 hex

Parameter name	Data type	Description
Size of Additional Status	USINT	Size of Additional Status: 00 hex
Length	UINT	Read data size (Byte)
Read data	Depends on data type	Read data

When the processing failed

Parameter name	Data type	Description
Reply Service	USINT	Read NX object service response: B3 hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Current error code defined by CIP
Size of Additional Status	USINT	Size of Additional Status: 00 hex or 01 hex*1
Additional Status	UINT	Additional Status*2

*1. When the General Status of the response code is 1F hex (Vendor specific error), becomes 01 hex.
*2. This is stored only when the Size of Additional Status is 01 hex.

● CIP Error Codes

General status code (hex)	Error name	Cause
02	Resource unavailable	Resources needed for the object to perform the requested service were unavailable.
0C	Object state conflict	The object cannot perform the requested service in its current mode/state.
10	Device state conflict	The state of the NX object is not in a state to execute the required service.
11	Reply data too large	Data larger than the maximum response data length was read.
13	Not enough data	Data required for the execution of the required service is insufficient.
15	Too much data	Extra data for the execution of the required service is included.
1F	Vendor specific error	The service could not be executed due to internal reasons.
20	Invalid parameter	• The Unit number is out of the supported range. • The object of the index specified for the NX object does not exist. • The Index specified for the NX object exists but the Sub Index does not exist.

Write NX Object (Service Code: 34 hex)

Write the NX object of the specified NX Unit.

● Request Data Format

Parameter name	Data type	Description
Service	USINT	Write NX object service: 34 hex
Request Path Size	USINT	Size of Request Path: 02 hex

Parameter name	Data type	Description
Request Path	Padded EPATH	Request path: 2074 2401 hex Class ID: 74 hex Instance ID: 01 hex
Unit No	UINT	Unit number 0001 to 0020 hex: NX Unit 0000, 0021 hex or above: Not supported
Index	UINT	NX object index
Sub index	USINT	NX object sub index
Control Field	USINT	Complete access specification 00 hex: Not specified
Length	UINT	Write data size (Byte)
Write Data	Depends on data type	Write data

● Response Data Format

When the processing is successful

Parameter name	Data type	Description
Reply Service	USINT	Write NX object service response: B4 hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Code indicating normal: 00 hex
Size of Additional Status	USINT	Size of Additional Status: 00 hex

When the processing failed

Parameter name	Data type	Description
Reply Service	USINT	Write NX object service response: B4 hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Current error code defined by CIP
Size of Additional Status	USINT	Size of Additional Status: 00 hex or 01 hex*1
Additional Status	UINT	Additional Status*2

*1. When the General Status of the response code is 1F hex (Vendor specific error), becomes 01 hex.

*2. This is stored only when the Size of Additional Status is 01 hex.

● CIP Error Codes

General status code (hex)	Error name	Cause
02	Resource unavailable	Resources needed for the object to perform the requested service were unavailable.
0C	Object state conflict	The object cannot perform the requested service in its current mode/state.
0E	Attribute not settable	The NX object which is not modifiable is specified.
10	Device state conflict	• Carried out writing in a state that was not the parameter write mode. • The state of the NX object is not in a state to execute the required service.
13	Not enough data	Data required for the execution of the required service is insufficient.

General status code (hex)	Error name	Cause
15	Too much data	Extra data for the execution of the required service is included.
1F	Vendor specific error	The service could not be executed due to internal reasons.
20	Invalid parameter	• The Unit number is out of the supported range. • The sizes of the specified object and Length do not match. • The object of the index specified for the NX object does not exist. • The Index specified for the NX object exists, but the Sub Index does not exist. • Write data is out of the range.

Restart NX Unit (Service Code: 35 hex)

Restart the specified NX Units.

● Request Data Format

Parameter name	Data type	Description
Service	USINT	Restart NX Unit service: 35 hex
Request Path Size	USINT	Size of Request Path: 02 hex
Request Path	Padded EPATH	Request path: 2074 2401 hex Class ID: 74 hex Instance ID: 01 hex
Unit No	UINT	Unit number 0000 hex: All NX Units 0001 to 0020 hex: NX Unit 0021 hex or above: Not supported

● Response Data Format

When the processing is successful

Parameter name	Data type	Description
Reply Service	USINT	Restart NX Unit service response: B5 hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Code indicating normal: 00 hex
Size of Additional Status	USINT	Size of Additional Status: 00 hex

When the processing failed

Parameter name	Data type	Description
Reply Service	USINT	Restart NX Unit service response: B5 hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Current error code defined by CIP*1
Size of Additional Status	USINT	Size of Additional Status: 00 hex or 01 hex*2
Additional Status	UINT	Additional Status*3

*1. When the request is made to an NX Unit that does not support the Restart NX Unit service, error codes are returned. (General status: 1F hex, Additional status: 2601 hex)

*2. When the General Status of the response code is 1F hex (Vendor specific error), becomes 01 hex.
*3. This is stored only when the Size of Additional Status is 01 hex.

● CIP Error Codes

General status code (hex)	Error name	Cause
02	Resource unavailable	Resources needed for the object to perform the requested service were unavailable.
0C	Object state conflict	The object cannot perform the requested service in its current mode/state.
10	Device state conflict	The target unit is not in a state to execute the required service.
13	Not enough data	Data required for the execution of the required service is insufficient.
15	Too much data	Extra data for the execution of the required service is included.
1F	Vendor specific error	The service could not be executed due to internal reasons.
20	Invalid parameter	• The Unit number is out of the supported range. • The Unit does not exist.

Save Parameter (Service Code: 36 hex)

Save the parameters of the specified NX Unit.
The saved parameter is valid after the NX Unit is restarted.

● Request Data Format

Parameter name	Data type	Description
Service	USINT	Save parameter service: 36 hex
Request Path Size	USINT	Size of Request Path: 02 hex
Request Path	Padded EPATH	Request path: 2074 2401 hex Class ID: 74 hex Instance ID: 01 hex
Unit No	UINT	Unit number 0001 to 0020 hex: NX Unit 0000, 0021 hex or above: Not supported

● Response Data Format

When the processing is successful

Parameter name	Data type	Description
Reply Service	USINT	Save parameter service response: B6 hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Code indicating normal: 00 hex
Size of Additional Status	USINT	Size of Additional Status: 00 hex

When the processing failed

Parameter name	Data type	Description
Reply Service	USINT	Save parameter service response: B6 hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Current error code defined by CIP
Size of Additional Status	USINT	Size of Additional Status: 00 hex or 01 hex*1
Additional Status	UINT	Additional Status*2

*1. When the General Status of the response code is 1F hex (Vendor specific error), becomes 01 hex.
*2. This is stored only when the Size of Additional Status is 01 hex.

● CIP Error Codes

General status code (hex)	Error name	Cause
02	Resource unavailable	Resources needed for the object to perform the requested service were unavailable.
0C	Object state conflict	The object cannot perform the requested service in its current mode/state.
13	Not enough data	Data required for the execution of the required service is insufficient.
15	Too much data	Extra data for the execution of the required service is included.
19	Store operation failure	The parameters could not be saved due to internal reasons.
1F	Vendor specific error	The service could not be executed due to internal reasons.
20	Invalid parameter	• The Unit number is out of the supported range. • The Unit does not exist.

Switch Parameter Write Mode (Service Code: 37 hex)

Switch the parameter write mode of the specified NX Units.

● Request Data Format

Parameter name	Data type	Description
Service	USINT	Switch parameter write mode service: 37 hex
Request Path Size	USINT	Size of Request Path: 02 hex
Request Path	Padded EPATH	Request path: 2074 2401 hex Class ID: 74 hex Instance ID: 01 hex
Unit No	UINT	Unit number 0000 hex: All NX Units 0001 to 0020 hex: NX Unit 0021 hex or above: Not supported

● Response Data Format

When the processing is successful

Parameter name	Data type	Description
Reply Service	USINT	Switch parameter write mode service response: B7 hex

Parameter name	Data type	Description
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Code indicating normal: 00 hex
Size of Additional Status	USINT	Size of Additional Status: 00 hex

When the processing failed

Parameter name	Data type	Description
Reply Service	USINT	Switch parameter write mode service response: B7 hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Current error code defined by CIP
Size of Additional Status	USINT	Size of Additional Status: 00 hex or 01 hex*1
Additional Status	UINT	Additional Status*2

*1. When the General Status of the response code is 1F hex (Vendor specific error), becomes 01 hex.
*2. This is stored only when the Size of Additional Status is 01 hex.

● CIP Error Codes

General status code (hex)	Error name	Cause
02	Resource unavailable	Resources needed for the object to perform the requested service were unavailable.
0C	Object state conflict	The object cannot perform the requested service in its current mode/state.
10	Device state conflict	This service could not change because the transition to the parameter write mode is in progress.
13	Not enough data	Data required for the execution of the required service is insufficient.
15	Too much data	Extra data for the execution of the required service is included.
1F	Vendor specific error	The service could not be executed due to internal reasons.
20	Invalid parameter	• The Unit number is out of the supported range. • The Unit does not exist.

Read Total Power On Time (Service Code: 38 hex)

Read the total power on time of the specified NX Unit.

● Request Data Format

Parameter name	Data type	Description
Service	USINT	Read total power on time service: 38 hex
Request Path Size	USINT	Size of Request Path: 02 hex
Request Path	Padded EPATH	Request path: 2074 2401 hex Class ID: 74 hex Instance ID: 01 hex

Parameter name	Data type	Description
Unit No	UINT	Unit number 0001 to 0020 hex: NX Unit 0000, 0021 hex or above: Not supported

● Response Data Format

When the processing is successful

Parameter name	Data type	Description
Reply Service	USINT	Read total power on time service response: B8 hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Code indicating normal: 00 hex
Size of Additional Status	USINT	Size of Additional Status: 00 hex
Total power on time	ULINT	Total power on time of NX Units.

When the processing failed

Parameter name	Data type	Description
Reply Service	USINT	Read total power on time service response: B8 hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Current error code defined by CIP
Size of Additional Status	USINT	Size of Additional Status: 00 hex or 01 hex*1
Additional Status	UINT	Additional Status*2

*1. When the General Status of the response code is 1F hex (Vendor specific error), becomes 01 hex.
*2. This is stored only when the Size of Additional Status is 01 hex.

● CIP Error Codes

General status code (hex)	Error name	Cause
02	Resource unavailable	Resources needed for the object to perform the requested service were unavailable.
0C	Object state conflict	The object cannot perform the requested service in its current mode/state.
13	Not enough data	Data required for the execution of the required service is insufficient.
15	Too much data	Extra data for the execution of the required service is included.
1F	Vendor specific error	The service could not be executed due to internal reasons.
20	Invalid parameter	• The Unit number is out of the supported range. • The Unit does not exist.

Get Current Error (Service Code: 3A hex)

Obtain the current errors of the Controller or specified NX Unit.

● Request Data Format

Parameter name	Data type	Description
Service	USINT	Get current error service: 3A hex
Request Path Size	USINT	Size of Request Path: 02 hex
Request Path	Padded EPATH	Request path: 2074 2401 hex Class ID: 74 hex Instance ID: 01 hex
Unit No	UINT	Unit number 0000 hex: Controller 0001 to 0020 hex: NX Unit 0021 hex or above: Not supported
Start number of read record	UINT	Top number of read record
Number of read record	UINT	Number of read records Controller (0 to 5) NX Units (0 to 9) When the registered number of records is smaller than the number of read records, an error does not occur, and all the registered event codes are read.

● Response Data Format

When the processing is successful

Parameter name	Data type	Description
Reply Service	USINT	Get current error service response: BA hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Code indicating normal: 00 hex
Size of Additional Status	USINT	Size of Additional Status: 00 hex
Error update count	UINT	Total count value of errors
Record size	UINT	Size of one record (Byte) Controller error: 0060 hex NX Unit error: 0032 hex
Number of registered record	UINT	Number of registered records
Number of readout record	UINT	Number of readout records
Current error record[0] to Current error record[8]	Array of struct Current error record	Current error array Stores data for the "Number of readout record" from index 0 of the current error record. The remaining elements of the current error record array are not included in the response data. Example: When the "Number of readout record" is 3 and the response data includes the current error record array [0-2], the current error record array [3-8] is not included in the response data. For details of the specifications of the structure, refer to *Current Error Record Structure* on page 7-62.

When the processing failed

Parameter name	Data type	Description
Reply Service	USINT	Get current error service response: BA hex
Reserved	USINT	Reserved: 00 hex

Parameter name	Data type	Description
General Status	USINT	Current error code defined by CIP
Size of Additional Status	USINT	Size of Additional Status: 00 hex or 01 hex*1
Additional Status	UINT	Additional Status*2

*1. When the General Status of the response code is 1F hex (Vendor specific error), becomes 01 hex.
*2. This is stored only when the Size of Additional Status is 01 hex.

● Current Error Record Structure

The format of the current error record information differs between the Controller and NX Unit.

Controller

Parameter name	Data type	Description
Index	UDINT	Current error index number This number is assigned when system event logs and access event logs are registered.
Event occurred time	ULINT	Error occurred time
Event source	UINT	Error source
Event priority	UINT	Error level
Event code	UDINT	Event code
Code system	UINT	Code system
Event source details	UINT	Error source details
Reserved	UDINT	Reserved
Vendor code	UDINT	Vendor code
Device type code	UDINT	Device type code
Product code	UDINT	Product code of the Unit in which errors occurred
Additional information[0] to Additional information[31]	Array of BYTE	Attached information (system information) of event.
Reserved[0] to Reserved[23]	Array of BYTE	Reserved

NX Unit

Parameter name	Data type	Description
Index	UDINT	Current error index number This number is assigned when system event logs and access event logs are registered.
Unit number	USINT	Unit number 0000 hex: Controller 0001 to 0020 hex: NX Unit
Event priority	USINT	Error level
Event occurred time	UDINT	Error occurred time
Product code	UDINT	Product code of the Unit in which errors occurred
Event code	UDINT	Event code
Additional information[0] to Additional information[31]	Array of BYTE	Attached information (system information) of event.

● CIP Error Codes

General status code (hex)	Error name	Cause
02	Resource unavailable	Resources needed for the object to perform the requested service were unavailable.
0C	Object state conflict	The object cannot perform the requested service in its current mode/state.
13	Not enough data	Data required for the execution of the required service is insufficient.
15	Too much data	Extra data for the execution of the required service is included.
1F	Vendor specific error	The service could not be executed due to internal reasons.
20	Invalid parameter	• The Unit number is out of the supported range. • The specification of the number of readout records is out of the range. • The Unit does not exist.

● Method of Use

1 The following variables are generated and initialized to 0.
- Total number of readout records (UINT)
- Previous error update count (UINT)

2 Specify the following parameters and execute Get current error (3A hex).
- Unit No: Unit number subject to error information read
- Start number of read record: 0
- Number of read record: Number of read records

3 The following parameters are read from the response data.
- Error update count
- Number of registered record
- Number of readout record
- Current error record

When the first response is obtained, the value of Error update count is retained as the previous error update count.
When the second response onwards is obtained, the previous error update count and the Error update count are compared. If the value is updated with any additional current errors of the Unit, execute this operation from step1 again.

4 Add the Number of readout record value of the response data to the total number of readout records.

5 If the total number of readout records does not reach the Number of registered record, it means that some records have not been read yet. Specify the following parameters and execute Get current error again.
- Start number of read record: Start number of read record when the previous service was executed + Number of readout record of response.

- Number of read record: Number of read records

Repeat steps (3) to (5) until the total number of readout records matches the Number of registered record.

Get Event Log (Service Code: 3B hex)

Obtain the event log of the Controller or specified NX Unit.
When the Controller is specified, the event log saved in the Controller is obtained. Event logs of slaves connected to the Controller such as EtherCAT slaves cannot be obtained.

● Request Data Format

Parameter name	Data type	Description
Service	USINT	Get event log service: 3B hex
Request Path Size	USINT	Size of Request Path: 02 hex
Request Path	Padded EPATH	Request path: 2074 2401 hex Class ID: 74 hex Instance ID: 01 hex
Unit No	UINT	Unit number 0000 hex: Controller 0001 to 0020 hex: NX Unit 0021 hex or above: Not supported
Event log type	UINT	Event log type 0000 hex: System event log 0001 hex: Access event log 0002 hex: User event log
Start index of read record	UDINT	Top index number of read record If the record specified by the Start index of read record is not found in the Unit, the record will be read from the oldest index. If the maximum number of event log records which can be registered for the Unit is exceeded, this will occur since old records are overwritten by new records.
Number of read record	UINT	Number of read records Controller (0 to 5) NX Units (0 to 9) When the registered number of records is smaller than the number of read records, an error does not occur, and all the registered event logs are read.

● Response Data Format

When the processing is successful

Parameter name	Data type	Description
Reply Service	USINT	Get event log service response: BB hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Code indicating normal: 00 hex
Size of Additional Status	USINT	Size of Additional Status: 00 hex

Parameter name	Data type	Description
Record size	UINT	Size of one record (Byte) Controller event log: 0060 hex NX Unit event log: 0032 hex
Number of registered record	UINT	Number of registered records
Latest index of registered record	UDINT	Index number of the latest registered record
Last index of readout record	UDINT	Index number of last readout record
Number of readout record	UINT	Number of readout records
Reserved	UINT	Reserved
Event log record[0] to Event log record[8]	Array of struct Event Log Record	Event log array Stores data for the "Number of readout record" from index 0 of the event log record. The remaining elements of the event log record array are not included in the response data. Example: When the "Number of readout record" is 3 and the response data includes the event log record array [0-2], the event log record array [3-8] is not included in the response data. For details of the specifications of the structure, refer to *Event Log Record Structure* on page 7-65.

When the processing failed

Parameter name	Data type	Description
Reply Service	USINT	Get event log service response: BB hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Current error code defined by CIP
Size of Additional Status	USINT	Size of Additional Status: 00 hex or 01 hex*1
Additional Status	UINT	Additional Status*2

*1. When the General Status of the response code is 1F hex (Vendor specific error), becomes 01 hex.
*2. This is stored only when the Size of Additional Status is 01 hex.

● Event Log Record Structure

The format of the event log record information differs between the Controller and NX Unit.

Controller system event log and access event log

Parameter name	Data type	Description
Index	UDINT	Event log index number This number is assigned when system event logs and access event logs are registered.
Event occurred time	ULINT	Event occurred time
Event source	UINT	Event source
Event priority	UINT	Event level
Event code	UDINT	Event code
Code system	UINT	Code system
Event source details	UINT	Event source details
Reserved	UDINT	Reserved
Vendor code	UDINT	Vendor code
Device type code	UDINT	Device type code

Parameter name	Data type	Description
Product code	UDINT	Product code of the Unit in which event occurred
Additional information[0] to Additional information[31]	Array of BYTE	Attached information (system information) of event.
Reserved[0] to Reserved[23]	Array of BYTE	Reserved

Controller user event log

Parameter name	Data type	Description
Index	UDINT	Event log index number This number is assigned when system event logs and access event logs are registered.
Event occurred time	ULINT	Event occurred time
Event source	UINT	Event source
Event priority	UINT	Event level
Event code	UDINT	Event code
Event priority details	UINT	Event level details
Additional information[0] to Additional information[39]	Array of BYTE	Attached information (system information) of event.
Reserved[0] to Reserved[31]	Array of BYTE	Reserved

NX Unit

Parameter name	Data type	Description
Index	UDINT	Event log index number This number is assigned when system event logs and access event logs are registered.
Unit number	USINT	Unit number 0000 hex: Controller 0001 to 0020 hex: NX Unit
Event priority	USINT	Event level
Event occurred time	UDINT	Event occurred time
Product code	UDINT	Product code of the Unit in which event occurred
Event code	UDINT	Event code
Additional information[0] to Additional information[31]	Array of BYTE	Attached information (system information) of event.

● CIP Error Codes

General status code (hex)	Error name	Cause
02	Resource unavailable	Resources needed for the object to perform the requested service were unavailable.

General status code (hex)	Error name	Cause
0C	Object state conflict	The object cannot perform the requested service in its current mode/state.
13	Not enough data	Data required for the execution of the required service is insufficient.
15	Too much data	Extra data for the execution of the required service is included.
1F	Vendor specific error	The service could not be executed due to internal reasons.
20	Invalid parameter	• The Unit number is out of the supported range. • The specification of the number of readout records is out of the range. • The Unit does not exist.

● Method of Use

1 The following variables are generated and initialized to 0.
- Total number of readout records (UINT)
- Record index during the previous readout (UDINT)
- Previous latest record index (UDINT)

2 Specify the following parameters and execute Get event log(3B hex).
- Unit No: Unit number subject to event information readout
- Start number of read record: 0
- Number of read record: Number of read records

3 The following parameters are read from the response data.
- Number of registered record
- Latest index of registered record
- Last index of readout record
- Number of readout record
- Event log record

When the first response is obtained, the value of Latest index of registered record value is retained as the record index during the previous readout.
When the second response onwards is obtained, the record index during the previous readout and Latest index of registered record value are compared. If the value is updated with any additional event logs of the Unit, execute this operation from step1 again.

4 Add the Number of readout record value of the response data to the total number of readout records.

5 If the total number of readout records does not reach the Number of registered record, it means that some records have not been read yet. Specify the following parameters and execute Get event log again.
- Start number of read record: Last index of readout record when the previous service was executed + 1.
- Number of read record: Number of read records

Repeat steps (3) to (5) until the total number of readout records matches the Number of registered record.

Clear Event Log (Service Code: 3C hex)

Clear the event log of the Controller or specified NX Unit.
The event log is immediately cleared after the service is successful. When it is executed for the Controller, only the event log saved in the Controller is cleared. Event logs of slaves connected to the Controller such as EtherCAT slaves are not cleared.

● Request Data Format

Parameter name	Data type	Description
Service	USINT	Clear event log service: 3C hex
Request Path Size	USINT	Size of Request Path: 02 hex
Request Path	Padded EPATH	Request path: 2074 2401 hex Class ID: 74 hex Instance ID: 01 hex
Unit No	UINT	Unit number 0000 hex: Controller 0001 to 0020 hex: NX Unit 0021 hex or above: Not supported
Event log type	UINT	Event log type 0000 hex: System event log 0001 hex: Access event log 0002 hex: User event log*1 0003 hex: All types of the system event log, access event log, user event log.

*1. The User event log is valid only when the Controller is specified for the Unit number.

● Response Data Format

When the processing is successful

Parameter name	Data type	Description
Reply Service	USINT	Clear event log service response: BC hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Code indicating normal: 00 hex
Size of Additional Status	USINT	Size of Additional Status: 00 hex

When the processing failed

Parameter name	Data type	Description
Reply Service	USINT	Clear event log service response: BC hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Current error code defined by CIP
Size of Additional Status	USINT	Size of Additional Status: 00 hex or 01 hex*1
Additional Status	UINT	Additional Status*2

*1. When the General Status of the response code is 1F hex (Vendor specific error), becomes 01 hex.
*2. This is stored only when the Size of Additional Status is 01 hex.

● CIP Error Codes

General status code (hex)	Error name	Cause
02	Resource unavailable	Resources needed for the object to perform the requested service were unavailable.
0C	Object state conflict	The object cannot perform the requested service in its current mode/state.
13	Not enough data	Data required for the execution of the required service is insufficient.
15	Too much data	Extra data for the execution of the required service is included.
1F	Vendor specific error	The service could not be executed due to internal reasons.
20	Invalid parameter	• The Unit number is out of the supported range. • The Unit does not exist.

Initialize Unit Operation Parameter (Service Code: 3D hex)

Initializes the Unit operation settings (NX object) of the specified NX Unit.
The initialized parameters are valid after the NX Unit is restarted.
By executing this service, NX Unit Memory All Cleared (95810000 hex) is registered in the event log.
When the NX Unit is Operational or Safe-Operational, you need to initialize the status beforehand with the Switch parameter write mode service. If the Initialize unit operation parameter is executed without carrying out this step, error will result, and Device state conflict (10 hex) will be returned to the General Status.
This service does not support the NX-series Safety Control Unit. If this service is executed for the NXseries Safety Control Unit, an error will occur.

● Request Data Format

Parameter name	Data type	Description
Service	USINT	Initialize unit operation parameter service: 3D hex
Request Path Size	USINT	Size of Request Path: 02 hex
Request Path	Padded EPATH	Request path: 2074 2401 hex Class ID: 74 hex Instance ID: 01 hex
Unit No	UINT	Unit number 0001 to 0020 hex: NX Unit 0000 hex, 0021 hex or above: Not supported

● Response Data Format

When the processing is successful

Parameter name	Data type	Description
Reply Service	USINT	Initialize unit operation parameter service response: BD hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Code indicating normal: 00 hex
Size of Additional Status	USINT	Size of Additional Status: 00 hex

When the processing failed

Parameter name	Data type	Description
Reply Service	USINT	Initialize unit operation parameter service response: BD hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Current error code defined by CIP
Size of Additional Status	USINT	Size of Additional Status: 00 hex or 01 hex*1
Additional Status	UINT	Additional Status*2

*1. When the General Status of the response code is 1F hex (Vendor specific error), becomes 01 hex.
*2. This is stored only when the Size of Additional Status is 01 hex.

● CIP Error Codes

General status code (hex)	Error name	Cause
02	Resource unavailable	Resources needed for the object to perform the requested service were unavailable.
0C	Object state conflict	The object cannot perform the requested service in its current mode/state.
10	Device state conflict	The device state is not in a state to execute the required service.
13	Not enough data	Data required for the execution of the required service is insufficient.
15	Too much data	Extra data for the execution of the required service is included.
1F	Vendor specific error	The service could not be executed due to internal reasons.
20	Invalid parameter	• The Unit number is out of the supported range. • The Unit does not exist.

Get Current User Error (Service Code: 3E hex)

Obtain the user-defined errors of the Controller.

● Request Data Format

Parameter name	Data type	Description
Service	USINT	Get current user error service: 3E hex
Request Path Size	USINT	Size of Request Path: 02 hex
Request Path	Padded EPATH	Request path: 2074 2401 hex Class ID: 74 hex Instance ID: 01 hex
Unit No	UINT	Unit number 0000 hex: Controller 0001 hex or above: Not supported
Start number of read record	UINT	Top number of read record

Parameter name	Data type	Description
Number of read record	UINT	Number of read records (0 to 5) When the registered number of records is smaller than the number of read records, an error does not occur, and all the registered event logs are read.

● Response Data Format

When the processing is successful

Parameter name	Data type	Description
Reply Service	USINT	Get current user error service response: BE hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Code indicating normal: 00 hex
Size of Additional Status	USINT	Size of Additional Status: 00 hex
Error update count	UINT	Total count value of errors
Record size	UINT	Size of one record (Byte):0060 hex
Number of registered record	UINT	Number of registered records
Number of readout record	UINT	Number of readout records
User error record[0] to User error record[8]	Array of struct User error record	User-defined error array Stores data for the "Number of readout record" from index 0 of the User error record. The remaining elements of the User error record array are not included in the response data. Example: When the "Number of readout record" is 3 and the response data includes the User error record array [0-2], the User error record array [3-8] is not included in the response data. For details of the specifications of the structure, refer to User Error Record Structure.

When the processing failed

Parameter name	Data type	Description
Reply Service	USINT	Get current user error service response: BE hex
Reserved	USINT	Reserved: 00 hex
General Status	USINT	Current error code defined by CIP
Size of Additional Status	USINT	Size of Additional Status: 00 hex or 01 hex*1
Additional Status	UINT	Additional Status*2

*1. When the General Status of the response code is 1F hex (Vendor specific error), becomes 01 hex.

*2. This is stored only when the Size of Additional Status is 01 hex.

● User Error Record Structure

Parameter name	Data type	Description
Index	UDINT	User-defined error index number This number is assigned when system event logs and access event logs are registered.
Event occurred time	ULINT	Error occurred time
Event source	UINT	Error source
Event priority	UINT	Error level
Event code	UDINT	Event code

Parameter name	Data type	Description
Event priority details	UINT	Error level details
Reserved	UINT	Reserved
Additional information[0] to Additional information[39]	Array of BYTE	Attached information (system information) of event.
Reserved[0] to Reserved[31]	Array of BYTE	Reserved

● CIP Error Codes

General status code (hex)	Error name	Cause
02	Resource unavailable	Resources needed for the object to perform the requested service were unavailable.
0C	Object state conflict	The object cannot perform the requested service in its current mode/state.
13	Not enough data	Data required for the execution of the required service is insufficient.
15	Too much data	Extra data for the execution of the required service is included.
1F	Vendor specific error	The service could not be executed due to internal reasons.
20	Invalid parameter	• The Unit number is out of the supported range. • The Unit does not exist.

● Method of Use

1 The following variables are generated and initialized to 0.
- Total number of readout records (UINT)
- Previous error update count (UINT)

2 Specify the following parameters and execute Get current user error (3E hex).
- Unit No: Unit number subject to error information readout
- Start number of read record: 0
- Number of read record: Number of read records

3 The following parameters are read from the response data.
- Error update count
- Number of registered record
- Number of readout record
- User error record

When the first response is obtained, the value of Error update count is retained as the previous error update count.
When the second response onwards is obtained, the previous error update count and the Error update count are compared. If the value is updated with any additional user-defined errors of the Unit, execute this operation from step1 again.

4 Add the Number of readout record value of the response data to the total number of readout records.

5 If the total number of readout records does not reach the Number of registered record, it means that some records have not been read yet. Specify the following parameters and execute Get current error again.
- Start number of read record: Start number of read record when the previous service was executed + Number of readout record of response.
- Number of read record: Number of read records

Repeat steps (3) to (5) until the total number of readout records matches the Number of registered record.

7-5-4 TCP/IP Interface Object (Class ID: F5 hex)

This object is used to read and write settings such as the IP address, subnet mask, and default gateway.
For NX701 and NX102 CPU Units, it is necessary to use the route path of the CIP communications instruction (the *RoutePath* input variable) to specify the port number (1 or 2) of the built-in EtherNet/IP port to access.

Service Codes

Specify the service to execute with the service code.

Service code	Parameter name	Description	Supported services	
			Classes	Instances
01 hex	Get_Attribute_All	Reads the values of the attributes.	Supported	Not supported
0E hex	Get_Attribute_Single	Reads the value of the specified attribute.	Supported	Supported
10 hex	Set_Attribute_Single	Writes a value to the specified attribute. The built-in EtherNet/IP port restarts automatically after the value is written to the attribute. When the next Set_Attribute_Single is executed before the restart process is completed, the general status "0C hex" (Object State Conflict) is returned.	Not supported	Supported

Class ID

Specify F5 hex.

Instance ID

Specify 00 or 01 hex.
00: Specify the class
01: Built-in EtherNet/IP Port

Attribute ID

The attribute ID specifies the information to read.

● Class Attribute ID

The class attribute ID specifies the attribute of the entire object.

Attribute ID	Parameter name	Description	Attribute	Read data	
				Data type	Value
01 hex	Revision	Revision of the object	Read	UINT	0001 hex: Unit version 1.01 or earlier 0002 hex: Unit version 1.02 to 1.09 0003 hex: Unit version 1.10 0004 hex: Unit version 1.11 or later
02 hex	Max Instance	The maximum instance number	Read	UINT	0001 hex
03 hex	Number of Instances	The number of object instances	Read	UINT	0001 hex

● Instance Attribute ID

The instance attribute ID specifies the attribute of the instance.

Attribute ID	Parameter name	Description	Attribute	Read/write data	
				Data type	Value
01 hex	Interface Configuration Status	Indicates the IP address setting status of the interface.	Read	DWORD	Bits 0 to 3: Interface Configuration Status: 0 = IP address is not set. (This includes when BOOTP is starting.) 1 = IP address is set. Bits 4 and 5: Reserved (always FALSE). bit6: AcdStatus[*1]: FALSE = IP address collisions have not been detected. TRUE = IP address collisions have been detected. Bits 7 to 31: Reserved (always FALSE).

Attribute ID	Parameter name	Description	Attribute	Read/write data	
				Data type	Value
02 hex	Configuration Capability	Indicates a Controller Configurations and Setup that can be set to the interface.	Read	DWORD	Bit 0: BOOTP Client: Always TRUE. Bit 1: DNS Client: Always TRUE. Bit 2: DHCP Client: Always FALSE. Bit 3: DHCP-DNS Update: Always FALSE. Bit 4: Configuration Settable: Always TRUE. Bit 5: Hardware Configurable: Always FALSE. Bit 6: Interface Configuration Change Requires Reset: Always FALSE. bit7: ACD Capable*1: Always TRUE. Bits 8 to 31: Reserved (always FALSE).
03 hex	Configuration Control	Sets the method used to set the IP address when the interface starts.	Read/ Write	DWORD	Bit 0 to 3: IP Address Setting Method 0 = Setting the static IP address. 1 = Setting by BOOTP. Bit 4: DNS Enable/Disable Setting FALSE = DNS disabled. TRUE = DNS enabled. Bits 5 to 31: Reserved (always FALSE).
04 hex	Physical Link Object	The path to the link object in the physical layer.	Read	Struct	---
	Path size	The path size (WORD size).		UINT	0002 hex
	Path	The path to the link object in the physical layer (static).		EPATH	20 F6 24 01 hex
05 hex	Interface Configuration	The interface settings.	Read/ Write	Struct	---
	IP Address	IP address.		UDINT	Set value
	Network Mask	Subnet mask.		UDINT	Set value
	Gateway Address	The default gateway.		UDINT	Set value
	Name Server	The primary name server.		UDINT	Set value
	Name Server2	The secondary name server.		UDINT	Set value
	Domain Name	The domain name.		STRING	Set value*2
06 hex	Host Name	The host name (reserved).	Read/ Write	STRING	Set value*3

*1. The value is always FALSE for a CPU Unit with unit version 1.01 or earlier.
*2. The value is the size of domain name (2 bytes) + domain name (48 bytes max.).
*3. The value is the size of host name (2 bytes) + host name (64 bytes max.).

Request Paths (IOIs) to Specify Objects

When you specify an object, specify the request path (IOI) for each service code as given below.

Service code		Class ID	Instance ID	Attribute ID
01 hex	Get_Attribute_All	F5 hex	• Specifying a service for a class: 00 hex • Specifying a service for an instance: 01 hex	Not required.
0E hex	Get_Attribute_Single			• Reading a class attribute: 01 or 03 hex • Reading and writing an instance attribute: 01 to 06 hex
10 hex	Set_Attribute_Single			

7-5-5 Ethernet Link Object (Class ID: F6 hex)

This object is used to set and read Ethernet communications and read Ethernet communications status information.
For NX701 and NX102 CPU Units, it is necessary to use the route path of the CIP communications instruction (the *RoutePath* input variable) to specify the port number (1 or 2) of the built-in EtherNet/IP port to access.

Service Codes

Specify the service to execute with the service code.

Service code	Parameter name	Description	Supported service range	
			Class	Instance
0E hex	Get_Attribute_Single	Reads the value of the specified attribute.	Supported	Supported
10 hex	Set_Attribute_Single	Writes a value to the specified attribute.	Supported	Supported
4C hex	Get_and_Clear	Specify Attribute4 or Attribute5 to reset the value of the attribute to 0.	Not supported	Supported

Class ID

Specify F6 hex.

Instance ID

Specify 00 or 01 hex.
00: Specify the class
01: Built-in EtherNet/IP Port

Attribute ID

The attribute ID specifies the information to read.

● Class Attribute ID

The class attribute ID specifies the attribute of the entire object.

Attribute ID	Parameter name	Description	Attribute	Read data	
				Data type	Value
01 hex	Revision	Revision of the object	Read	UINT	0002 hex: Unit version 1.11 or earlier 0004 hex: Unit version 1.12 or later
02 hex	Max Instance	The maximum instance number	Read	UINT	0001 hex
03 hex	Number of Instances	The number of object instances	Read	UINT	0001 hex

● Instance Attribute ID

The instance attribute ID specifies the attribute of the instance.

Attribute ID	Parameter name	Description	Attribute	Read/write data	
				Data type	Value
01 hex	Interface Speed	Gives the baud rate for the interface.	Read	UDINT	Reads the current value.
02 hex	Interface Flags	Gives the status of the interface.	Read	DWORD	Refer to (1) Interface Flag Details, below.
03 hex	Physical Address	Gives the MAC address of the interface.	Read	ARRAY [0...5] OF USINT	Reads the current value of the MAC address.

Attribute ID	Parameter name	Description	Attribute	Read/write data	
				Data type	Value
04 hex	Interface Counters	The number of packets sent/received through the interface.	Read	Struct	---
	In Octets	The number of octets received through the interface. This includes unnecessary multicast packets and discarded packets counted by InDiscards.		UDINT	Reads the current value.
	In Unicast Packets	The number of unicast packets received through the interface. This does not include discarded packets counted by InDiscards.		UDINT	Reads the current value.
	In NonUnicast Packets	The number of packets besides unicast packets received through the interface. This includes unnecessary multicast packets, but does not include discarded packets counted by InDiscards.		UDINT	Reads the current value.
	In Discards	The number of discarded incoming packets received through the interface.		UDINT	Reads the current value.
	In Errors	The number of incoming packets that had errors. This is not included in InDiscards.		UDINT	Reads the current value.
	In Unknown Protos	The number of incoming packets that were of an unknown protocol.		UDINT	Reads the current value.
	Out Octets	The number of octets sent through the interface.		UDINT	Reads the current value.
	Out Unicast Packets	The number of unicast packets sent through the interface.		UDINT	Reads the current value.
	Out NonUnicast Packets	The number of packets besides unicast packets sent through the interface.		UDINT	Reads the current value.
	Out Discards	The number of discarded sent packets.		UDINT	Reads the current value.
	Out Errors	The number of sent packets that had errors.		UDINT	Reads the current value.

Attribute ID	Parameter name	Description	Attribute	Read/write data	
				Data type	Value
05 hex	Media Counters	Media counters for the communications port.	Read	Struct	---
	Alignment Errors	Number of frames received that were not octets in length.		UDINT	Reads the current value.
	FCS Errors	Number of frames received that did not pass the FCS check.		UDINT	Reads the current value.
	Single Collisions	Number of frames sent successfully with only one collision.		UDINT	Reads the current value.
	Multiple Collisions	Number of frames sent successfully with two or more collisions.		UDINT	Reads the current value.
	SQE Test Errors	Number of times a SQE test error message was generated.		UDINT	Reads the current value.
	Deferred Transmissions	The number of frames for which the first attempt to send was delayed because the media was busy.		UDINT	Reads the current value.
	Late Collisions	The number of collisions detected in packets that were sent after 512 bit times.		UDINT	Reads the current value.
	Excessive Collisions	The number of frames that failed to be sent because of excessive collisions.		UDINT	Reads the current value.
	MAC Transmit Errors	The number of frames that failed to be sent due to an internal MAC sublayer transmission error.		UDINT	Reads the current value.
	Carrier Sense Errors	The number of times the carrier sense condition was lost or the number of times an assertion did not occur when an attempt was made to send the frame.		UDINT	Reads the current value.
	Frame Too Long	The number of frames received that exceeded the maximum allowed frame size.		UDINT	Reads the current value.
	MAC Receive Errors	The number of frames that could not be received through the interface due to an internal MAC sublayer reception error.		UDINT	Reads the current value.

Attribute ID	Parameter name	Description	Attribute	Read/write data	
				Data type	Value
06 hex	Interface Control	Control settings for the interface.	Read/ Write	Struct	---
	Control Bits	Auto Nego for Ethernet communications that specifies full duplex.		WORD	Refer to (2) Control Bit Details, below.
	Forced Interface Speed	Gives the set value of the Ethernet baud rate.		UINT	Reads the set value.
0C hex *1	HC Interface Counters	The number of packets sent/ received through the HC interface.	Read	Struct	---
	HCInOctets	The number of octets received through the interface. This counter is the 64-bit edition of In Octets.		ULINT	Reads the current value.
	HCInUnicastPkts	The number of unicast packets received through the interface. This counter is the 64-bit edition of In Ucast Packets.		ULINT	Reads the current value.
	HCInMulticastPkts	The number of multicast packets received through the interface.		ULINT	Reads the current value.
	HCInBroadcastPkts	The number of broadcast packets received through the interface.		ULINT	Reads the current value.
	HCOutOctets	The number of octets sent through the interface.		ULINT	Reads the current value.
	HCOutUnicastPkts	The number of unicast packets sent through the interface. This counter is the 64-bit edition of Out Octets.		ULINT	Reads the current value.
	HCOutMulticastPkts	The number of multicast packets sent through the interface.		ULINT	Reads the current value.
	HCOutBroadcastPkts	The number of broadcast packets sent through the interface.		ULINT	Reads the current value.

Attribute ID	Parameter name	Description	Attribute	Read/write data	
				Data type	Value
0D hex*1	HC Media Counters	Media counters for the communications port.	Read	Struct	---
	HCStatsAlignmentErrors	The number of frames received that were not octets in length. This counter is the 64-bit edition of Alignment Errors.		ULINT	Reads the current value.
	HCStatsFCSErrors	The number of frames received that did not pass the FCS check. This counter is the 64-bit edition of FCS Errors.		ULINT	Reads the current value.
	HCStatsInternalMacTransmitErrors	The number of frames that failed to be sent due to an internal MAC sublayer transmission error. This counter is the 64-bit edition of MAC Transmit Errors.		ULINT	Reads the current value.
	HCStatsFrameTooLongs	The number of frames received that exceeded the maximum allowed frame size. This counter is the 64-bit edition of Frame Too Long.		ULINT	Reads the current value.
	HCStatsInternalMacReceiveErrors	The number of frames that could not be received through the interface due to an internal MAC sublayer reception error. This counter is the 64-bit edition of MAC Receive Errors.		ULINT	Reads the current value.
	HCStatsMacSymbolErrors	The number of frames that could not be received through the interface due to an internal MAC sublayer rsymbol error.		ULINT	Reads the current value.

*1. A CPU Unit with unit version 1.13 or later is required to use this attribute.

1. Interface Flag Details

Bit	Name	Description
0	LinkStatus	FALSE: The link is down. TRUE: The link is up.
1	Half/FullDuplex	FALSE: Half duplex TRUE: Full duplex
2 to 4	Negotiation Status	00 hex: Auto-negotiation is in progress. 01 hex: Auto-negotiation and speed detection failed. 02 hex: Auto-negotiation failed, but speed detection succeeded. 03 hex: Speed and duplex mode negotiation succeeded. 04 hex: Auto-negotiation was not attempted.
5	Manual Setting Requires Speed	Always FALSE: Changes can be applied automatically.
6	Local Hardware Fault	Always FALSE

Bit	Name	Description
7 to 31	Reserved	Always FALSE

2. Control Bit Details

Bit	Name	Description
0	Auto-negotiate	FALSE: Auto-negotiation is disabled. TRUE: Auto-negotiation is enabled.
1	ForcedDuplex Mode	FALSE: Half duplex TRUE: Full duplex*1
2 to 16	Reserved	Always FALSE

*1. When auto-negotiation is enabled (bit 0 is TRUE), this should always be FALSE.

Request Paths (IOIs) to Specify Objects

When you specify an object, specify the request path (IOI) for each service code as given below.

Service code		Class ID	Instance ID	Attribute ID
0E hex	Get_Attribute_Single	F6 hex	• Specifying a service for a class: 00 hex • Specifying a service for an instance: Always 01 hex	• Reading a class attribute: 01 to 03 hex • Reading and writing a instance attribute: 01 to 06 hex, 0C hex, and 0D hex
10 hex	Set_Attribute_Single			
4C hex	Get_and_Clear			Specify an attribute to clear the value to 0: 04 hex, 05 hex, 0C hex, 0D hex

7-5-6 Controller Object (Class ID: C4 hex)

This object is used to get the status of the Controller or to change the operating mode of the Controller.

Service Codes

Specify the service to execute with the service code.

Service code	Parameter name	Description	Supported service range	
			Class	Instance
0E hex	Get_Attribute_Single	Reads the value of the specified attribute.	Supported	Not supported
10 hex	Set_Attribute_Single	Writes a value to the specified attribute.	Supported	Not supported
51 hex	Reset_System_Alarm_All	Clears all errors of CPU Unit.	Supported	Not supported

Class ID

Specify C4 hex.

Instance ID

Specify 00 hex.

● Class Attribute ID

The class attribute ID specifies the attribute (value) of the entire object.

Attribute ID	Parameter name	Description	Attribute	Read/write data	
				Data type	Value
01 hex	Revision	Revision of the object	Read	UINT	Always 0002 hex.
02 hex	Max Instance	The maximum instance number	Read	UINT	Always 0001 hex.
64 hex	PLC Mode	This can be used to read and modify the Controller operating mode.	Read/ Write	UINT	Specify this when you want to write to an attribute. 0000 hex: PROGRAM mode 0004 hex: RUN mode
65 hex	PLC Error Status	Indicates when there is a Controller error. Changes to TRUE when a fatal or non-fatal error occurs.	Read	UINT	0000 hex: There is no Controller error. 0001 hex: There is a Controller error.
66 hex	PLC Model	Indicates the model of the Controller. The length is always 2 bytes for the size + 20 bytes for the name. Unused area is padded with spaces.	Read	STRING	

● Instance Attribute ID

None

Request Paths (IOIs) to Specify Objects

When you specify an object, specify the request path (IOI) for each service code as given below.

Service code		Class ID	Instance ID	Attribute ID
0E hex	Get_Attribute_Single	C4 hex	00 hex	Specifies the attribute of the class to read or write : 01 hex, 02 hex, or 64 to 66 hex
10 hex	Set_Attribute_Single			

7-6 Read and Write Services for Variables

This section shows services that specify the CIP object in the Request Path and access the CIP message server function of the NJ/NX-series Controllers.

7-6-1 Read Service for Variables

Specify service code 4C hex to read the value of the variable that is specified by the request path.

Service code: 4C hex

● Request Data Format

Request Path Data

Variable name specification

Request Service Data

+0	Num of Element	(L)	UINT	Specify the number of elements to read for an array variable.
+1		(H)		

Response Service Data

Data Type	USINT	Data type of variable to read
AddInfo Length	USINT	Additional information: Field length in bytes
(AddInfo)		Additional information: CRC value of structure
Actual data*		

*1. The actual data is stored in little-endian format.

Data Type	Code for data type of variable to read. Refer to *7-7-1 Data Type Codes* on page 7-88.
AddInfoLength	The size of the AddInfo area is stored only when accessing a structure variable. Set 02 hex for a structure variable. Otherwise, set 00 hex.
AddInfo	The CRC code of the structure definition is stored only when accessing a structure variable. In this case, the size of AddInfo will be 2 bytes.
Actual data	The actual data is stored in little-endian format. If 0001 hex is specified for an array, the actual data is stored in the same format as when you access a variable with the data type of the elements of the array.

Response Codes

CIP status	Meaning	Add status	Cause
00	SUCCESS	---	The service ended normally.
02	RESOURCE_UNAVAILABLE	---	The internal processing buffer is not available.
04	PATH_SEGMENT_ERROR	---	The request path specification is not correct.
05	PATH_DESTINATION_UNKNOWN	---	The variable specification is not correct.
0C	OBJECT_STATE_CONFLICT	8010	Downloading, starting up
		8011	There is an error in tag memory.

CIP status	Meaning	Add status	Cause
11	REPLY_DATA_TOO_LARGE	---	The response exceeds the maximum response length.
13	NOT_ENOUGH_DATA	---	The data length was too short for the specified service.
15	TOO_MUCH_DATA	---	The data length was too long for the specified service.
1F	VENDOR_SPECIFIC_ERROR	0102,2104	An attempt was made to read an I/O variable that cannot be read.
		0104,1103	The specified address and size exceed a segment boundary.
		8001	An internal error occurred.
		8007	An inaccessible variable was specified.
		8031	An internal error occurred. (A memory allocation error occurred.)
20	INVALID_PARAMETER	8009	A segment type error occurred.
		800F	There is an inconsistency in data length information in the request data
		8017	More than one element was specified for a variable that does not have elements.
		8018	Zero elements or data that exceeded the range of the array was specified for an array.
		8023	An internal error occurred. (An illegal command format was used.)
		8024	An internal error occurred. (An illegal command length was used.)
		8025	An internal error occurred. (An illegal parameter was used.)
		8027	An internal error occurred. (A parameter error occurred.)
		8028	• An attempt was made to write an out-of-range value for a variable for which a subrange is specified. • An attempt was made to write an undefined value to an enumeration variable.

7-6-2 Write Service for Variables

Specify service code 4D hex to write the value of the variable that is specified by the request path.

Request Data Format for Writing a Variable

Request Path Data

Variable name specification

Request Service Data

Data Type		USINT	Data type of variable to write
AddInfo Length		USINT	Additional information: Field length in bytes
(AddInfo)			Additional information: CRC value of structure
Num of Element	(L)	UINT	
	(H)		
Actual data*			

Response Service Data

There is no response service data.

*1. Data to write: Store the data to write in little-endian format.

Data Type	Code for data type of variable to write. Refer to *7-7 Variable Data Types* on page 7-88.
AddInfoLength	Specify the size of the AddInfo area only when accessing a structure variable. Set 02 hex for a structure variable. Otherwise, set 00 hex.
AddInfo	The CRC code of the structure definition is specified only when accessing a structure variable. In this case, the size of AddInfo will be 2 bytes.
NumOfElement	Specify the number of elements in the array. Do not specify 0000 hex (an error will occur). For variables other than arrays, set 0001 hex.
Actual data	Specify the actual data in little-endian format. If 0001 hex is specified for an array, specify the actual data in the same format as when you access a variable with the data type of the elements of the array.

Response Codes

CIP status	Meaning	Add status	Cause
00	SUCCESS	---	The service ended normally.
02	RESOURCE_UNAVAILABLE	---	The internal processing buffer is not available.
04	PATH_SEGMENT_ERROR	---	The request path specification is not correct.
05	PATH_DESTINATION_UNKNOWN	---	The link was followed to the end, but the variable was not found.
0C	OBJECT_STATE_CONFLICT	8010	Downloading, starting up
		8011	There is an error in tag memory.
13	NOT_ENOUGH_DATA	---	The data length was too short for the specified service.
15	TOO_MUCH_DATA	---	The data length was too long for the specified service.

CIP status	Meaning	Add status	Cause
1F	VENDOR_SPECIFIC_ERROR	0102,2103	An attempt was made to write a constant or read-only variable.
		0104,1103	The specified address and size exceed a segment boundary.
		8001	An internal error occurred. (An information inconsistency was detected in the interface in the Module.)
		8007	An inaccessible variable was specified.
		8029	A region that all cannot be accessed at the same time was specified for SimpleData-Segment.
		8031	An internal error occurred. (A memory allocation error occurred.)
20	INVALID_PARAMETER	8009	A segment type error occurred.
		800F	There is an inconsistency in data length information in the Request Data.
		8017	More than one element was specified for a variable that does not have elements.
		8018	Zero elements or data that exceeded the range of the array was specified for an array.
		8021	A value other than 0 and 2 was specified for an AddInfo area.
		8022	• The data type that is specified in the request service data does not agree with the tag information. • The AddInfo Length in the request service data is not 0.
		8023	An internal error occurred. (An illegal command format was used.)
		8024	An internal error occurred. (An illegal command length was used.)
		8025	An internal error occurred. (An illegal parameter was used.)
		8027	An internal error occurred. (A parameter error occurred.)
		8028	• An attempt was made to write an out-of-range value for a variable for which a subrange is specified. • An attempt was made to write an undefined value to an enumeration variable.

7-7 Variable Data Types

This section provides the data types of variables that can be used with CIP message communications.

7-7-1 Data Type Codes

The following codes are given to variable data types.

Data Type	Code (hex)	Group*1
Boolean (bit)	C1	CIP Common
SINT (1-byte signed binary)	C2	CIP Common
INT (1-word signed binary)	C3	CIP Common
DINT (2-word signed binary)	C4	CIP Common
LINT (4-word signed binary)	C5	CIP Common
USINT (1-byte unsigned binary)	C6	CIP Common
UINT (1-word unsigned binary)	C7	CIP Common
UDINT (2-word unsigned binary)	C8	CIP Common
ULINT (4-word unsigned binary)	C9	CIP Common
REAL (2-word floating point)	CA	CIP Common
LREAL (4-word floating point)	CB	CIP Common
STRING	D0	CIP Common
BYTE (1-byte hexadecimal)	D1	CIP Common
WORD (1-word hexadecimal)	D2	CIP Common
DWORD (2-word hexadecimal)	D3	CIP Common
TIME (8-byte data)	DB	CIP Common
LWORD (4-word hexadecimal)	D4	CIP Common
Abbreviated STRUCT	A0	CIP Common
STRUCT	A2	CIP Common
ARRAY	A3	CIP Common
UINT BCD (1-word unsigned BCD)	04	Vendor Specific
UDINT BCD (2-word unsigned BCD)	05	Vendor Specific
ULINT BCD (4-word unsigned BCD)	06	Vendor Specific
ENUM	07	Vendor Specific
DATE_NSEC	08	Vendor Specific
TIME_NSEC	09	Vendor Specific
DATE_AND_TIME_NSEC	0A	Vendor Specific
TIME_OF_DAY_NSEC	0B	Vendor Specific
Union	0C	Vendor Specific

*1. “CIP Common” indicates codes that are defined in the CIP Common Specifications. “Vendor Specific” indicates codes that are assigned by OMRON.

7-7-2 Common Format

The basic format on the data line is shown below.

Data Format

USINT	Data Type		Refer to *Data Type Codes* on page 8-43 for specific values.
USINT	AddInfo Length		Additional information: Field length in bytes
	(AddInfo)		Additional information: CRC value of structure or other information
UINT	Num of Element	(L)	This field exists only in the parameters for the variable write service.
		(H)	
	Actual data		

7-7-3 Elementary Data Types

Fixed-length Byte Data

Applicable data types: BYTE, USINT, and SINT

Data Format

USINT	Data Type		
USINT	00h		
UINT	Num of Elem	(L)	01 hex
		(H)	00 hex
USINT	Data		

Fixed-length 2-byte Data

Applicable data types: INT, UINT, UINT BCD, and WORD

Data Format

USINT	Data Type		
USINT	00h		
UINT	Num of Elem	(L)	01 hex
		(H)	00 hex
	Data	(L)	
		(H)	

Fixed-length 4-byte Data

Applicable data types: DINT, UDINT, UDINT BCD, REAL, and DWORD

Data Format

USINT	Data Type		
USINT	00h		
UINT	Num of Elem	(L)	01 hex
		(H)	00 hex
	Data	(LL)	
		(LH)	
		(HL)	
		(HH)	

Fixed-length 8-byte Data

Applicable data types: LINT, ULINT, ULINT BCD, LREAL, and LWORD
Data Format

USINT	Data Type		
USINT	00 hex		
UINT	Num of Elem	(L)	01 hex
		(H)	00 hex
	Data	(Least-significant byte)	
		:	
		:	
		:	
		:	
		:	
		:	
		(Most-significant byte)	

Boolean Data

Data Format

USINT	Data Type		C1 hex
USINT	00 hex		
UINT	Num of Elem	(L)	01 hex
		(H)	00 hex
USINT	Status		01 hex: TRUE, 00 hex : FALSE
USINT	Forced set/reset information*		01 hex: Forced, 00 hex: Not forced

*1. Specify 0 when writing data.

7-7-4 Derived Data Types

Arrays and structures are handled as derived data types.

Accessing One Member

The data format for accessing one element of an array or one member of a structure is the same as the data format for the corresponding elementary data type.
Example: If you specify Var[5] to access a variable defined with UINT Var[10], use the same data format as for UINT data.

Accessing More Than One Element at the Same Time

● Arrays

- Accessing an Entire Array
  If you access an array variable without specifying an element, the entire array is accessed.

The following data format is used.

Data Format

USINT	Data Type	Data type of array elements (A1 hex is not used.)
USINT	00 hex	
UINT	Num of Elem (L) / (H)	Gives the number of elements in the array.
	Data : Data	The actual data for the elements of the array are given in order in the same format as when the elements are accessed individually.*

*1. For STRING data, the output format differs from the format when accessing individual elements in the following ways.

- There are no fields for the text string lengths. Only the text strings (including NULL) are given.
- The transferred data length is not the combined lengths of the text strings, but the memory size that is allocated to the STRING variable.
- Upper and lower bytes are reversed.

Example: The outputs will be as follows for a STRING array named s that has two elements (with the data quantity per element is set to 4 bytes)
when s[0]="ab", and s[1] is "d".
Individual [0]: D0 00 03 00 61 62 63 (hex)
Entire array: D0 00 62 61 ?? 00 00 64 ?? ?? (hex) (??: Invalid data.)

- Handling Multi-dimensional Array

  Elements for a multi-dimensional array are given in order from the deepest elements.
  For example, the data is read in the following format when Var is specified for a variable defined with UINT Var[2][2].

  Data Format

USINT	C7 hex	Data type code for UINT
USINT	00 hex	
UINT	Value of *Var[0][0]* (L) / (H)	
UINT	Value of *Var[0][1]* (L) / (H)	
UINT	Value of *Var[1][0]* (L) / (H)	
UINT	Value of *Var[1][1]* (L) / (H)	

  The following data format is used for a BOOL array (using BOOL b[2][3] as an example).

  Data Format

USINT	C1 hex (data type code for BOOL)							
USINT	00 hex							
(WORD)	rsv	rsv	b[1][2]	b[1][1]	b[1][0]	b[0][2]	b[0][1]	b[0][0]
	rsv	rsv	rsv	rsv	rsv	rsv	rsv	rsv

- Exceptions When Specifying the Num of Element Field

The following data format is used if a specification is made in the Num of Element field for a BOOL array. (Refer to *7-4-5 Specifying Variable Names in Request Paths* on page 7-44 for information on the Num of Element field.) The status (TRUE/FALSE) is given in order for each element of the BOOL variable.

Data Format

USINT	Data Type		C1 hex
USINT	00 hex		
UINT	Num of Elem	(L)	Gives the number of elements in the array.
		(H)	
USINT	Status		01 hex: TRUE, 00 hex: FALSE
:	:		
USINT	Status		

● Structure Variables

- Accessing an Entire Structure

  If a structure variable is specified, it is treated as an access request for all of the members of the structure.

Data Format

USINT	Data Type		A0 hex (Abbreviated STRUCT)
USINT	02 hex		
UINT	CRC	(L)	CRC value for the structure de
		(H)	
UINT	Num of Elem	(L)	01 hex
		(H)	00 hex
	:		
	Data		
	:		

8

Socket Service

8-1 Basic Knowledge on Socket Communications

8-1-1 Sockets

A socket is an interface that allows you to directly use TCP or UDP functions from the user program.
On a host computer (e.g., personal computer), sockets are provided in the form of a C language interface library. If you load the library, you can program communications via TCP and UDP in the user program.
On a UNIX computer, a socket interface is provided in the format of system calls.
With a built-in EtherNet/IP port, you can execute instructions in the user program by using sockets.
Through the communications services with sockets, you can send and receive data to and from remote nodes, i.e., between the host computer and Controllers or between Controllers.
Built-in EtherNet/IP ports support UDP socket service as well as TCP socket service.

8-1-2 Port Numbers for Socket Services

Ports 0 to 1023 to be used for TCP/IP are reserved as well-known ports. In addition, ports 1024 to 49151 are reserved as registered ports by the protocols that are used.
Therefore, we recommend that you use port numbers 49152 to 65535 for applications other than the protocols that are registered with the socket service.
You cannot specify port number 0 for the built-in EtherNet/IP port.
Furthermore, the built-in EtherNet/IP port uses TCP/UDP ports for some applications, therefore make sure to avoid those ports when you set ports. Refer to *5-2 TCP/ UDP Port Numbers Used for the Built-in EtherNet/IP Port* on page 5-15 for details on the TCP/UDP port numbers that are used by the built-in EtherNet/IP ports.

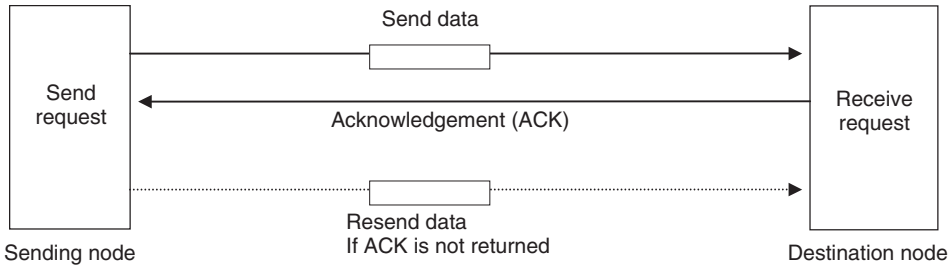
8-2 Basic Knowledge on Protocols

8-2-1 Differences between TCP and UDP

The TCP and UDP functions used on socket services differ as shown below.

TCP Communications

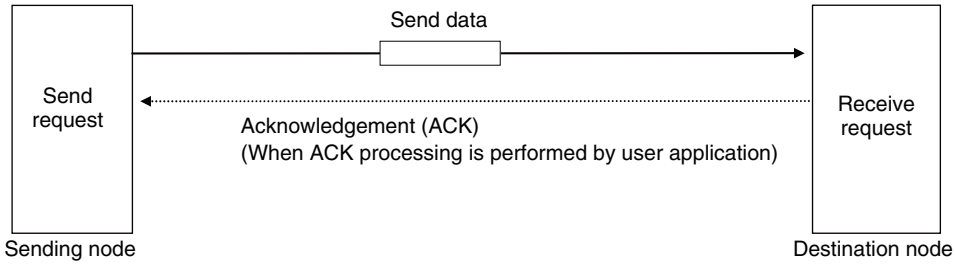
The following operations are performed each time data is sent to ensure that it reaches the destination node.
The destination node returns an acknowledgment (ACK) when data is received normally.
The sending node sends the next data after ACK is returned. It resends the same data if ACK is not received within a certain length of time.



In TCP, the remote IP address and the remote TCP port number are specified when a request is made to open a socket. The variables that store the data to send are specified when the send request is made.

UDP Communications

Data is simply sent to the destination node, and neither acknowledgment nor resends are performed like they are for TCP.
To increase the reliability of communications, some user application must be used to perform data re-send processing.

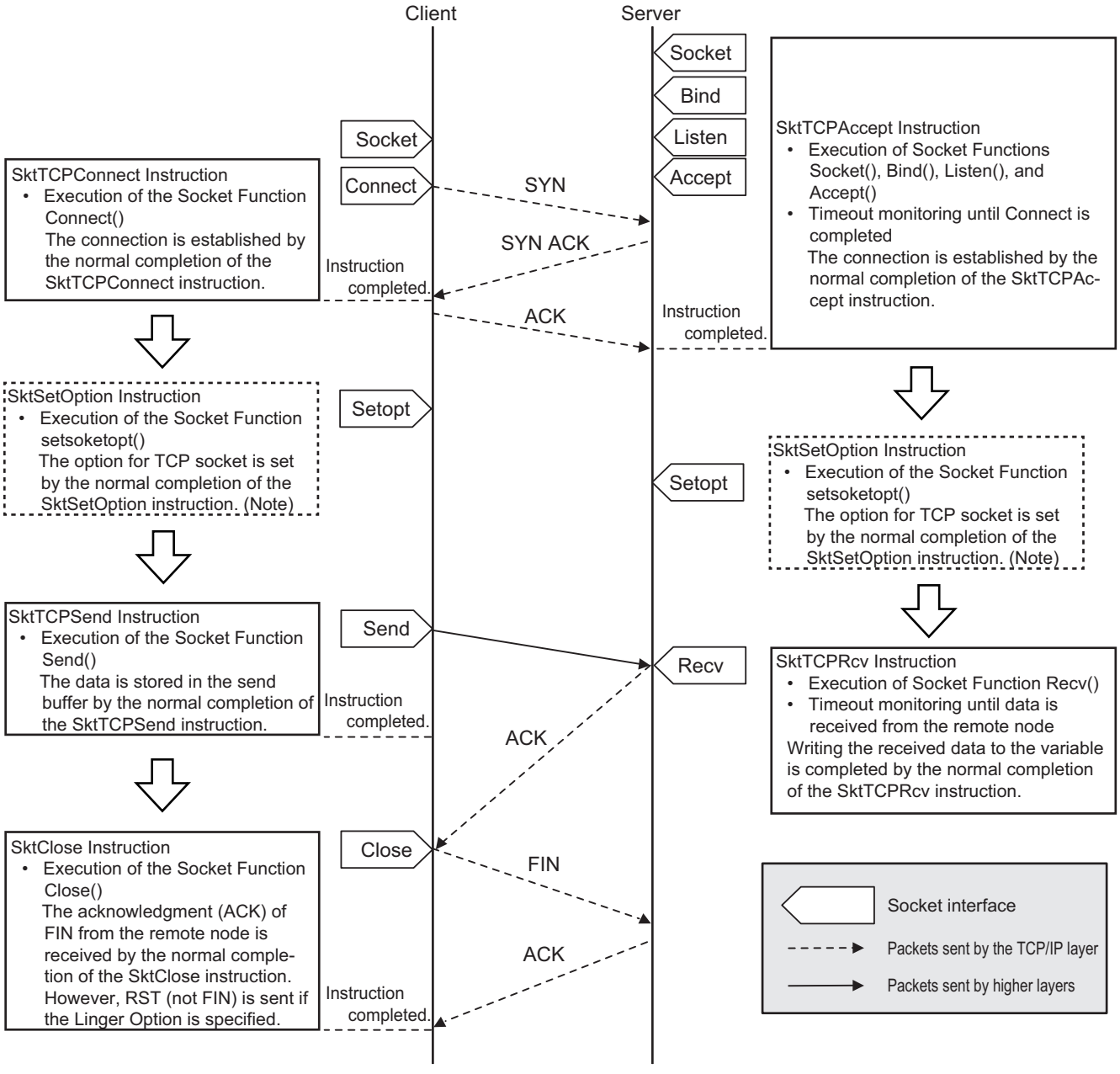


In UDP, the remote IP address and the remote UDP port number are not specified when a request is made to open a socket. The variables that store the remote IP address, the remote UDP port number, and the data to send are specified when the send request is made.
(The send data includes information on the IP address and UDP port number of the sending node.)

Furthermore, once a socket is opened in UDP, communications with other remote nodes is possible without closing the socket.

TCP Communications Procedure

You execute socket communications instructions in sequence to perform TCP communications for the built-in EtherNet/IP port.



Note Set the socket option as required. Refer to the *NJ/NX-series Instructions Reference Manual (Cat. No. W502)* for the socket option.

8-2-2 Fragmenting of Send Data

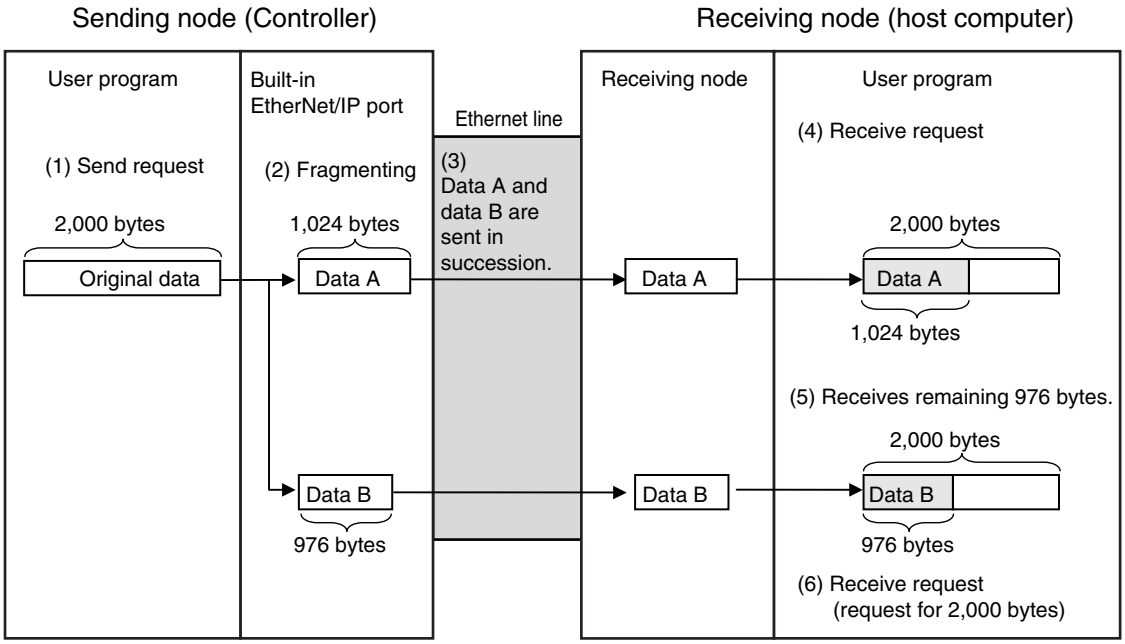
The receive buffer for the built-in EtherNet/IP port is a maximum of 9,000 bytes per socket handle. If any data that is larger than 9,000 bytes is received, the data is discarded.

Up to 2,000 bytes can be received for a single request. In this case, the data is sent in fragments as described below.

Using TCP

The following figure shows what occurs when data is sent in fragments in TCP communications.

1. A send request is sent from the user program at the sending node. It specifies a variable with a data length of 2,000 bytes.
2. The built-in EtherNet/IP port separates the send data into 1,024 bytes as data A and 976 bytes as data B.
3. Data A and data B are sent in sequence by the sending node.
4. After data A is received, the remaining data B is received.



Data is delivered to the user program in a fragmented form in TCP communications, as shown above.

The size of received data must be checked to confirm that all the data was received before the next receive request is made. (You can use the *RecvSize* output variable of the socket receive request instruction to check the received data.)

Additional Information

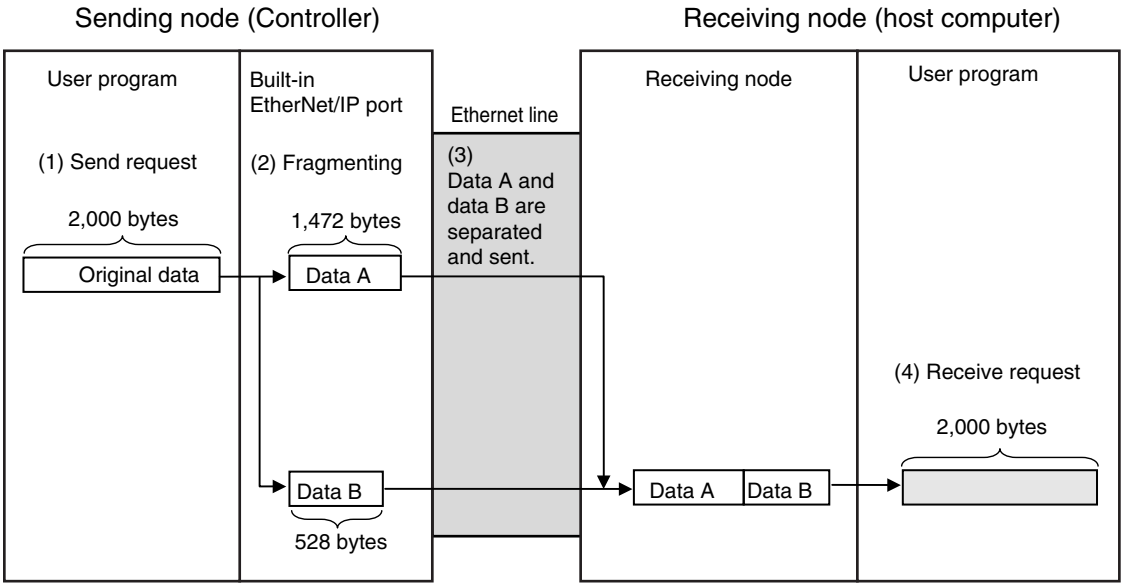
If TCP is used to send data to a different segment, the data is separated into 536-byte fragments.

Using UDP

The following figure shows what occurs when data is sent in fragments in UDP communications.

1. A send request is sent from the user program at the sending node. It specifies a variable with a data length of 2,000 bytes.

2. The built-in EtherNet/IP port separates the send data into 1,472 bytes as data A and 528 bytes as data B.
3. Data A and data B are sent in sequence by the sending node.
4. Data A and data B are joined and restored as the original send data, and the data is passed to the user program.



Since UDP communications are performed in datagram units as shown above, send data is restored in the original data format before it is passed to the user program.

8-2-3 Data Receive Processing

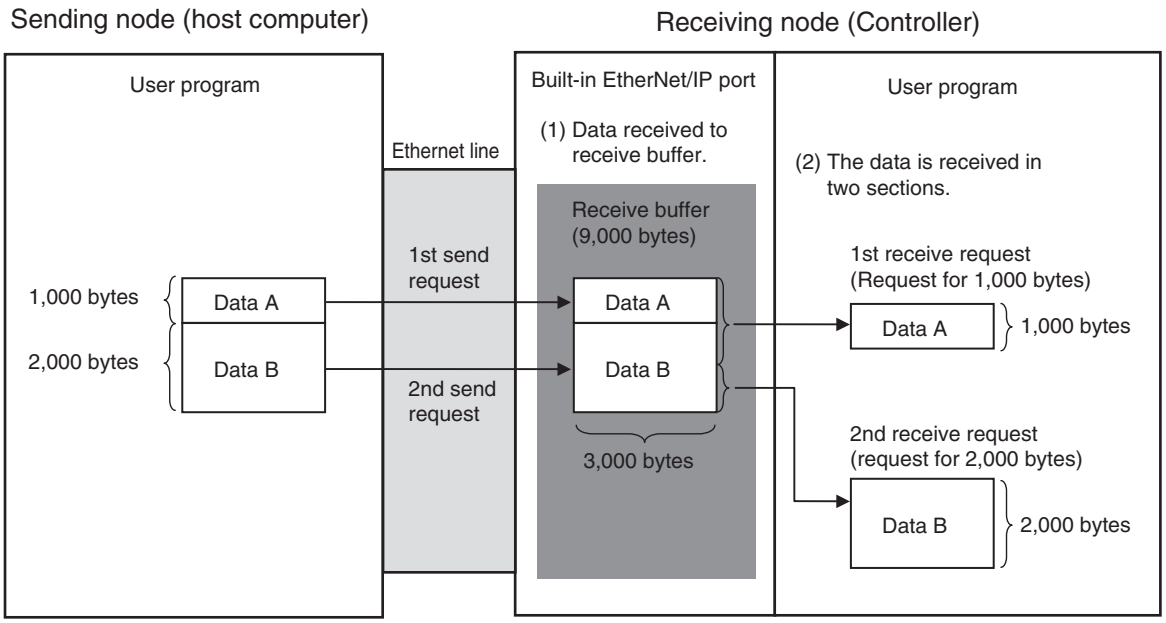
This section describes data receive processing for TCP and UDP.

TCP Receive Processing

In TCP communications, receive data stored in the receive buffer (a maximum of 9,000 bytes) can be divided to be received.
Thus, if received data is larger than the maximum size of data that can be received with one data request (2,000 bytes), more than one receive request can be sent to receive all of the data.
If the data in the receive buffer is smaller than the size of the variable specified by the receive request, the entire receive data is received.

Example) Receiving 3,000 Bytes of Receive Data in Two Sections

1. The data is divided to be sent in two sends from the sending node, and is stored in the receive buffer.
2. More than one receive request is used to receive all of the send data.



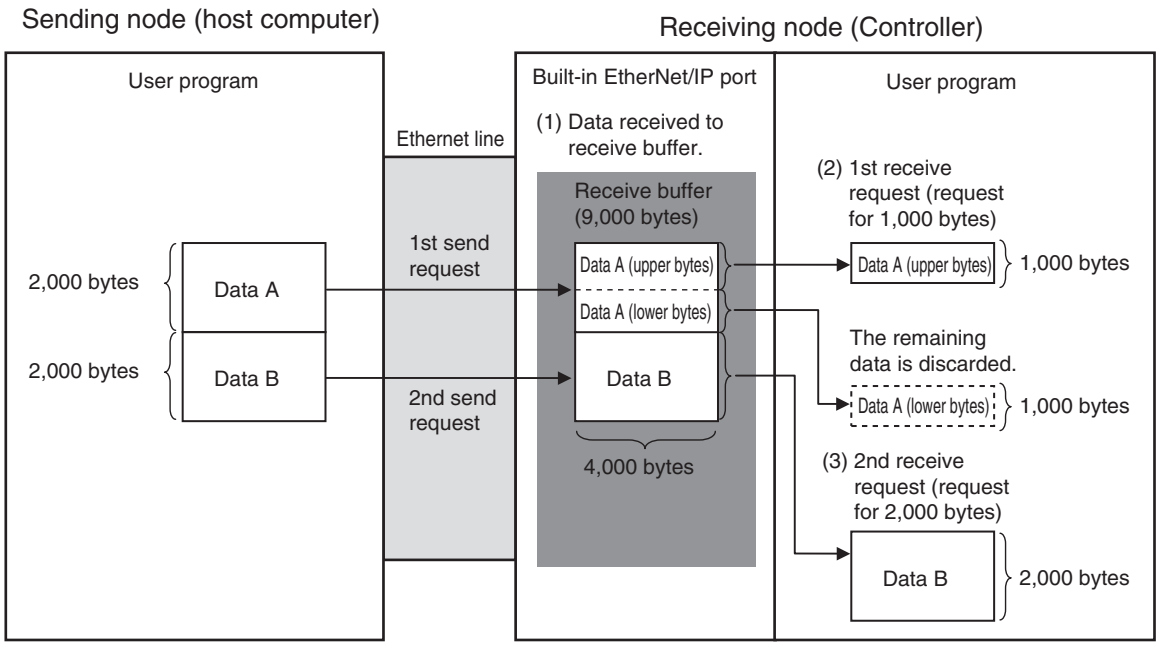
UDP Receive Processing

In UDP communications, receive data stored in the receive buffer (a maximum of 9,000 bytes) cannot be divided to be received.
Therefore, if data is sent for one send request, it must be received with one receive request.
The following must be considered to receive data at the receiving node.

- When the Size of the Variable Specified in the Receive Request Is Smaller Than the Data Sent with the Send Request
  If receive data exceeds the size of the variable specified in the receive request, the excess of the data is discarded.
  If the data in the receive buffer is smaller than the size of the variable specified in the receive request, the entire receive data is received.

  Example 1: 1,000-Byte Receive Request Is Made for 2,000-Byte Data

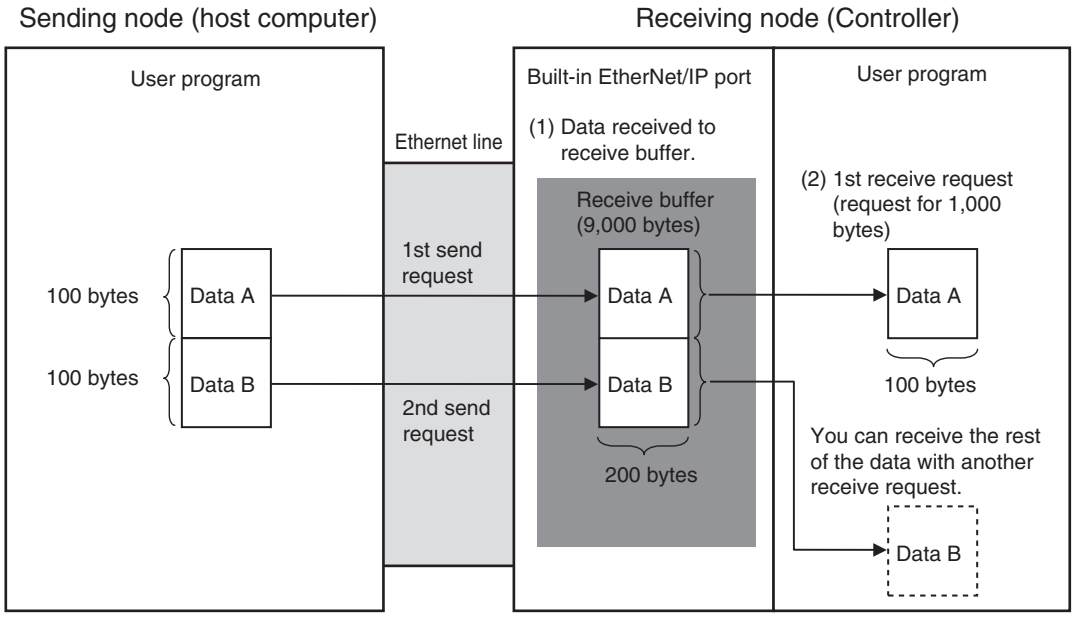
  1. The data is divided to be sent in two sends from the sending node, and is stored in the receive buffer.
  2. If a 1,000-byte receive request is made for the first send, the remaining 1,000 bytes of the data is discarded.
  3. If the next receive request is made for 2,000 bytes, the data for the second send is all received.



- When Only One Receive Request Is Made for Data Sent for Multiple Send Requests
  If data is sent for multiple sent requests, you cannot receive the entire data with one receive request regardless of the size of the data.

  Example 2: 1,000-Byte Receive Request Is Made for 200-Byte Data Sent for Two Send Requests

  1. The data is divided to be sent in two sends from the sending node, and is stored in the receive buffer.
  2. Even if a receive request is made for 2,000 bytes, only 100 bytes of the data is received as requested with the first send request.



8-2-4 Broadcasting

If you specify a broadcast address as the destination IP address for a UDP socket, data can be broadcast to all nodes on the network to which the host for the EtherNet/IP port belongs.
If there is a router on the network, packets are not sent beyond the router.
You can broadcast up to 1,472 bytes of data. Data larger than the maximum size cannot be broadcast.

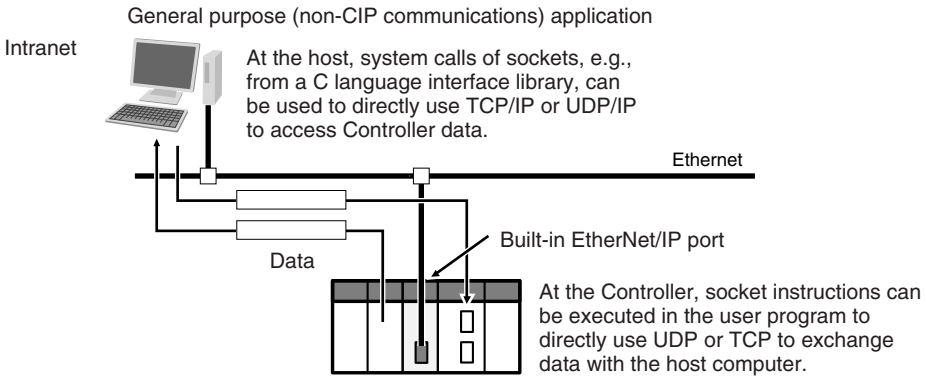
You can specify either of the two following types of broadcast addresses.

- Local Broadcast
  If no destination IP address is specified, the following IP address is specified automatically.
  Network segment: The network segment of the local IP address is set.
  Host segment: All bits are set to 1.
- Global Broadcast
  Specify this type when the IP address of the local node or the subnet to which the local node belongs is unknown.
  As shown below, every bit of the 32-bit address is set to 1.
  255.255.255.255

8-3 Overview of Built-in EtherNet/IP Port Socket Services

8-3-1 Overview

Socket services on the built-in EtherNet/IP port are used to exchange data between Controllers and general-purpose applications that do not support CIP message communications.
The Controller requests the socket service from the user program.



Overview of Socket Services with Socket Service Instructions

You can use socket services by executing socket service instructions.
The maximum total number of UDP and TCP sockets that you can use is given in the following table.

UDP/TCP	Maximum number of sockets			
	NX-series CPU Unit		NJ-series CPU Unit	
	NX102	Other than NX102	Unit version 1.00 to 1.02	Unit version 1.03 or later
UDP socket service	Total of 60 sockets	Total of 30 sockets	Total of 16 sockets	Total of 30 sockets
TCP socket service				
Secure socket service*1			Not supported	

*1. An NX102-□□00 CPU Unit with unit version 1.46 or later or an NX102-□□20 CPU Unit with unit version 1.37 or later and Sysmac Studio version 1.46 or higher are required to use the secure socket services.
An NX1P2-□□□□□□ CPU Unit with unit version 1.46 or later and Sysmac Studio version 1.46 or higher are required.

8-3-2 Procedure

1	Make the settings that are required for socket services. Refer to *8-4 Settings Required for the Socket Services* on page 8-11.

↓

2	Execute the socket service instructions from the user program. Refer to *8-5 Socket Service Instructions* on page 8-12.

8-4 Settings Required for the Socket Services

Make the following settings in the Unit Setup to use the socket services.

Sysmac Studio Unit Settings Tab Page	Setting	Setting conditions
Setting	Local IP Address	Required
	Subnet Mask	Required
	TCP/IP Keep Alive	Optional (Change when the default setting of 5 minutes is unacceptable.)
	Linger Option	Optional

Additional Information

Make this setting in the **TCP/IP Settings** Display. Refer to *4-1 **TCP/IP Settings** Display* on page 4-2 for information on the **TCP/IP Settings** Display.

8-5 Socket Service Instructions

You can use the following socket service instructions for socket services.
Refer to the *NJ/NX-series Instructions Reference Manual (Cat. No. W502)* for information on the socket service instructions.

UDP/TCP	Instruction	Socket service
UDP sockets	SktUDPCreate	Create UDP Socket instruction
	SktUDPRcv	UDP Socket Receive instruction
	SktUDPSend	UDP Socket Send instruction
TCP sockets	SktTCPAccept	Accept TCP Socket instruction
	SktTCPConnect	Connect TCP Socket instruction
	SktTCPRcv	TCP Socket Receive instruction
	SktTCPSend	TCP Socket Send instruction
	SktGetTCPStatus	Read TCP Socket Status instruction
Services for both UDP and TCP sockets	SktClose	Close TCP/UDP Socket instruction
	SktClearBuf	Clear TCP/UDP Socket Receive Buffer instruction
	SktSetOption	Set TCP Socket Option instruction

Precautions for Correct Use

You can execute a maximum of 32 socket service instructions (64 for NX102) at the same time. Perform exclusive control in the user program so that 33 or more socket instructions (65 or more for NX102) will not be executed at the same time.

8-6 Details on Using the Socket Services

8-6-1 Using the Socket Services

The following table shows the maximum total number of TCP and UDP sockets for the built-in EtherNet/IP port.

UDP/TCP	Maximum number of sockets			
	NX-series CPU Unit		NJ-series CPU Unit	
	NX102	Other than NX102	Unit version 1.00 to 1.02	Unit version 1.03 or later
UDP socket service	Total of 60 sockets	Total of 30 sockets	Total of 16 sockets	Total of 30 sockets
TCP socket service				

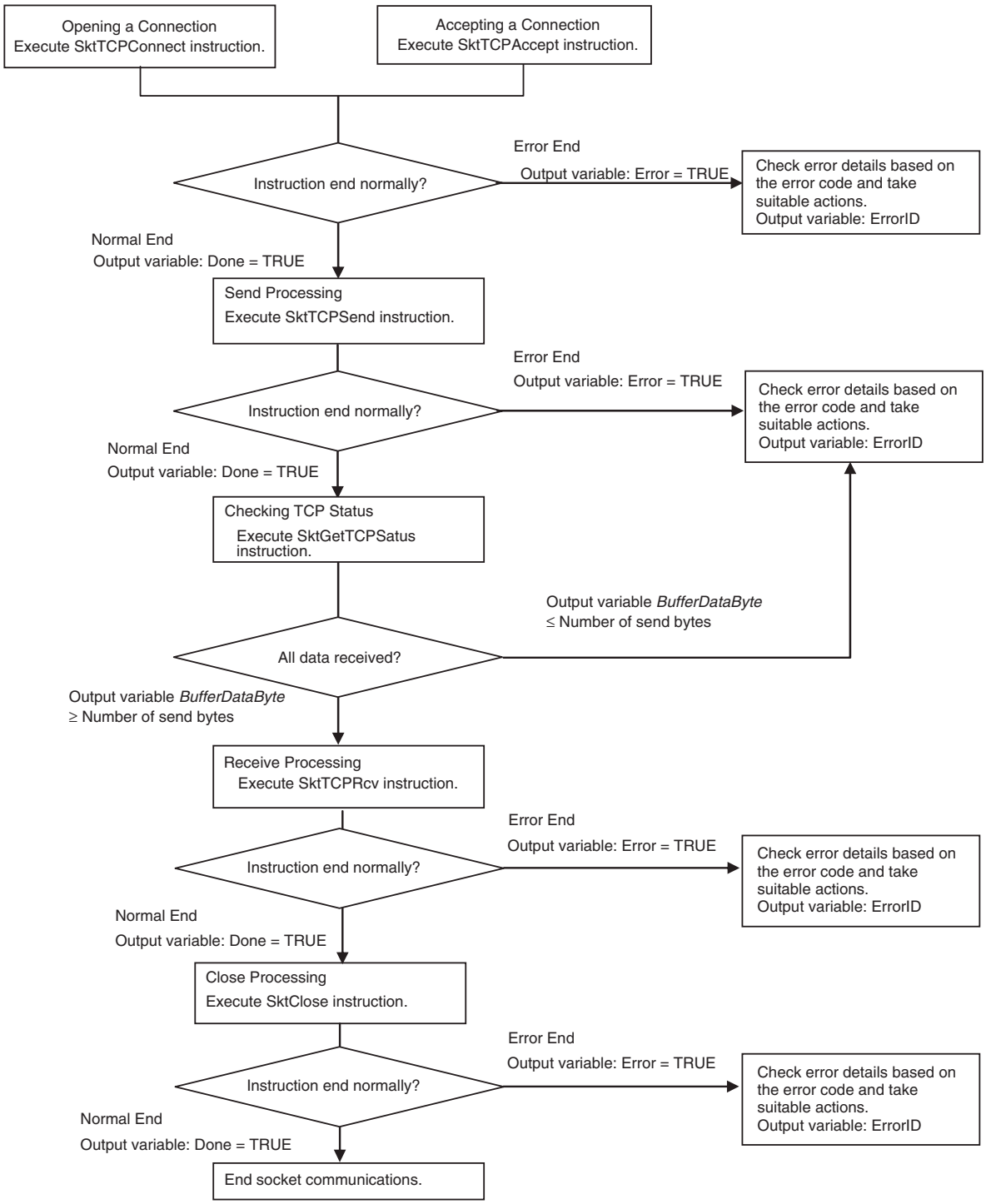
To use these sockets for communications, special ST instructions for sockets are executed to perform the following processes.

Open processing:	This process places the socket in a usable state. This is the first process to use socket services. With TCP, open processing is performed until a connection is established.
Close processing:	This process ends the use of the socket. With TCP, it closes the connection.
Send processing	This process sends data from the socket.
Receive processing	This process receives data from the socket.
Clear processing:	This process clears the receive buffer to remove data received from the remote node.

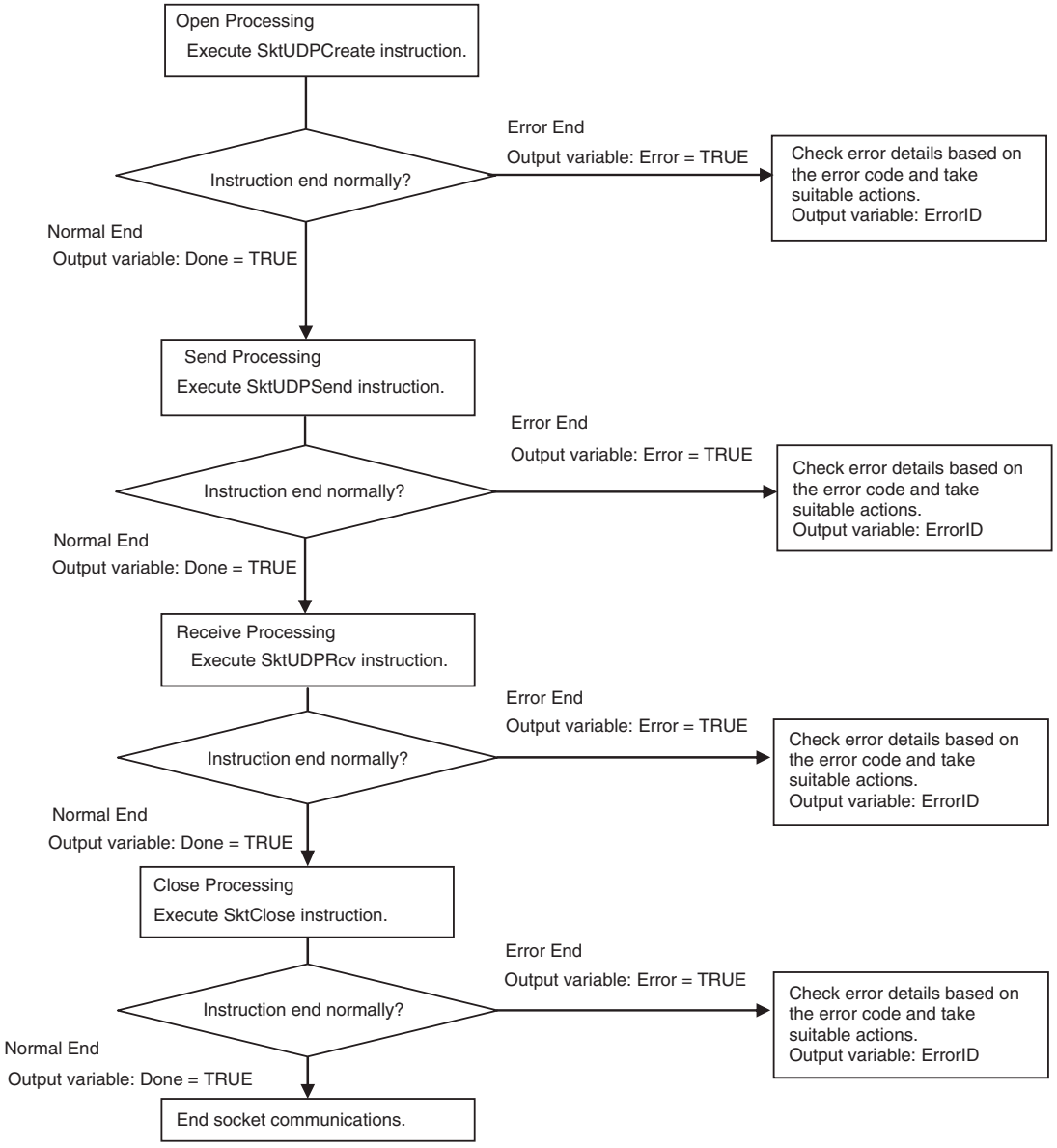
8-6-2 Procedure to Use Socket Services

You execute special instructions for sockets in sequence to use the socket services according to the procedure shown below.
Use the values of the output variables for each instruction to confirm that each instruction is normally completed.

TCP


Opening a Connection
Execute SktTCPConnect instruction.
Accepting a Connection
Execute SktTCPAccept instruction.
Instruction end normally?
Error End
Output variable: Error = TRUE
Check error details based on the error code and take suitable actions.
Output variable: ErrorID
Normal End
Output variable: Done = TRUE
Send Processing
Execute SktTCPSend instruction.
Instruction end normally?
Error End
Output variable: Error = TRUE
Check error details based on the error code and take suitable actions.
Output variable: ErrorID
Normal End
Output variable: Done = TRUE
Checking TCP Status
Execute SktGetTCPSatus instruction.
Output variable BufferDataByte ≤ Number of send bytes
All data received?
Output variable BufferDataByte ≥ Number of send bytes
Receive Processing
Execute SktTCPRcv instruction.
Instruction end normally?
Error End
Output variable: Error = TRUE
Check error details based on the error code and take suitable actions.
Output variable: ErrorID
Normal End
Output variable: Done = TRUE
Close Processing
Execute SktClose instruction.
Instruction end normally?
Error End
Output variable: Error = TRUE
Check error details based on the error code and take suitable actions.
Output variable: ErrorID
Normal End
Output variable: Done = TRUE
End socket communications.

UDP

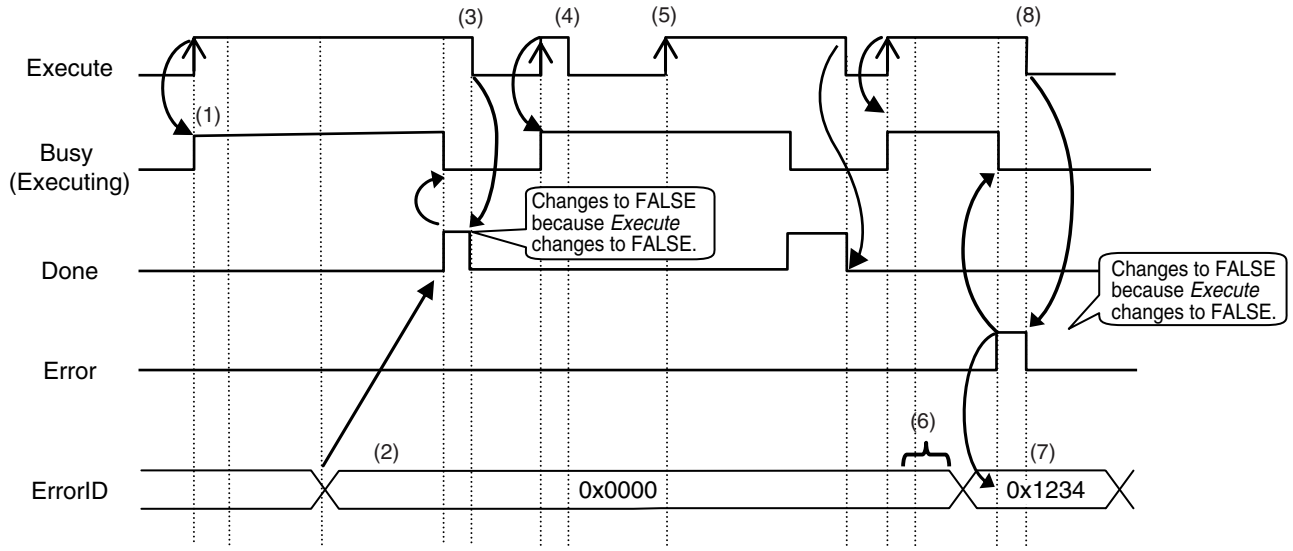


8-6-3 Timing Chart for Output Variables Used in Communications

Output Variable Operation and Timing

You can monitor the values of the output variables to determine the status throughout instruction execution.

The following timing chart shows the operation of the output variables.

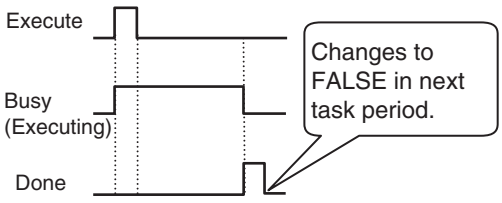


1. When *Execute* changes to TRUE, the instruction is executed and *Busy* changes to TRUE.
2. After the results of instruction execution are stored in the output variables, *Done* changes to TRUE and *Busy* changes to FALSE.
3. When *Execute* changes to FALSE, *Done* returns to FALSE.
4. When *Execute* changes to TRUE again, *Busy* changes to TRUE.
5. *Execute* is ignored if it changes to TRUE during instruction execution (i.e., when *Busy* is TRUE).
6. If an error occurs, several retries are attempted internally. The error code in *ErrorID* is not updated during the retries.
7. When a communications error occurs, *Error* changes to TRUE and the *ErrorID* is stored. Also, *Busy* and *Done* change to FALSE.
8. When *Execute* changes to FALSE, *Error* changes to FALSE.

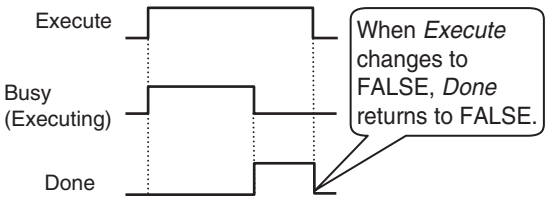
Precautions for Correct Use

If *Execute* changes back to FALSE before *Done* changes to TRUE, *Done* stays TRUE for only one task period. (Example 1)
If you want to see if *Done* is TRUE at any time, make sure to keep *Execute* TRUE until you confirm that *Done* is TRUE.
If *Execute* is TRUE until *Done* changes to TRUE, *Done* stays TRUE until *Execute* changes to FALSE. (Example 2)

Example 1

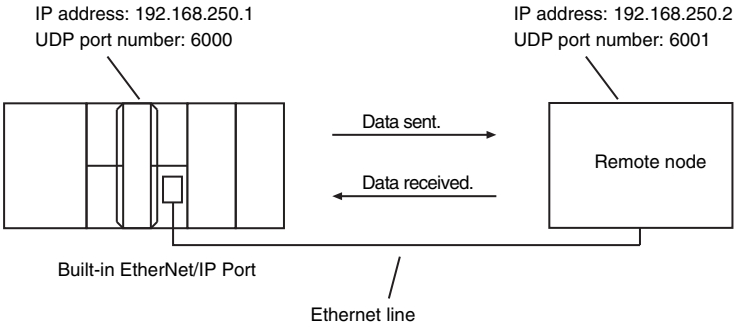


Example 2



8-6-4 UDP Sample Programming

In this sample, the UDP socket service is used for data communications between the NJ/NX-series Controller and a remote node.
In this example, programming is also required in the remote node. The order of sending and receiving is reversed in comparison with the above procedure.



Local Node Programming

The processing procedure at the local node is as follows:

1 The SktUDPCreate instruction is used to make a request to create a UDP socket.

2 The SktUDPSend instruction is used to make a send request. The data in SendSocketDat[] is sent.

3 The SktUDPRcv instruction is used to make a receive request. The received data is stored in RcvSocketDat[].

4 The SktClose instruction is used to close the socket.

ST

Internal variables	Variable	Data type	Initial value	Comment
	Trigger	BOOL	False	Execution condition
	DoSendAndRcv	BOOL	False	Processing
	Stage	INT	0	Status change
	RcvSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	Received data
	WkSocket	_sSOCKET	(Handle:=0, SrcAdr:=(PortNo:=0,IpAdr:=''), DstAdr:=(PortNo:=0,IpAdr:=''))	Socket
	SendSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	Send data
	SktUDPCreate_instance	SktUDPCreate		
	SktUDPSend_instance	SktUDPSend		

Internal variables	Variable	Data type	Initial value	Comment
	SktUDPRcv_instance	SktUDPRcv		
	SktClose_instance	SktClose		

External variable	Variable	Data type	Constant	Comment
	_EIP_EtnOnlineSta*1	BOOL	☑	Online

*1. For an NX701 CPU Unit and an NX102 CPU Unit, replace the variable with _EIP1_EtnOnlineSta (Port1 Online) or _EIP2_EtnOnlineSta (Port2 Online), depending on the built-in EtherNet/IP port which is used.
For an NX1P2 CPU Unit, replace the variable with _EIP1_EtnOnlineSta (Port1 Online).

```
// Start sequence when Trigger changes to TRUE.
IF((Trigger=TRUE) AND(DoSendAndRcv=FALSE) AND (_EIP_EtnOnlineSta=TRUE))THEN
    DoSendAndRcv             :=TRUE;
    Stage                    :=INT#1;
    SktUDPCreate_instance(Execute:=FALSE);                // Initialize instance.
    SktUDPSend_instance(                                  // Initialize instance.
        Execute              :=FALSE,
        SendDat              :=SendSocketDat[0]);         // Dummy
    SktUDPRcv_instance(                                   // Initialize instance.
        Execute              :=FALSE,
        RcvDat               :=RcvSocketDat[0]);          // Dummy
    SktClose_instance(Execute:=FALSE);                    // Initialize instance.
END_IF;

IF (DoSendAndRcv=TRUE) THEN
    CASE Stage OF
    1 :                                                   // Request to create a s
ocket.
        SktUDPCreate_instance(
            Execute          :=TRUE,
            SrcUdpPort       :=UINT#6000,                 // Local UDP port number
            Socket    =>WkSocket);                        // Socket

        IF (SktUDPCreate_instance.Done=TRUE) THEN
            Stage            :=INT#2;                     // Normal end
        ELSIF (SktUDPCreate_instance.Error=TRUE) THEN
            Stage            :=INT#10;                    // Error end
        END_IF;

    2 :                                                   // Send request
        WkSocket.DstAdr.PortNo :=UINT#6001;
        WkSocket.DstAdr.IpAdr  :='192.168.250.2';
```

```
        SktUDPSend_instance(
            Execute          :=TRUE,
            Socket           :=WkSocket,                    // Socket
            SendDat          :=SendSocketDat[0],            // Send data
            Size             :=UINT#2000);                  // Send data size

        IF (SktUDPSend_instance.Done=TRUE) THEN
            Stage            :=INT#3;                       // Normal end
        ELSIF (SktUDPSend_instance.Error=TRUE) THEN
            Stage            :=INT#20;                      // Error end
        END_IF;
    3 :                                                     // Receive request
        SktUDPRcv_instance(
            Execute          :=TRUE,
            Socket           :=WkSocket,                    // Socket
            TimeOut          :=UINT#0,                      // Timeout value
            Size             :=UINT#2000,                   // Receive data size
            RcvDat           :=RcvSocketDat[0]);            // Receive data

        IF (SktUDPRcv_instance.Done=TRUE) THEN
            Stage            :=INT#4;                       // Normal end
        ELSIF (SktUDPRcv_instance.Error=TRUE) THEN
            Stage            :=INT#30;                      // Error end
        END_IF;

    4 :                                                     // Requestto close the s
ocket
        SktClose_instance(
            Execute          :=TRUE,
            Socket           :=WkSocket);                   // Socket

        IF (SktClose_instance.Done=TRUE) THEN
            Stage            :=INT#0;                       // Normal end
        ELSIF (SktClose_instance.Error=TRUE) THEN
            Stage            :=INT#40;                      // Error end
        END_IF;

    0 :                                                     // Normal end
        DoSendAndRcv         :=FALSE;
        Trigger              :=FALSE;

    ELSE                                                    // Interrupted by error.
        DoSendAndRcv         :=FALSE;
        Trigger              :=FALSE;
    END_CASE;

END_IF;
```

Remote Node Programming

The processing procedure at the remote node is as follows:

1. The SktUDPCreate instruction is used to make a request to create a UDP socket.
2. The SktUDPRcv instruction is used to make a receive request. The received data is stored in RcvSocketDat[].
3. The SktUDPSend instruction is used to make a send request. The data in SendSocketDat[] is sent.
4. The SktClose instruction is used to close the socket.

ST

Internal variables	Variable	Data type	Initial value	Comment
	Trigger	BOOL	False	Execution condition
	DoSendAndRcv	BOOL	False	Processing
	Stage	INT	0	Status change
	RcvSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	Received data
	WkSocket	_sSOCKET	(Handle:=0, SrcAdr:=(PortNo:=0, IpAdr:=''), DstAdr:=(PortNo:=0, IpAdr:=''))	Socket
	SendSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	Send data
	SktUDPCreate_instance	SktUDPCreate		
	SktUDPSend_instance	SktUDPSend		
	SktUDPRcv_instance	SktUDPRcv		
	SktClose_instance	SktClose		

External variable	Variable	Data type	Constant	Comment
	_EIP_EtnOnlineSta*1	BOOL	☑	Online

*1. For an NX701 CPU Unit and an NX102 CPU Unit, replace the variable with _EIP1_EtnOnlineSta (Port1 Online) or _EIP2_EtnOnlineSta (Port2 Online), depending on the built-in EtherNet/IP port which is used.
For an NX1P2 CPU Unit, replace the variable with _EIP1_EtnOnlineSta (Port1 Online).

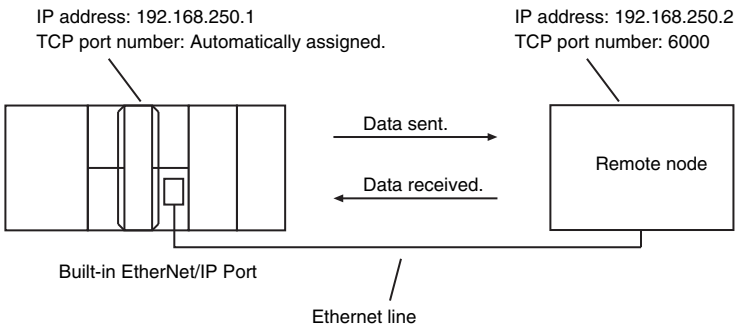
```
// Start sequence when Trigger changes to TRUE.
IF((Trigger=TRUE) AND (DoSendAndRcv=FALSE) AND (_EIP_EtnOnlineSta=TRUE))THEN
    DoSendAndRcv             :=TRUE;
    Stage                    :=INT#1;
    SktUDPCreate_instance(Execute:=FALSE);                // Initialize instance.
```

```
    SktUDPSend_instance(                                    // Initialize instance.
        Execute             :=FALSE,
        SendDat             :=SendSocketDat[0]);            // Dummy
    SktUDPRcv_instance(                                     // Initialize instance.
        Execute             :=FALSE,
        RcvDat              :=RcvSocketDat[0]);             // Dummy
    SktClose_instance(Execute:=FALSE);                      // Initialize instance.
END_IF;

IF (DoSendAndRcv=TRUE) THEN
    CASE Stage OF
    1 :                                                     // Request to create a s
ocket
        SktUDPCreate_instance(
            Execute         :=TRUE,
            SrcUdpPort      :=UINT#6001,                    // Local UDP port number
            Socket     =>WkSocket);                         // Socket

        IF (SktUDPCreate_instance.Done=TRUE) THEN
            Stage           :=INT#2;                        // Normal end
        ELSIF (SktUDPCreate_instance.Error=TRUE) THEN
            Stage           :=INT#10;                       // Error end
        END_IF;

    2 :                                                     // Receive request
        SktUDPRcv_instance(
            Execute         :=TRUE,
            Socket          :=WkSocket,                     // Socket
            TimeOut         :=UINT#0,                       // Timeout value
            Size            :=UINT#2000,                    // Receive data size
            RcvDat          :=RcvSocketDat[0]);             // Receive data

        IF (SktUDPRcv_instance.Done=TRUE) THEN
            Stage           :=INT#3;                        // Normal end
        ELSIF (SktUDPRcv_instance.Error=TRUE) THEN
            Stage           :=INT#20;                       // Error end
        END_IF;

    3 :                                                     // Send request
        WkSocket.DstAdr.PortNo :=UINT#6000;
        WkSocket.DstAdr.IpAdr  :='192.168.250.1';
        SktUDPSend_instance(
            Execute         :=TRUE,
            Socket          :=WkSocket,                     // Socket
            SendDat         :=SendSocketDat[0],             // Send data
            Size            :=UINT#2000);                   // Send data size
```

```
        IF (SktUDPSend_instance.Done=TRUE) THEN
            Stage            :=INT#4;                    // Normal end
        ELSIF (SktUDPSend_instance.Error=TRUE) THEN
            Stage            :=INT#30;                   // Error end
        END_IF;

    4 :                                                  // Request to close the
socket
        SktClose_instance(
            Execute          :=TRUE,
            Socket           :=WkSocket);                // Socket

        IF (SktClose_instance.Done=TRUE) THEN
            Stage            :=INT#0;                    // Normal end
        ELSIF (SktClose_instance.Error=TRUE) THEN
            Stage            :=INT#40;                   // Error end
        END_IF;

    0 :                                                  // Normal end
        DoSendAndRcv         :=FALSE;
        Trigger              :=FALSE;
    ELSE                                                 // Interrupted by error.
        DoSendAndRcv         :=FALSE;
        Trigger              :=FALSE;
    END_CASE;

END_IF;
```

8-6-5 TCP Sample Programming

In this sample, the TCP socket service is used for data communications between the NJ/NX-series Controller and a remote node.
In this example, programming is also required in the remote node. The order of sending and receiving is reversed in comparison with the above procedure.



Local Node Programming

The processing procedure at the local node is as follows:

1. The SktTCPConnect instruction is used to make a request for connection to the TCP port on the remote node.
2. The SktClearBuf instruction is used to clear the receive buffer of a TCP socket.
3. The SktGetTCPStatus instruction is used to read the status of the TCP socket.
4. The SktTCPSend instruction is used to make a send request The data in SendSocketDat[] is sent.
5. The SktTCPRcv instruction is executed to make a receive request. The received data is stored in RcvSocketDat[].
6. The SktClose instruction is used to close the socket.

ST

Internal variables	Variable	Data type	Initial value	Comment
	Trigger	BOOL	False	Execution condition
	DoTCP	BOOL	False	Processing
	Stage	INT	0	Status change
	RcvSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	Received data
	WkSocket	_sSOCKET	(Handle:=0, SrcAdr:=(PortNo:=0, IpAdr:=''), DstAdr:=(PortNo:=0, IpAdr:=''))	Socket
	SendSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	Send data
	SktTCPConnect_instance	SktTCPConnect		
	SktClearBuf_instance	SktClearBuf		
	SktGetTCPStatus_instance	SktGetTCPStatus		
	SktTCPSend_instance	SktTCPSend		
	SktTCPRcv_instance	SktTCPRcv		
	SktClose_instance	SktClose		

External variable	Variable	Data type	Constant	Comment
	_EIP_EtnOnlineSta*1	BOOL	☑	Online

*1. For NX701 and NX102 CPU Units, replace the variable with _EIP1_EtnOnlineSta (Port1 Online) or _EIP2_EtnOnlineSta (Port2 Online), depending on the built-in EtherNet/IP port which is used.
For an NX1P2 CPU Unit, replace the variable with _EIP1_EtnOnlineSta (Port1 Online).

```
// Start sequence when Trigger changes to TRUE.
IF ((Trigger=TRUE) AND (DoTCP=FALSE) AND (_EIP_EtnOnlineSta=TRUE)) THEN
    DoTCP                   :=TRUE;
    Stage                   :=INT#1;
    SktTCPConnect_instance(Execute:=FALSE);             // Initialize instance.
    SktClearBuf_instance(Execute:=FALSE);               // Initialize instance.
    SktGetTCPStatus_instance(Execute:=FALSE);           // Initialize instance.
    SktTCPSend_instance(                                // Initialize instance.
        Execute             :=FALSE,
        SendDat             :=SendSocketDat[0]);        // Dummy
    SktTCPRcv_instance(                                 // Initialize instance.
        Execute             :=FALSE,
        RcvDat              :=RcvSocketDat[0]);         // Dummy
    SktClose_instance(Execute:=FALSE);                  // Initialize instance.
END_IF;

IF (DoTCP=TRUE) THEN
    CASE Stage OF
    1 :                                                 // Connection request
        SktTCPConnect_instance(
            Execute         :=TRUE,
            SrcTcpPort      :=UINT#0,                   // Local TCP port number
: Automatically assigned
            DstAdr          :='192.168.250.2',          // Remote IP address
            DstTcpPort      :=UINT#6000,                // Destination TCP port
number
            Socket    =>WkSocket);                      // Socket

        IF (SktTCPConnect_instance.Done=TRUE) THEN
            Stage           :=INT#2;                    // Normal end
        ELSIF (SktTCPConnect_instance.Error=TRUE) THEN
            Stage           :=INT#10;                   // Error end
        END_IF;

    2 :                                                 // Receive buffer clear
        SktClearBuf_instance(
            Execute         :=TRUE,
            Socket          :=WkSocket);                // Socket
```

```
        IF (SktClearBuf_instance.Done=TRUE) THEN
            Stage           :=INT#3;                    //Normal end
        ELSIF (SktClearBuf_instance.Error=TRUE) THEN
            Stage           :=INT#20;                   //Error end
        END_IF;

    3 :                                                 // Status read request
        SktGetTCPStatus_instance(
            Execute         :=TRUE,
            Socket          :=WkSocket);                // Socket

        IF (SktGetTCPStatus_instance.Done=TRUE) THEN
            Stage           :=INT#4;                    // Normal end
        ELSIF (SktGetTCPStatus_instance.Error=TRUE) THEN
            Stage           :=INT#30;                   // Error end
        END_IF;

    4 :                                                 // Send request
        SktTCPSend_instance(
            Execute         :=TRUE,
            Socket          :=WkSocket,                 // Socket
            SendDat         :=SendSocketDat[0],         // Send data
            Size            :=UINT#2000);               // Send data size

        IF (SktTCPSend_instance.Done=TRUE) THEN
            Stage           :=INT#5;                    // Normal end
        ELSIF (SktTCPSend_instance.Error=TRUE) THEN
            Stage           :=INT#40;                   // Error end
        END_IF;

    5 :                                                 // Receive request
        SktTCPRcv_instance(
            Execute         :=TRUE,
            Socket          :=WkSocket,                 // Socket
            TimeOut         :=UINT#0,                   // Timeout value
            Size            :=UINT#2000,                // Receive data size
            RcvDat          :=RcvSocketDat[0]);         // Receive data

        IF (SktTCPRcv_instance.Done=TRUE) THEN
            Stage           :=INT#6;                    // Normal end
        ELSIF (SktTCPRcv_instance.Error=TRUE) THEN
            Stage           :=INT#50;                   // Error end
        END_IF;

    6 :                                                 // Request to close the
socket
        SktClose_instance(
```

```
            Execute          :=TRUE,
            Socket           :=WkSocket);                    // Socket

        IF (SktClose_instance.Done=TRUE) THEN
            Stage            :=INT#0;                        // Normal end
        ELSIF (SktClose_instance.Error=TRUE) THEN
            Stage            :=INT#60;                       // Error end
        END_IF;

    0 :                                                      // Normal end
        DoTCP                :=FALSE;
        Trigger              :=FALSE;

    ELSE                                                     // Interrupted by error.
            DoTCP            :=FALSE;
            Trigger          :=FALSE;
    END_CASE;

END_IF;
```

Remote Node Programming

The processing procedure at the remote node is as follows:

1 The SktTCPAccept instruction is used to make a request to accept the connection on the TCP socket.

2 The SktTCPRcv instruction is used to make a receive request. The received data is stored in RcvSocketDat[].

3 The SktTCPSend instruction is used to make a send request The data in SendSocketDat[] is sent.

4 The SktClose instruction is used to close the socket.

ST

Internal variables	Variable	Data type	Initial value	Comment
	Trigger	BOOL	False	Execution condition
	DoTCP	BOOL	False	Processing
	Stage	INT	0	Status change
	RcvSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	Receive data

Internal variables	Variable	Data type	Initial value	Comment
	WkSocket	_sSOCKET	(Handle:=0, SrcAdr:=(PortNo:=0, IpAdr:=''), DstAdr:=(PortNo:=0, IpAdr:=''))	Socket
	SendSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	Send data
	SktTCPAccept_instance	SktTCPAccept		
	SktTCPSend_instance	SktTCPSend		
	SktTCPRcv_instance	SktTCPRcv		
	SktClose_instance	SktClose		

External variable	Variable	Data type	Constant	Comment
	_EIP_EtnOnlineSta*1	BOOL	☑	Online

*1. For an NX701 CPU Unit and an NX102 CPU Unit, replace the variable with _EIP1_EtnOnlineSta (Port1 Online) or _EIP2_EtnOnlineSta (Port2 Online), depending on the built-in EtherNet/IP port which is used.
For an NX1P2 CPU Unit, replace the variable with _EIP1_EtnOnlineSta (Port1 Online).

```
// Start sequence when Trigger changes to TRUE.
IF ((Trigger=TRUE) AND (DoTCP=FALSE) AND (_EIP_EtnOnlineSta=TRUE)) THEN
    DoTCP                    :=TRUE;
    Stage                    :=INT#1;
    SktTCPAccept_instance(Execute:=FALSE);                // Initialize instance.
    SktTCPSend_instance(        // Initialize instance.
        Execute              :=FALSE,
        SendDat              :=SendSocketDat[0]);         // Dummy
    SktTCPRcv_instance(        // Initialize instance.
        Execute              :=FALSE,
        RcvDat               :=RcvSocketDat[0]);          // Dummy
    SktClose_instance(Execute:=FALSE);                    // Initialize instance.
END_IF;

IF (DoTCP=TRUE) THEN
    CASE Stage OF
    1 :                                                   // Request to accept a s
ocket connection
        SktTCPAccept_instance(
            Execute          :=TRUE,
            SrcTcpPort       :=UINT#6000,                 // Local TCP port number
            TimeOut          :=UINT#0,                    // Timeout value
            Socket    =>WkSocket);                        // Socket

        IF (SktTCPAccept_instance.Done=TRUE) THEN
            Stage            :=INT#2;                     // Normal end
        ELSIF (SktTCPAccept_instance.Error=TRUE) THEN
```

```
            Stage           :=INT#10;                   // Error end
        END_IF;

    2 :                                                 // Receive request
        SktTCPRcv_instance(
            Execute         :=TRUE,
            Socket          :=WkSocket,                 // Socket
            TimeOut         :=UINT#0,                   // Timeout value
            Size            :=UINT#2000,                // Receive data size
            RcvDat          :=RcvSocketDat[0]);         // Receive data

        IF (SktTCPRcv_instance.Done=TRUE) THEN
            Stage           :=INT#3;                    // Normal end
        ELSIF (SktTCPRcv_instance.Error=TRUE) THEN
            Stage           :=INT#20;                   // Error end
        END_IF;

    3 :                                                 // Send request
        SendSocketDat:=RcvSocketDat;
        SktTCPSend_instance(
            Execute         :=TRUE,
            Socket          :=WkSocket,                 // Socket
            SendDat         :=SendSocketDat[0],         // Send data
            Size            :=UINT#2000);               // Send data size

        IF (SktTCPSend_instance.Done=TRUE) THEN
            Stage           :=INT#4;                    // Normal end
        ELSIF (SktTCPSend_instance.Error=TRUE) THEN
            Stage           :=INT#30;                   // Error end
        END_IF;

    4 :                                                 // Request to close the
socket
        SktClose_instance(
            Execute         :=TRUE,
            Socket          :=WkSocket);                // Socket

        IF (SktClose_instance.Done=TRUE) THEN
            Stage           :=INT#0;                    // Normal end
        ELSIF (SktClose_instance.Error=TRUE) THEN
            Stage           :=INT#40;                   // Error end
        END_IF;

    0 :                                                 // Normal end
        DoTCP               :=FALSE;
        Trigger             :=FALSE;
    ELSE                                                // Interrupted by error
```

```
        DoTCP                :=FALSE;
        Trigger              :=FALSE;
    END_CASE;

END_IF;
```

8-7 Precautions in Using Socket Services

8-7-1 Precautions for UDP and TCP Socket Services

- Communications processing are sometimes delayed when multiple functions of the built-in EtherNet/IP port are used simultaneously or due to the contents of the user program.
- Communications efficiency is sometimes reduced by high communications traffic on the network line.
- The close processing for a close request instruction discards all of the buffered send and receive data for the socket.
  For example, send data for a send request which is issued immediately before the close processing may not be sent.
- After a socket is open, the built-in EtherNet/IP port provides a receive buffer of 9,000 bytes per TCP socket and 9,000 bytes per UDP socket to enable data to be received at any time.
  If the receive buffer is full, data received by the socket is discarded. Make sure that the user application constantly issues receive requests to prevent the internal buffer from becoming full.
- If the **Use** Option is selected for Packet Filter of the built-in EtherNet/IP port, make sure to permit packets to be used for socket services. If they are not permitted, packets used by the socket services cannot be received. For the details on the settings, refer to *Packet Filter* on page 4-7.

8-7-2 Precautions for UDP Socket Services

- The destination IP address can be set to a broadcast address for a UDP socket to broadcast data to all nodes on the network.
  However, in this case, the maximum length of send data is 1,472 bytes.
  Data divided into multiple fragments (1,473 bytes or more in UDP) cannot be sent.
- UDP sockets do not perform controls intended to secure the reliability of communications, such as confirming if the send data is received. To improve the reliability of communications when you use UDP sockets, make sure the user program confirms that data is sent and resends the data when necessary.

8-7-3 Precautions for TCP Socket Services

- If the TCP socket is closed on the remote node without warning during communications (i.e., if the connection is closed), the socket at the local node must also be closed.
  You can use the Read TCP Socket Status instruction (SktGetTCPstatus) to see if the connection is closed.
  Immediately close the socket at the local node if the TCP socket at the remote node is closed.
- If the remote node's TCP socket closes without warning, the data to send may remain in the buffer at the local node. The remaining data is discarded in the local node's TCP close processing.
  The steps that are required in applications to avoid this include sending data from the sending node that permits closing and closing the socket only after checking the remote node.
- While open processing is performed for a TCP socket, a port that was closed first cannot be opened again for 60 seconds from the time the close processing is performed for the remote socket.
  However, this is not true if you specified 0 (automatic assignment by the Unit) as the port for the SktTCPConnect instruction.

- You can open a connection by performing Connect from one socket to another socket that is open with Accept. Connections cannot be opened if you attempt Connect from one socket to another socket which is open with Connect.
  Connections cannot be opened either if you attempt Accept from one socket to another socket which is open with Accept.
  Furthermore, you cannot use more than one Connect from another node to open multiple connections to a single TCP socket which is open with Accept on the build-in EtherNet/IP port.
- You can use the keep-alive function for TCP sockets at the built-in EtherNet/IP port.
  The keep alive function checks whether a connection is normally established when no data is sent or received for a certain period on the communications line where the connection was established.
  The built-in EtherNet/IP port responds to checks from other nodes even if keep alive is not specified.
- For TCP sockets, the send data is resent up to 12 times if an acknowledgment (ACK) from the remote node is not received. The resend interval increases every resend in a range from one second to 64 seconds.
- For TCP sockets, a connection request (SYN) is sent by performing an open connection. SYN is resent up to four times if an acknowledgment (SYN + ACK) from the remote node is not received. An error will occur if SYN + ACK is not received yet even after 75 seconds has elapsed since the open processing.

8-8 TCP/UDP Message Service

8-8-1 Outline of TCP/UDP Message Service

TCP/UDP message service provides a simple form of TCP/UDP socket communications intended for access to CIP objects of the Controller from a system where EtherNet/IP is not supported. With this function, you can change settings and perform I/O control for the Controller and Units connected to the NX Bus. TCP/UDP message service can be performed simultaneously with tag data link communications.

This function is available only with NX102 CPU Units.

8-8-2 Specifications of TCP/UDP Message Service

Item	Specifications
Maximum number of clients which can be connected simultaneously	32 (for UDP and TCP each)
Maximum message size	Request: 492 bytes Response: 496 bytes
Maximum NX data output size	Maximum size of NX output data which can be written with the TCP/UDP message service 488 bytes
Maximum NX data input size	Maximum size of NX input data which can be read with the TCP/UDP message service 496 bytes
Port number	Port number used in the TCP/UDP message service Default value: 64000 (decimal number)

8-8-3 Settings Required for TCP/UDP Message Service

When you use the TCP/UDP message service, you need to set the following unit settings.

The settings can be configured with the Sysmac Studio version 1.23 or higher.

Sysmac Studio Unit Settings Tab Page	Setting	Setting conditions	Setting range	Default
TCP/UDP message service	Use/Do not use TCP/UDP message service	Optional	Use/Do not use	Do not use
	Port 1-Port No.	Optional	1024-65535 *1	64000
	Port 2-Port No.	Optional	1024-65535 *1	64000

*1. When you use the TCP socket, the following port numbers are used by the system and cannot be set by the user: 20, 23, 25, 80, 110, 9610, and 44818.
When you use the UDP socket, the following port numbers are used by the system and cannot be set by the user: 25, 53, 68, 110, 2222, 2223, 2224, 9600, and 44818.

Precautions for Correct Use

If the **Use** Option is selected for Packet Filter on the built-in EtherNet/IP port, make sure to permit packets to be used for TCP/UDP message services. If they are not permitted, packets used by TCP/UDP message services cannot be received. For the details on the settings, refer to *Packet Filter* on page 4-7.

8-8-4 Command Format Specifications

Request Command

Parameter name	Offset address	Size (bytes)	Description	Example of VendorID readout*1
Sequence No.	0	2	The user specifies an arbitrary number. The number specified here is stored in the sequence No. of the response command corresponding to the request command.	1000
Reserved 1	2	2	Reserved. Specify 0.	0000
Data Size	4	2	Specify the command size after the **Reserved 2** parameter.	0800
Reserved 2	6	1	Reserved. Specify 0.	00
Service code	7	1	CIP service	0E
Class ID	8	2	Controller object class ID	0100
Instance ID	10	2	CIP object instance ID	0100
Attribute ID	12	2	CIP object attribute ID. Specify if attribute ID specification is required in the specified CIP service This can be omitted if such specification is not required.	0100
Data	12*2	Maximum 492*3	Specify request data.	---

*1. Hexadecimal data in little-endian format.
*2. The offset address will be 14 if the attribute ID is specified.
*3. The size will be 488 bytes if the attribute ID is specified.

Response Command

Parameter name	Offset address	Size (bytes)	Description	Example of VendorID readout*1
Sequence No.	0	2	This is the sequence number specified in the request command corresponding to the response command.	1000
Data Size	2	2	The command size after the **Reserved** parameter is stored.	0600
Reserved	4	1	Reserved. 0 is stored.	00
Service code	5	1	The executed service code + most significant bit 1 is stored.	8E

Parameter name	Offset address	Size (bytes)	Description	Example of VendorID readout*1
General status	6	1	00 is stored when the service ends normally, and a value other than 00 is stored when the service ends in error. Status codes stored when an error occurs conform to the CIP General Status Code.	00
Additional status size	7	1	00 is stored when the service ends normally. If the service ends in error, the Additional status size (word size) stored in the **Data** area will be stored.	00
Data	8	Maximum 496	The response data is stored when the service ends normally. If the service ends in error, the Additional status will be stored for the word size stored in the **Additional status size** parameter.	2F00

*1. Hexadecimal data in little-endian format.

8-9 Secure Socket Services

The secure socket services perform encrypted secure socket communications (hereinafter called “secure socket communications”) using TLS (Transport Layer Security).
The CPU Unit can be used as a client to connect to cloud and on-premises servers via TCP/IP and exchange messages.

Version Information

An NX102-□□00 CPU Unit with unit version 1.46 or later or an NX102-□□20 CPU Unit with unit version 1.37 or later and Sysmac Studio version 1.46 or higher are required to use the secure socket services.
An NX1P2-□□□□□□ CPU Unit with unit version 1.46 or later and Sysmac Studio version 1.46 or higher are required to use secure socket services.

Additional Information

Function Blocks (FBs) for MQTT communications are available for the secure socket communications between a CPU Unit and a MQTT broker.
Refer to the *Sysmac Library User’s Manual for MQTT Communications Library (Cat. No. W625)* for more information on FBs for MQTT communications.

8-9-1 Overview of Secure Socket Communications

Secure socket communications use TLS1.2 to encrypt communication data between the client and the server. By encrypting communication data, you can prevent third parties from eavesdropping or tampering with the data.
Client authentication also allows the server to detect client spoofing.

Client Authentication

In secure socket communications, client authentication, which allows only certain clients to access the server, is supported at the same time as encryption of communication data.
Using client certificates and client private keys, only devices with client certificates can establish TLS sessions with the server.
Request a signature from the Certification Authority (CA) to obtain the CA certificate to confirm the validity of the client certificate.
Client authentication allows you to operate a more secure system.

Precautions for Correct Use

- Determine the need for client authentication by taking into conditions such as the specifications, operating costs, and security policies of the server.
- Network security issues such as the server data be illegally obtained or tampered, or communications to the server be disabled may occur due to theft, information leaks and tampering of client certificates, private keys and secure socket setting by third parties. Take necessary measures for the management of client certificates, private keys and secure socket setting and for the prevention of theft, information leaks and tampering of those.
  Especially, use an encrypted safe communications path, etc. when obtaining the private key to avoid information leaks. Furthermore, store the private key in a safe location where the risk of information leakage is extremely low.

Additional Information

You can obtain the client certificate and client private key in the following ways.

a. Request to issue a certificate to the Certification Authority.
b. Create client certificates and client private keys by using OpenSSL or other tool.
   Create X.509 digital certificates with Base64 Encode (convert to Pem format).
c. Use an external certificate creation service.

Outline of Secure Socket Communications Processing Procedure

The outline of processing procedure of secure socket communications is as follows.

Client (CPU Unit)		Server
	TCP connection request ---------->	
	TLS session connection request ---------->	
	Sending the server certificate <----------	
	Client certificate request (when the server performs client authentication) <----------	
	Sending client certificate and client private key (when the server performs client authentication) ---------->	
	Determining cipher suite*1 <---------->	
	Establishment of TLS session <---------->	
	Encrypted data communications <---------->	

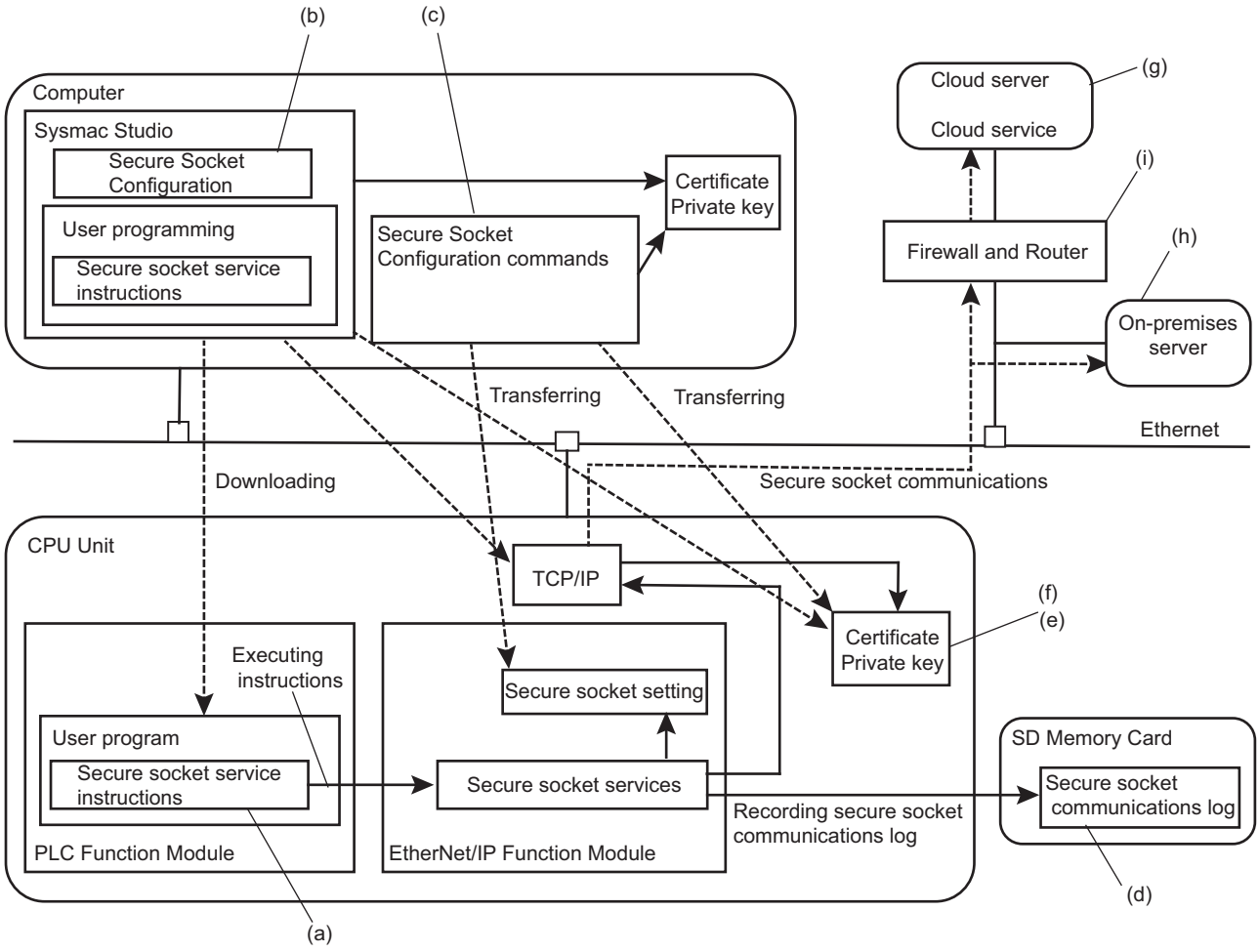
*1. A cipher suite is a set of key exchange algorithm, key authentication method, encryption method and message authentication code.

Precautions for Correct Use

Server certificates are used only to encrypt communications. It is not necessary to set the server certificate or CA certificate on the CPU Unit.

8-9-2 System Configuration of Secure Socket Services

The system configuration for performing the secure socket communications is shown below.



The system components are described in the following table.

	Component	Description
(a)	Secure socket service instructions	CPU Unit instructions that perform secure socket communications
(b)	Secure socket setting*1	A Sysmac Studio function to configure secure socket setting in a CPU Unit (such as transferring client certificates and private keys, and enabling or disabling secure socket communications log, etc.)
(c)	Secure Socket Configuration commands*2	A command-line tool to configure secure socket setting in a CPU Unit (such as transferring client certificates and private keys, and enabling or disabling secure socket communications log, etc.)
(d)	Secure socket communications logs	Logs of secure socket communications TLS session parameters, starting and ending of a TLS session, and communications error information are output as a log.
(e)	Certificate	A client certificate and a client private key used by a server for client (a CPU Unit) authentication. The certificate and the private key are transferred to a CPU Unit using the Secure Socket Configuration commands on the computer.
(f)	Private key	
(g)	Cloud server	A server that provides cloud services on an external network.
(h)	On-premises server	A server installed in your own facility.

	Component	Description
(i)	Firewall and Router	Communication devices that relay between different networks, such as a cloud server on an external network.

*1. An NX102 CPU Unit or NX1P2 CPU Unit with unit version 1.60 or later and Sysmac Studio version 1.53 or higher are required to use the settings.

*2. Use the commands for an NX102 CPU Unit or NX1P2 CPU Unit with unit version 1.50 or earlier.

Precautions for Correct Use

- Setting up an intranet through a global address involves network security considerations. Be sure to consult with a network specialist in advance and consider using a VPN or installing a firewall. After a firewall is set up by a communications technician, there may be some applications that cannot be used. Be sure to check first with the communications technician.
- To reduce the risk of unauthorized access by a third party using the Secure Socket Configuration commands, consider setting operation authority verification on the CPU Unit. You can restrict the use of Secure Socket Configuration commands to administrators only.
  For details on how to set operation authority verification, refer to "Operation Authority Verification" on the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)*.
  Refer to *Operation Authority Verification* on page A-75 for operating specifications of Secure Socket Configuration commands when operation authority verification is set.

8-9-3 Procedure to Use Secure Socket Setting Function of the Sysmac Studio

This section describes the procedure to use secure socket services for the following use cases.

- Starting to use secure socket services
  Refer to *Settings for Starting Secure Socket Services* on page 8-39.
- Replacing CPU Units
  Refer to *Procedure for Replacing the CPU Unit* on page 8-42.

The setting method of the secure socket service depends on the unit version of the CPU Unit as shown below.

Unit version	Setting method
Ver.1.50 or earlier	• Use the Secure Socket Configuration commands. *1 • If the unit version is 1.50, it is also necessary to set *Enabling connections to the Sysmac Studio and NA that are not supporting secure communication*.
Ver.1.60 or later	Use the secure socket setting function on the Sysmac Studio. *2

*1. Refer to *A-9 Procedure to Use Secure Socket Service with Secure Socket Configuration Commands* on page A-66 for details on how to use the secure socket service with the Secure Socket Configuration commands.

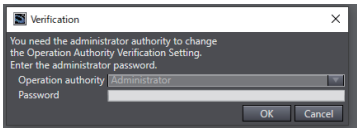
*2. Use the Sysmac Studio version 1.53 or higher.

When user authentication or operation authority verification is set, only *Administrator* can use the secure socket setting function.
Secure socket setting can be set only when the operating mode is PROGRAM mode. If the operating mode is RUN mode, change to PROGRAM mode before the settings.
The secure socket setting with the Sysmac Studio are as follows.
Refer to the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for details on the operations on the Sysmac Studio.

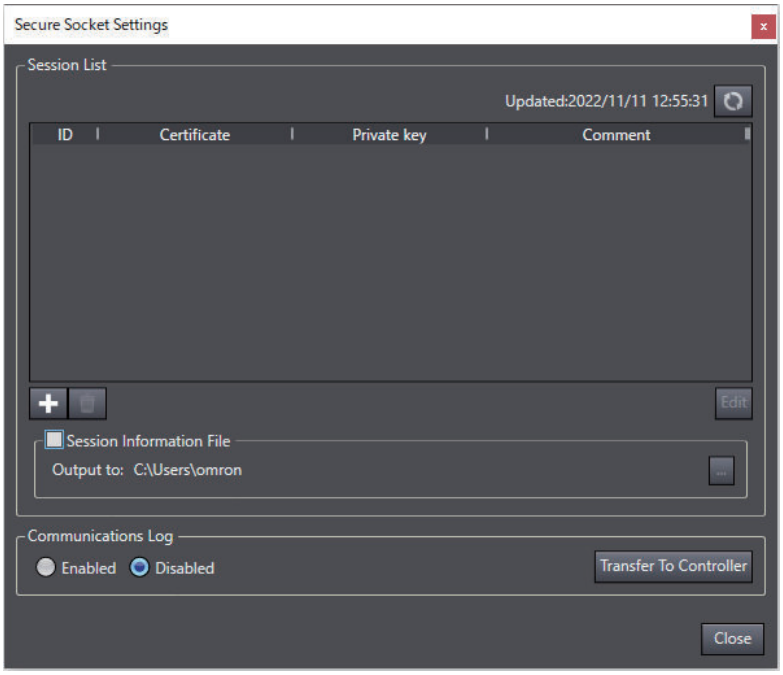
1 Select **Controller** - **Security** - **Secure Socket Settings** on the Sysmac Studio.

If user authentication is set, the following **Authentication** Dialog Box is displayed.

If operation authority verification is set, the following **Verification** Dialog Box is displayed.



2 Enter the *Administrator* password authenticated when connecting online, and click the **OK** Button.
After authentication is completed, the **Secure Socket Settings** Dialog Box is displayed.



Settings for Starting Secure Socket Services

The following two procedures describe how to set up a new configuration.

- If you do not use a client certificate and a client private key
- If you use a client certificate and a client private key

● If you do not use a client certificate and a client private key

The setting procedure to start secure socket services when the client certificate and client private key are not used is as follows.

As a prerequisite, set the built-in EtherNet/IP of the CPU Unit as follows.

- If the server is on the Internet, configure the default gateway and routing table.
  If the server is specified by an item other than the IP address, such as "xxx.com", configure the DNS server settings.

- Configure NTP Settings.
  The NTP Settings are optional. It is recommended for matching with the server time.

Check with the network administrator of the installation site for the settings of the default gateway, routing table, DNS server, and NTP server.

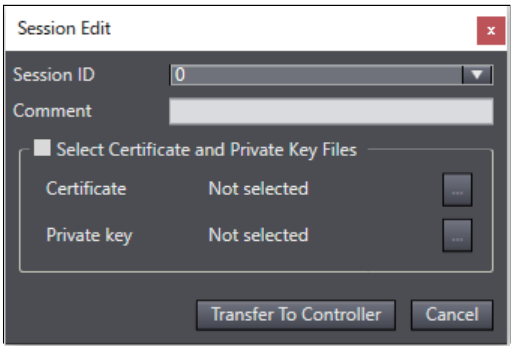
The secure socket setting in this procedure is described in the following example.

- The session ID set in the secure socket setting is 0.

1 Configure the server and check the server's IP address, HOST name, and other settings.
Check with the server installer for details on how to check.

2 Configure the secure socket setting.
Use the Sysmac Studio to configure secure socket setting for the session ID. Set different session IDs for all connected destinations.

1) Connect the Sysmac Studio online, and select **Controller** - **Security** - **Secure Socket Settings**.
2) Press the **+** Button in the **Session List** of the **Secure Socket Settings** Dialog Box.
   The **Session Edit** Dialog Box is displayed.
3) Select 0 for **Session ID** and enter the session comment if necessary.
4) Clear the **Select Certificate and Private Key Files** Check Box.
5) Click the **Transfer to Controller** Button to transfer the settings to the Controller.



To enable secure socket communications log, select **Communications Log** to **Enabled** in the **Secure Socket Settings** Dialog Box and click the **Transfer to Controller** Button.

3 Create a user program.
Create a session for secure socket communications with SktTCPConnect instruction to the server in step 1. Set the TLS session name for the session ID to *TLSSessionName*, which is the input variable of SktTLSConnect instruction. If the session ID in the **Session Edit** Dialog Box is *0*, the TLS session name is *TLSSession0*.

Use SktTLSRead and SktTLSWrite instructions to process data communications with the server.

4 Download the user program using the synchronization function.
Download the user program from the computer to the CPU Unit.
After sufficiently confirming that the connection destination is correct, start operation.

● If you use a client certificate and a client private key

The setting procedure to start secure socket services when the client certificate and client private key are used is as follows.

As a prerequisite, set the built-in EtherNet/IP of the CPU Unit as follows.

- If the server is on the Internet, configure the default gateway and routing table.
  If the server is specified by an item other than the IP address, such as "xxx.com", configure the DNS server settings.
- Configure NTP settings.
  The NTP settings are optional. It is recommended for matching with the server time.

Check with the network administrator of the installation site for the settings of the default gateway, routing table, DNS server, and NTP server.

The secure socket setting in this procedure is described in the following example.

- To connect the computer to the CPU Unit, an EtherNet/IP port is used. They are connected through Ethernet connection via a Hub or remote connection via USB.
- The IP address of the built-in EtherNet/IP port of the CPU Unit is set to 192.168.250.1.
- The session ID set in the secure socket setting is 0.

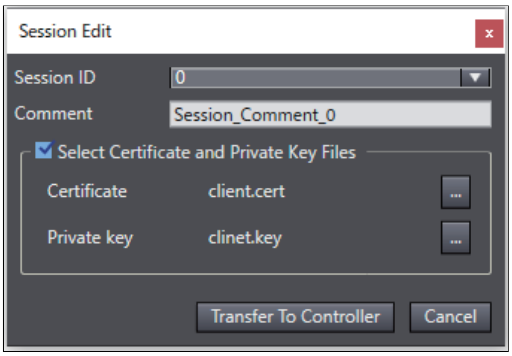
1 Prepare the client certificate, client private key, and CA certificate.
In this procedure, the file name of the prepared client certificate is *client.cert*. The file name of the client private key is *client.key*.
Note that the prepared client certificate and client private key must be stored and managed by the customer.

2 Install the client certificate and CA certificate on the server.
Check with the server administrator for details such as whether installation on the server is required.

3 Configure the server and check the server's IP address, HOST name, and other settings.
Check with the server installer for details on how to check.

4 Configure the secure socket setting.
Use the Sysmac Studio to configure session information for the session ID.

1) Press the **+** Button in the **Session List** of the **Secure Socket Settings** Dialog Box.
   The **Session Edit** Dialog Box is displayed.
2) Select 0 for **Session ID** and enter the session comment if necessary.
3) Select the **Select Certificate and Private Key Files** Check Box.
4) Click the buttons to display the file selection dialog box for **Certificate** and **Private key** and select the client certificate file *client.cert* and client private key file *client.key* respectively.
5) Click the **Transfer to Controller** Button to transfer the settings to the Controller.



To enable secure socket communications log, select **Communications Log** to **Enabled** in the **Secure Socket Settings** Dialog Box and click the **Transfer to Controller** Button.

5 Create a user program.
Create a session for secure socket communications with SktTCPConnect instruction to the server in step 3. Set the TLS session name for the session ID to *TLSSessionName*, which is the input variable of SktTLSConnect instruction. If the session ID in the **Session Edit** Dialog Box is *0*, the TLS session name is *TLSSession0*.

Use SktTLSRead and SktTLSWrite instructions to process data communication with the server.

6 Download the user program using the synchronization function.
Download the user program from the computer to the CPU Unit.
After sufficiently confirming that the connection destination is correct, start operation.

Procedure for Replacing the CPU Unit

This section describes the following three procedures for replacing the CPU Unit.
- If you do not use a client certificate and a client private key
- If you have stored the client certificate and client private key
- If you have not stored the client certificate and client private key

When you replace the CPU Unit, be sure to perform the following steps before proceeding to the replacement procedure.
Refer to the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for details on the operations on the Sysmac Studio.
The secure socket setting in this procedure is described in the following example.
- The session ID set in the secure socket setting is 2.
- The folder to save the secure socket setting is *C:\Users\omron*.

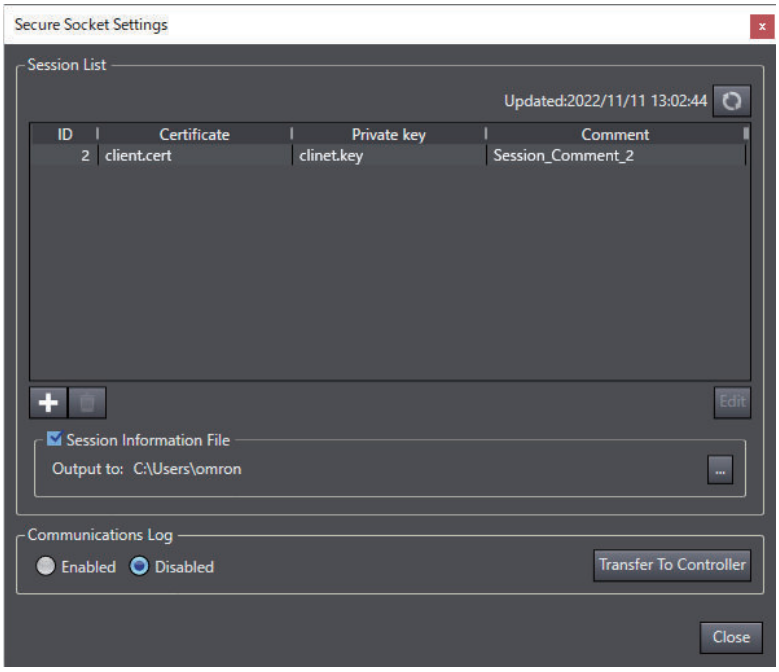
1 Back up the data in the Controller.
Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for details on Controller backups.

2 Read the secure socket setting.
Display the **Secure Socket Settings** Dialog Box and save the secure socket setting.

1) Select **Session Information File** Check Box in the **Secure Socket Settings** Dialog Box.

2) Click the folder selection button and select the folder to output the session information file.
The folder that you select to **Output to:** is displayed.

3) Click the  Button.
The session information file is output to the selected folder.



Check the status of **Communications Log** (Enabled or Disabled) in the **Secure Socket Settings** Dialog Box.

3 Check that the client certificate and client private key are stored.
Check the read secure socket setting to ensure that the required client private key is stored.

● If you do not use a client certificate and a client private key

The procedure for replacing the CPU Unit when the client certificate and client private key are not used is as follows.

The secure socket setting in the replacement procedure is described in the following example.
- The session ID in the secure socket setting before replacement is set to 2.

1 Replace to a new CPU Unit.

2 Check the secure socket setting.
Use the secure socket setting to check the session ID that is being used before replacing the CPU Unit.

3 Configure the secure socket setting.

1) Connect the Sysmac Studio online, and select **Controller** - **Security** - **Secure Socket Settings**.
2) Press the **+** Button in the **Session List** of the **Secure Socket Settings** Dialog Box.
The **Session Edit** Dialog Box is displayed.

3) Select 2 for **Session ID** and enter the session comment if necessary.
4) Clear the **Select Certificate and Private Key Files** Check Box.
5) Click the **Transfer to Controller** Button to transfer the settings to the Controller.

4 Check the secure socket setting.
Display the **Secure Socket Settings** Dialog Box and verify that it matches the session ID set in the folder of **Output to:** read in step 2 of *Procedure for Replacing the CPU Unit* on page 8-42.

Check the status of **Communications Log** (Enabled or Disabled) in the **Secure Socket Settings** Dialog Box.

5 Restore data to the Controller.
Restore is performed using the backed up data.
Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for details on restoration on the Controller.

6 Check the operation.
Verify that the program and settings are restored and the Controller is working correctly.

● If you have stored the client certificate and client private key

The procedure for replacing the CPU Unit when the client certificate and client private key have been stored is as follows.

The secure socket setting in the replacement procedure is described in the following example.
- The session ID in the secure socket setting before replacement is set to 2.
- The file name in the computer that stores the client certificate file used in the secure socket setting of session ID=2 is *client.cert*.
- The file name in the computer that stores the client private key file used in the secure socket setting of session ID=2 is *client.key*.

1 Replace to a new CPU Unit.

2 Check the secure socket setting.
Use the secure socket setting to check the session ID that is being used before replacing the CPU Unit.
Prepare the client certificate and client private key for each session ID that are stored in the computer.

3 Configure the secure socket setting.
1) Press the **+** Button in the **Session List** of the **Secure Socket Settings** Dialog Box.
The **Session Edit** Dialog Box is displayed.
2) Select 2 for **Session ID** and enter the session comment if necessary.
3) Select the **Select Certificate and Private Key Files** Check Box.
4) Click the buttons to display the file selection dialog box for **Certificate** and **Private key** and select the client certificate file *client.cert* and client private key file *client.key* respectively.

5) Click the **Transfer to Controller** Button to transfer the settings to the Controller.

To enable secure socket communications log, select **Communications Log** to **Enabled** in the **Secure Socket Settings** Dialog Box and click the **Transfer to Controller** Button.

4 Check the secure socket setting.
Display the **Secure Socket Settings** Dialog Box and verify that it matches the session ID set in the folder read in step 2 of *Procedure for Replacing the CPU Unit* on page 8-42 (*C:\Users \omron* in this example).

Check the status of **Communications Log** (Enabled or Disabled) in the **Secure Socket Settings** Dialog Box.

5 Restore data to the Controller.
Restore is performed using the backed up data.
Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for details on restoration on the Controller.

6 Check the operation.
Verify that the program and settings are restored and the Controller is working correctly.

● If you have not stored the client certificate and client private key

The procedure for replacing the CPU Unit when the client certificate and client private key have not been stored is as follows.

1 Create a client certificate and client private key.
Depending on whether you are creating a client certificate and client private key on the server or preparing the client private key and client certificate yourself, the procedures are different as follows.

Creating a client certificate and client private key on the server

1) Create a client certificate and client private key on the server and download them to the computer.
In this procedure, the file name of the downloaded client certificate is *client.cert*. The file name of the client private key is *client.key*.
Note that you must store and manage the downloaded client certificate and client private key yourself.

Creating a client certificate and client private key yourself

1) Prepare the client certificate, client private key, and CA certificate.
In this procedure, the file name of the prepared client certificate is *client.cert*. The file name of the client private key is *client.key*.
Note that the prepared client certificate, client private key, and CA certificate must be stored and managed by the customer.
2) Install the client certificate and CA certificate on the server.
Check with the server administrator for details such as whether installation on the server is required.

2 Check the secure socket setting.

Use the secure socket setting to check the session ID that is being used before replacing the CPU Unit.
Prepare the client certificate and client private key for each session ID that are stored in the computer.

3 Configure the secure socket setting.

1) Press the **+** Button in the **Session List** of the **Secure Socket Settings** Dialog Box.
   The **Session Edit** Dialog Box is displayed.
2) Select 2 for **Session ID** and enter the session comment if necessary.
3) Select the **Select Certificate and Private Key Files** Check Box.
4) Click the buttons to display the file selection dialog box for **Certificate** and **Private key** and select the client certificate file *client.cert* and client private key file *client.key* respectively.
5) Click the **Transfer to Controller** Button to transfer the settings to the Controller.

To enable secure socket communications log, select **Communications Log** to **Enabled** in the **Secure Socket Settings** Dialog Box and click the **Transfer to Controller** Button.

4 Restore data to the Controller.
Restore is performed using the backed up data.
Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for details on restoration on the Controller.

5 Check the operation.
Verify that the program and settings are restored and the Controller is working correctly.

8-9-4 Executing Instructions for Secure Socket Communications

You can execute the secure socket communications using the socket service instructions and secure socket service instructions.

Secure Socket Service Instructions

The following table lists all of the secure socket service instructions.

Instruction	Function
SktTLSConnect	Establish TLS Session
SktTLSWrite	Send TLS
SktTLSRead	Receive TLS
SktTLSClearBuf	Clear TLS Session Receive Buffer
SktTLSDisconnect	Disconnect TLS Session
SktTLSStopLog	Stop Secure Socket Communications Log

Additional Information

Specify the TLS session name of the TLS session information that is set on the Sysmac Studio or with Secure Socket Configuration commands for the input variable of SktTLSConnect instruction. Refer to *A-10 Secure Socket Configuration Commands* on page A-73 for details on the Secure Socket Configuration commands.

Refer to the *NJ/NX-series Instructions Reference Manual (Cat. No. W502)* for details on the secure socket service instructions.

Instruction Execution Flow for Secure Socket Communications

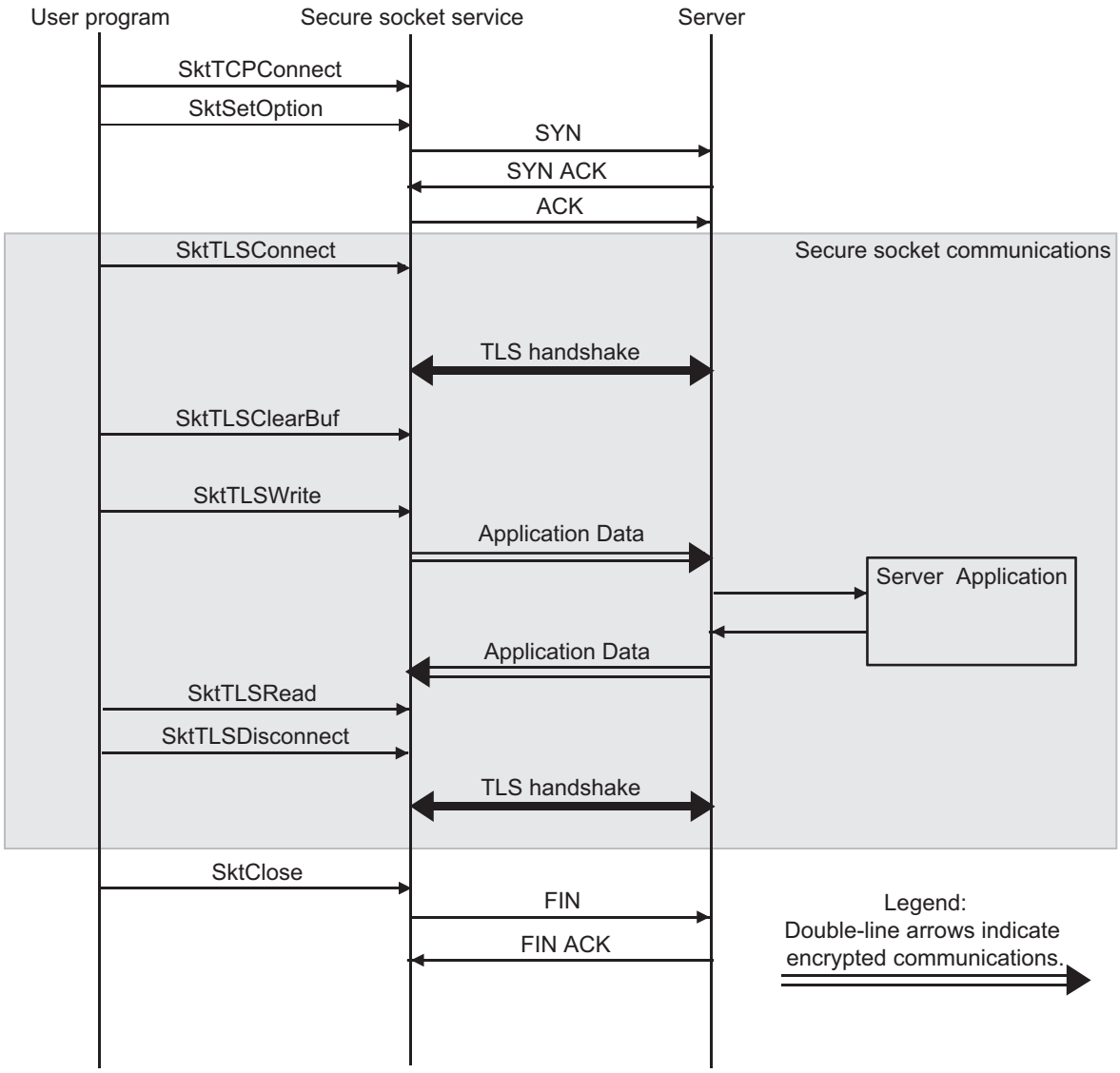
The instruction execution flow for secure socket communications is as follows.

1. Use SktTCPConnect instruction to connect to the destination TCP port and create a socket.
2. Set the socket with SktSetOption instruction as required.
3. SktTLSConnect instruction opens a session between the server and TLS.
4. The receive buffer is cleared by SktTLSClearBuf instruction, and communication with the server is performed using SktTLSWrite or SktTLSRead instructions.
5. When the communications with the server are completed, terminate the TLS session with SktTLSDisconnect instruction and close the socket with the SktClose instruction.

Precautions for Correct Use

The number of TLS sessions that can be established in the secure socket communications is equal to the number of sockets that you can use in the TCP socket service. Therefore, it is shared with sockets used by normal socket service. Refer to *Overview of Socket Services with Socket Service Instructions* on page 8-10 for the number of sockets that you can use for the TCP socket service.

The following diagram shows the exchanges with the server in secure socket communications by the execution of instructions on the CPU Unit.

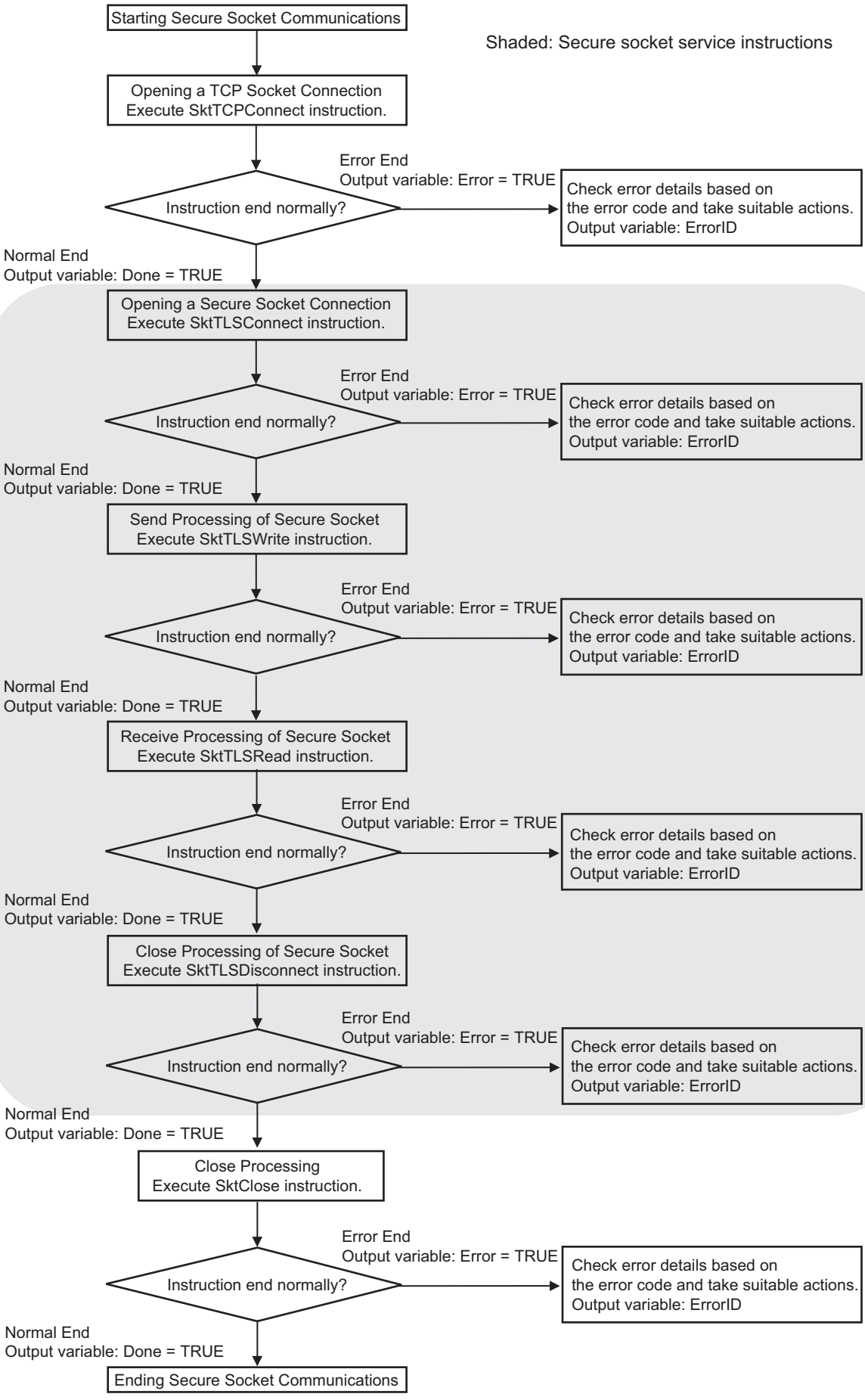


TLS Handshake exchanges and verifies the data (such as certificates) required for encrypted communications.

Troubleshooting Secure Socket Service Instructions

This section describes how to identify errors when secure socket service instructions are executed and how to confirm the error details for troubleshooting when instructions ended in error.
Check the values of the output variables of each instruction to confirm whether the execution of instruction ended normally. Refer to the *NJ/NX-series Instructions Reference Manual (Cat. No. W502)* for values of output variables of each instruction.
Furthermore, for secure socket service instructions, you can find more detailed error information from the secure socket communications log than from the ErrorID output variable for the instruction which is referenced in error end. Refer to *8-9-6 Secure Socket Communications Logging* on page 8-50 for details on the secure socket communications log.

The diagram below shows the troubleshooting flow when instructions to perform secure socket communications, which also include socket service instructions, are executed.


Starting Secure Socket Communications
Shaded: Secure socket service instructions
Opening a TCP Socket Connection
Execute SktTCPConnect instruction.
Error End
Output variable: Error = TRUE
Instruction end normally?
Check error details based on
the error code and take suitable actions.
Output variable: ErrorID
Normal End
Output variable: Done = TRUE
Opening a Secure Socket Connection
Execute SktTLSConnect instruction.
Error End
Output variable: Error = TRUE
Instruction end normally?
Check error details based on
the error code and take suitable actions.
Output variable: ErrorID
Normal End
Output variable: Done = TRUE
Send Processing of Secure Socket
Execute SktTLSWrite instruction.
Error End
Output variable: Error = TRUE
Instruction end normally?
Check error details based on
the error code and take suitable actions.
Output variable: ErrorID
Normal End
Output variable: Done = TRUE
Receive Processing of Secure Socket
Execute SktTLSRead instruction.
Error End
Output variable: Error = TRUE
Instruction end normally?
Check error details based on
the error code and take suitable actions.
Output variable: ErrorID
Normal End
Output variable: Done = TRUE
Close Processing of Secure Socket
Execute SktTLSDisconnect instruction.
Error End
Output variable: Error = TRUE
Instruction end normally?
Check error details based on
the error code and take suitable actions.
Output variable: ErrorID
Normal End
Output variable: Done = TRUE
Close Processing
Execute SktClose instruction.
Error End
Output variable: Error = TRUE
Instruction end normally?
Check error details based on
the error code and take suitable actions.
Output variable: ErrorID
Normal End
Output variable: Done = TRUE
Ending Secure Socket Communications

8-9-5 Troubleshooting Errors in Secure Socket Communications

1. Use Sysmac Studio on the computer to check the event log of the CPU Unit.
2. Check the secure socket communications log in the SD Memory Card in an editor of the computer.
   Refer to *8-9-6 Secure Socket Communications Logging* on page 8-50 for details on the secure socket communications log.
   To check the error details in the secure socket communications log, enable the secure socket communications log in the secure socket setting beforehand.
3. Identify error causes from the event log and secure socket communications log and take required measures.

8-9-6 Secure Socket Communications Logging

You can record communications logs of secure socket communications.
This log records parameters, starting and ending of a TLS session, and communications error information.
The secure socket communications log file is recorded in the SD Memory Card and you can use this log file for troubleshooting, etc., by viewing it in an editor.

How to Start and Stop Secure Socket Communication Log Output

- How to start
  Enable the secure socket communications log in the **Secure Socket Settings** Dialog Box or with the Secure Socket Configuration commands.
- How to stop
  Disable the secure socket communications log in the **Secure Socket Settings** Dialog Box or with the Secure Socket Configuration commands.
  Or, execute SktTLSStopLog instruction.

Refer to *8-9-3 Procedure to Use Secure Socket Setting Function of the Sysmac Studio* on page 8-38 for details on how to make secure socket settings on the Sysmac Studio.

Refer to *A-10 Secure Socket Configuration Commands* on page A-73 on how to set the Secure Socket Configuration commands.
Refer to the *NJ/NX-series Instructions Reference Manual (Cat. No. W502)* for details on the SktTLSStopLog instruction.

Precautions for Correct Use

Stop the output of secure socket communications log before interrupting power to the CPU Unit. If it is not stopped, the file of secure socket communications log may be corrupted.

File Types and Record Formats of Secure Socket Communications Log

You can find the file types, file names, file storage directories, and record formats of secure socket communications log in the following tables.

● Log File Types

The file name and storage directory for each file type are described in the table below.

File type	File name	Storage directory (in the SD Memory Card)	Remarks
Recording log file	SecureSocket.log	/fs/MEMCARD1/SecureSocket/Log/	Recording log file is renamed to a past log file name when the maximum number of records is reached in the recoding log file.
Past log file	SecureSocket_YYYYMMDDHHmmssSSS.log YYYY: Year, MM: Month, DD: Day, HH: Hour, mm: Minutes, ss: Seconds, SSS: Milliseconds If all the input digits are not filled, it is complemented by inputting “0”. E.g. SecureSocket_20170724220915040.log		The oldest file is deleted when the next file is created if the number of log files reaches the maximum number of files.
System file	SecureSocket.fjc	/fs/MEMCARD1/SecureSocket/System/	Log file management file

Precautions for Correct Use

If the recording log file (SecureSocket.log) or the log file management file (SecureSocket.fjc) in the SD Memory Card is deleted during operation, the secure socket communications log is not recorded.

● Number of Log Data and Formats

A log file stores the maximum of 12,000 records.

The parameters and corresponding formats contained in one record are listed in the table below.

The maximum size of one record is 256 bytes.

Parameter	Size	Format
Serial number	1 to 5 bytes	0 to 65535
Date	10 bytes (fixed)	Year, month, day YYYY-MM-DD
Time of day	8 bytes (fixed)	Hour, minutes, seconds hh:mm:ss
Milliseconds	3 bytes (fixed)	3-digit decimal integer (000 to 999) E.g. 10 msec: 010, 623 msec: 623

Parameter	Size	Format
Category	16 bytes max. (variable)	Category
Log code	4 bytes (fixed)	Unique identifying code within a category 4-digit decimal code (zero padding)
Log name	32 bytes max. (variable)	Name indicating the meaning of log
Detailed information	168 bytes max. (variable)	Detailed information of log Information is separated with a tab when multiple information is provided.
CR+LF	2 bytes (fixed)	---

● Detailed Information of Log Data

Category	Log code (decimal)	Log name	Definition	Detailed information
INFO	1000	Parameter	TLS session parameter HOST, PORT	HOST=[host name or ip address] <tab> PORT=[port] Remarks HOST: Destination host name or IP address PORT: Destination port number
	1001	Parameter	TLS session parameter CAFile	CAFile=[root certificate of server] CAFile: File name of CA-signed server certificate
	1002	Parameter	TLS session parameter CERT	CERT=[session name]/[client certificate file name] Example. CERT=TLSSession0/client.crt
	1003	Parameter	TLS session parameter KEY	KEY=[session name]/[client private key file name] Example. KEY=TLSSession0/client.key
	1010	Established	TLS session established	None
	1011	Disconnect	TLS session terminated	None
ERROR	5000	SessionFail	TLS session error	API=[API name]<tab>Code=[Error Code]<tab>[Message]
	5001	Timeout	Timeout in secure socket communications	None
	5002	CommError	Communications error	[message]
	5103	ClientCertificateError	Client certificate error	FILE=[session name]/[file name] Example. FILE=TLSSession0/client.crt
	5104	ClientPrivateKeyError	Client private key error	FILE=[session name]/[file name] Example. FILE=TLSSession0/client.key

● Example of Log Data

This is an example of log data output to the log file.

```
0 2021-06-14 16:30:48 000 INFO 1000 Parameter HOST=192.168.250.40 PORT=8883
1 2021-06-14 16:30:48 002 INFO 1001 Parameter CAFile=none
2 2021-06-14 16:30:48 002 INFO 1002 Parameter CERT=TLSSession0/server.crt
3 2021-06-14 16:30:48 005 INFO 1003 Parameter KEY=TLSSession0/server.key
4 2021-06-14 16:30:48 024 INFO 1010 Established
```

8-9-7 Handling of Secure Socket Communications Setting Information

The following table shows whether each setting information of secure socket communications is supported for synchronization (transfer), backup and restoration or Clear All Memory operation.

No: Not applicable.

Secure socket communications setting	Operation				
	Synchronization from Sysmac Studio (transfer)	Backup	Restoration		Clear All Memory operation from Sysmac Studio
		• SD Memory Card backups • Sysmac Studio Controller backups	• SD Memory Card Backups • Sysmac Studio Controller backups	• Automatic transfers from SD Memory Card • Sysmac Studio Controller backups	
Secure socket setting	No	No	No	No	Not cleared *1
Client certificate	No	No	No	No	Not cleared *1
Client private key	No	No	No	No	Not cleared *1
Secure socket communications log	No	No	No	No	Not cleared

*1. Use the **Secure Socket Settings** Dialog Box on the Sysmac Studio or the Secure Socket Configuration commands to clear the settings.

Precautions for Correct Use

- The client certificate and client private key that are related to the secure socket communications are information attached to the CPU Unit itself, therefore, the information is out of the target of backup and restoration.
  When you replace the hardware of the CPU Unit, use the **Secure Socket Settings** Dialog Box on the Sysmac Studio or the Secure Socket Configuration commands to transfer the client certificate, private key, and secure socket setting to the CPU Unit.
  Similarly, the secure socket setting is also not the backup and restoration target. Use the **Secure Socket Settings** Dialog Box on the Sysmac Studio or the Secure Socket Configuration commands to make settings to the CPU Unit.
- Network security issues such as the server data be illegally obtained or tampered, or communications to the server be disabled may occur due to theft, information leaks and tampering of client certificates, private keys and secure socket setting by third parties. Take necessary measures for the management of client certificates, private keys and secure socket setting and for the prevention of theft, information leaks and tampering of those.
  Especially, use an encrypted safe communications path, etc. when obtaining the private key to avoid information leaks. Furthermore, store the private key in a safe location where the risk of information leakage is extremely low.
- It is not possible to clear client certificates, private keys, and secure socket setting information on secure socket communications by Clear All Memory operation from the Sysmac Studio. To clear the information on secure socket communications, for example when discarding a CPU Unit, use the **Secure Socket Settings** Dialog Box on the Sysmac Studio, select and execute **Erase the data completely** of the Clear All Memory option, or use the Secure Socket Configuration commands.

Additional Information

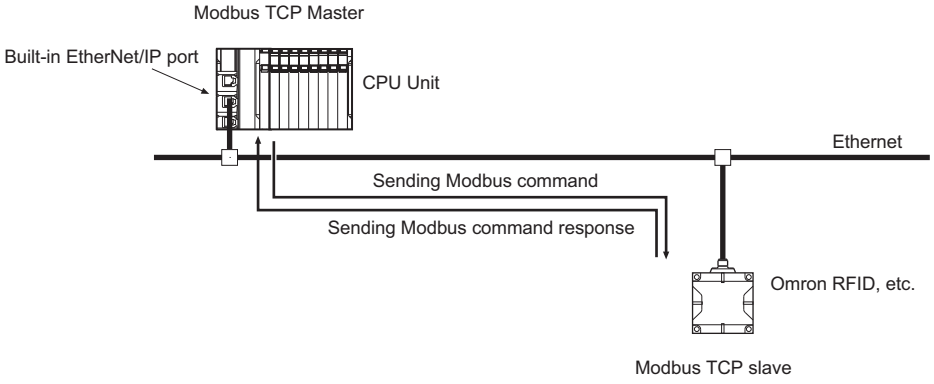
Secure socket communications log is out of the target of backup and restoration.
If you want to carry over the contents of the secure socket communications log after the CPU Unit replacement, mount the SD Memory Card that was in use in the previous Unit to the restored CPU Unit.

9

Modbus TCP Master Function

9-1 Overview of Modbus TCP Master Function

The Modbus TCP is a protocol for using the message of the Modbus protocol on Ethernet. The Modbus TCP Master function sends Modbus commands to the Modbus TCP slave and receives responses from the Modbus TCP slave.



9-2 Modbus TCP Master Function Details

The Modbus TCP Master Function can be used by executing Modbus TCP instructions in the user program.

9-2-1 Modbus TCP Instruction Type

The Modbus TCP instruction type and function are as follows.

Instruction	Function
ModbusTCPCmd	Sends general commands to the Modbus TCP slave and receives responses.
ModbusTCPRead	Sends read commands to the Modbus TCP slave and receives responses.
ModbusTCPWrite	Sends write commands to the Modbus TCP slave and receives responses.

For details on Modbus TCP instructions, refer to the *NJ/NX-series Instructions Reference Manual (Cat. No. W502)*.

9-2-2 Modbus TCP Instruction Function

This section describes Modbus TCP instruction functions.

Instruction	Function
ModbusTCPCmd	The ModbusTCPCmd instruction sends Modbus commands of the specified protocol data unit (PDU) to the specified Modbus TCP slave and receives responses.
ModbusTCPRead	The ModbusTCPRead instruction sends read commands to the specified Modbus TCP slave and receives responses. The following four Modbus commands can be sent by the ModbusTCPRead instruction. • Output read • Input read • Retained register read • Input register read
ModbusTCPWrite	The ModbusTCPWrite instruction sends write commands to the specified Modbus TCP slave and receives responses. The following four Modbus commands can be sent by the ModbusTCPWrite instruction. • One output write • One retained register write • Multiple output write • Multiple retained register write

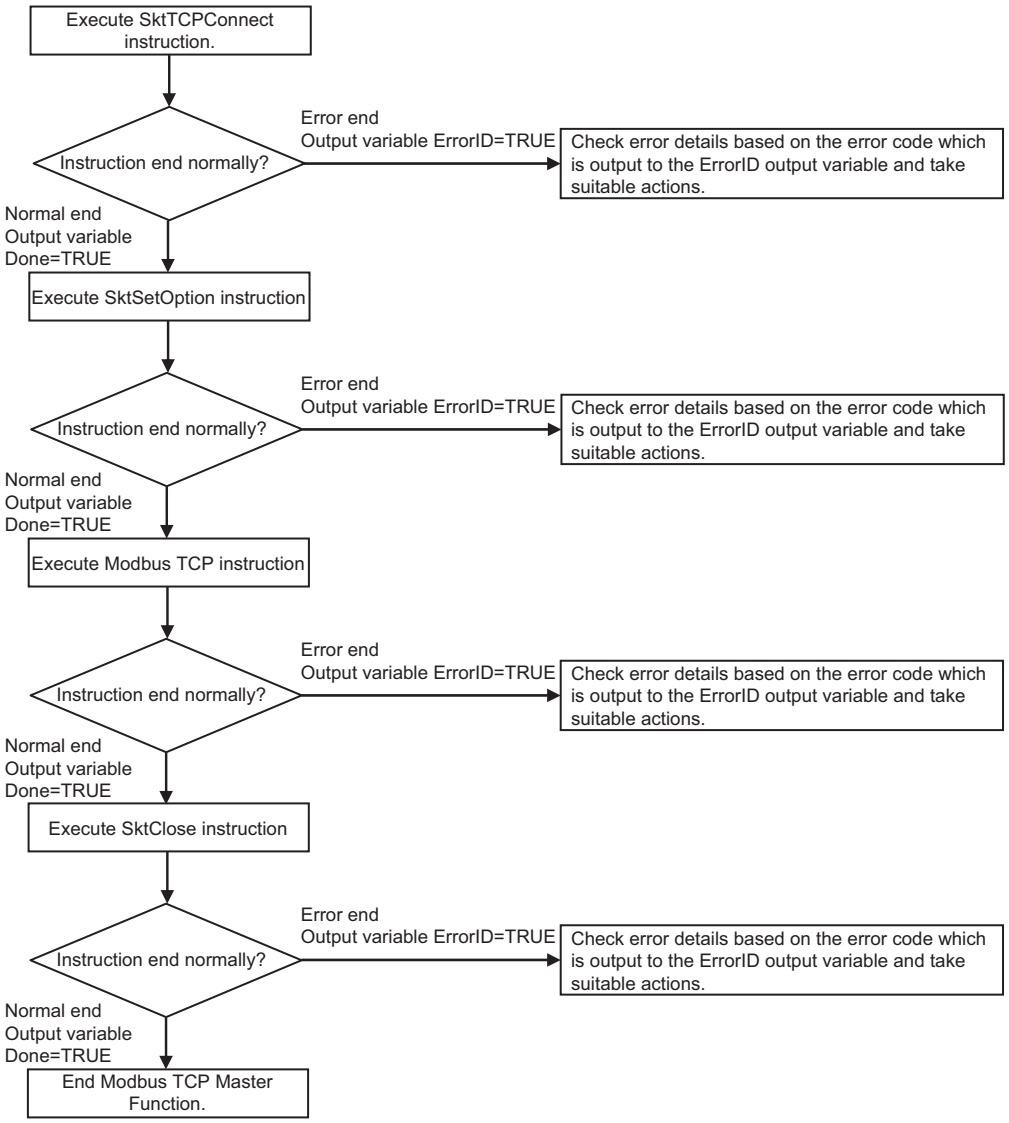
9-3 Modbus TCP Master Function Procedure

When you use the Modbus TCP Master Function, you need to also use the following instructions other than the Modbus TCP instruction.

Instruction	Description
SktTCPConnect	Establishes the TCP/IP connection with the Modbus TCP slave before the execution of the Modbus TCP instruction. The default connection port is 502.
SktClose	Disconnects the TCP/IP connection with the Modbus TCP slave after the execution of the Modbus TCP instruction.
SktSetOption	The application of the TCP-NODELAY option in the TCP/IP settings with the Modbus standard is recommended. Set it before the execution of the Modbus TCP instruction after the TCP/IP connection is established with the Modbus TCP slave.
SktClearBuf	The receive buffer is not cleared during the execution of the Modbus TCP instruction. This instruction is executed if the receive buffer needs to be cleared during use of the Modbus TCP instruction. For example, execute this instruction when the previous Modbus TCP command response may be stored in the receive buffer.

Procedure

Use the Modbus TCP Master Function as follows. Check the values of the output variables of each instruction to confirm whether the instruction ended normally.



If the response from the other equipment is slow and the Modbus TCP instruction ends before the response is returned, there may be data remaining in the receive buffer. In such cases, execute the Modbus TCP instruction after the receive buffer is cleared with the SktClearBuf instruction or SktTCPConnect instruction.

Refer to the Modbus TCP instructions in the *NJ/NX-series Instructions Reference Manual (Cat. No. W502)* for sample programming.

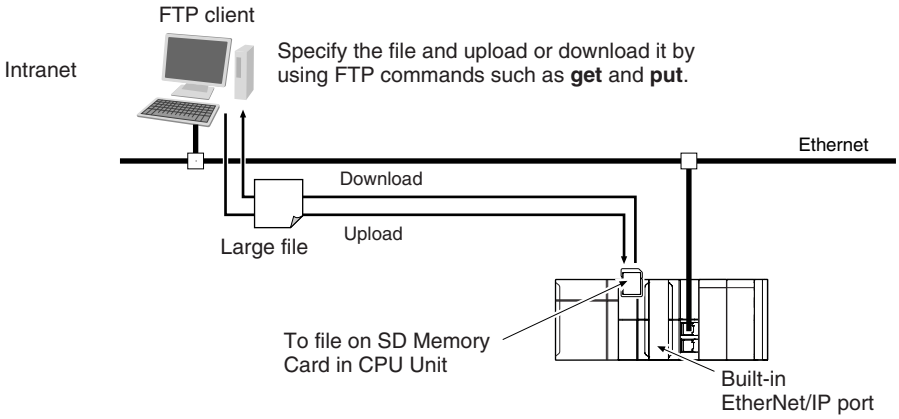
10

FTP Server

10-1 Overview and Specifications

10-1-1 Overview

The built-in EtherNet/IP port has FTP (File Transfer Protocol) server capabilities. You can therefore send FTP commands from an FTP client software application on a computer on the Ethernet network to upload and download large files from and to an SD Memory Card.



10-1-2 Specifications

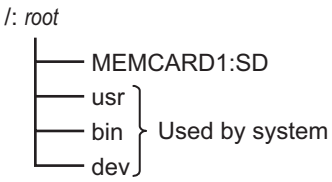
Item	Specifications	
Executable commands	open	: Connects the specified host FTP server.
	user	: Specifies a user name for the remote FTP server.
	ls	: Displays file names in the remote host.
	mls	: Displays file names in multiple remote hosts.
	dir	: Displays file names and details in the remote host.
	mdir	: Displays file names and details in multiple remote hosts.
	rename	: Changes a file name.
	mkdir	: Creates a new directory in the working directory on the remote host.
	rmdir	: Deletes a directory from the working directory on the remote host.
	cd	: Changes the work directory on the remote host to the specified directory.
	pwd	: Displays the work directory on the remote host.
	type	: Changes the file transfer type.
	get	: Transfers a specified remote file to the local host.
	mget	: Transfers specified multiple remote files to the local host.
	put	: Transfers a specified local file to the remote host.
	mput	: Transfers specified multiple local files to the remote host.
	delete	: Deletes a specified file from the remote host.
	mdelete	: Deletes specified multiple files from the remote host.
	append	: Uses the currently specified file data type to append a local file to the remote host.
	close	: Disconnects the FTP server.
	bye	: Closes the FTP client.
	quit	: Closes the FTP client.

Item	Specifications
Protection	Login name (up to 12 characters) Password consists of 8 to 32 characters.
Protocol used	FTP (Port No.: 20/TCP, 21/TCP)
Number of connections	6

10-2 FTP Server Function Details

10-2-1 Supported Files

The file system in the Controller that can be accessed by the built-in EtherNet/IP port includes files in an SD Memory Card mounted in the CPU Unit.
The directory tree is shown below.



A connection is initially made to the root directory.

Additional Information

- The date of the MEMCARD1 directory displayed for ls, dir, and mkdir commands in the root directory is the date of the file system volume label.
- The login date is displayed for MEMCARD1 if a volume label has not been created.

10-2-2 Connecting to the FTP Server

Input the FTP login name and password to login to the built-in EtherNet/IP port from an FTP client application. Use the Built-in EtherNet/IP Port Settings in the Sysmac Studio to set the FTP login name and password.

Additional Information

When a general-purpose FTP application is used, you can use a graphical user interface similar to Explorer to transfer and read files.

● Login Name and Password Setting

The FTP login name and password are not set by default.
Use the Built-in EtherNet/IP Port Settings to set any login name and password.

● Login Messages

Status	Message
Normal connection	220 xxx.xx.xx.xx FTP server ready. xxx.xx.xx.xx: CPU Unit model (example: NJ501-1300)
Connected to maximum number of connections (6)	530 FTP server busy, Goodbye.

● Restrictions on Login Name and Password Setting

The following restrictions apply to login names and passwords.

- Only single-byte alphanumeric characters can be used for login names and passwords. The login name and password are case sensitive.
- A login name consists of up to 12 characters.

- A password consists of 8 to 32 characters.
- Always set a password when you set a new login name. The login name will not be valid unless a password is set for it.
- The login name is invalid if the login name is not set or characters other than single-byte alphanumeric characters are used.

● FTP File Transfer Mode

FTP has two file transfer modes: ASCII mode and binary mode. Before you start to transfer files, use the type command (specifies the data type of transferred files) to select the required mode.

- To transfer a file in binary format: Select binary mode.
- To transfer a file in ASCII format: Select ASCII mode.

● Multiple Accesses to the Same File

Files accessed with the FTP server may be simultaneously accessed by multiple sources with communications commands from other FTP servers or programming instructions.
Exclusive control is required to prevent multiple accesses.
This is to prevent reading and writing the same file at the same time.
The CPU Unit automatically performs exclusive control as shown below only when the following combinations of instructions are used.
In other cases, use file operation instructions (Change File Name, Copy File, etc.) or communications commands and perform exclusive control.

- Exclusive Control When Accessing the Same File on the SD Memory Card

<table>
<tr><th colspan="3" rowspan="3"></th><th colspan="6">First access</th></tr>
<tr><th colspan="2" rowspan="2">Instructions*1</th><th colspan="2" rowspan="2">File operations from the Sysmac Studio</th><th colspan="2">FTP server</th></tr>
<tr><th colspan="2">Reading</th><th colspan="2">Writing</th></tr>
<tr><th colspan="3"></th><th>Reading</th><th>Writing</th><th>Reading</th><th>Writing</th><th>Reading</th><th>Writing</th></tr>
<tr><td rowspan="6">Late access</td><td rowspan="2">Instructions</td><td>Reading</td><td colspan="2" rowspan="2">Exclusive control is performed automatically, and an error occurs for the instruction that is executed later.</td><td>Exclusive control is not required.</td><td rowspan="2">Perform exclusive control.</td><td>Exclusive control is not required.</td><td rowspan="5">Perform exclusive control.</td></tr>
<tr><td>Writing</td><td>Perform exclusive control.</td><td>Perform exclusive control.</td></tr>
<tr><td rowspan="2">File operations from the Sysmac Studio</td><td>Reading</td><td>Exclusive control is not required.</td><td>Perform exclusive control.</td><td>Exclusive control is not required.</td><td>Perform exclusive control.</td><td>Exclusive control is not required.</td></tr>
<tr><td>Writing</td><td colspan="2">Perform exclusive control.</td><td colspan="2">Perform exclusive control.</td><td>Perform exclusive control.</td></tr>
<tr><td rowspan="2">FTP server</td><td>Reading</td><td>Exclusive control is not required.</td><td>Perform exclusive control.</td><td>Exclusive control is not required.</td><td>Perform exclusive control.</td><td>Exclusive control is not required.</td><td>Perform exclusive control.</td></tr>
<tr><td>Writing</td><td colspan="4">Perform exclusive control.</td><td colspan="2">Perform exclusive control.</td></tr>
</table>

*1. The instructions include the SD Memory Card operation instructions and the FTP client communications instructions.

Restrictions on Connection to FTP Server

If you repeat connection to and disconnection from the FTP server frequently in a short period of time, access to the server may be restricted temporarily for system protection. If you cannot connect to the FTP server, wait for 10 minutes and try again.

10-3 Using the FTP Server Function

10-3-1 Procedure

1. Make the basic settings.
   Refer to *1-5 EtherNet/IP Communications Procedures* on page 1-29 for the basic operation flow.
2. Set up the FTP server on the Sysmac Studio. (Refer to *4-3 **FTP Settings** Display* on page 4-12.)
3. Select **Controller Setup** - **Built-in EtherNet/IP Port Settings** on the Sysmac Studio. Make the following settings on the **FTP Settings** Display.
   - FTP server
   - Port number
   - Login name
   - Password
4. Connect the CPU Unit online and transfer the settings to the Controller.
5. Insert the SD Memory Card into the CPU Unit.
6. Connect to the built-in EtherNet/IP port from an FTP client.
7. Input the FTP login name and password that you set in the Built-in EtherNet/IP Port Settings to log in to the built-in EtherNet/IP port.
8. After you are logged in, you can use ftp commands, such as cd (Change Directory) and get (Obtain File) for the MEMCARD1 directory in the SD Memory Card in the Controller.
9. Close the connection.

10-3-2 List of Settings Required for the FTP Server Function

Make the following settings for the unit setup when the FTP server function is used.

Built-in EtherNet/IP Port Settings Tab Page on Sysmac Studio	Setting	Setting conditions	Reference
FTP	FTP server	Required	page 7-48
	Port No.	Any number*1 Required when changing the default value of 21.	
	Login name	Required*1	
	Password	Required*1	

*1. If the **Do not use** Option is selected for the **FTP server**, these settings are not required.

Precautions for Correct Use

Allow packets from the FTP client if the **Use** Option is selected for Packet Filter of the built-in EtherNet/IP port. If they are not permitted, communication with the FTP client is not possible. For the details on the settings, refer to *7-5 CIP Object Services* on page 7-48.

Additional Information

Make settings in the **FTP Settings** Display if the FTP server is used. Refer to *7-5 CIP Object Services* on page 7-48 for information on the **FTP Settings** Display.

10-4 FTP Server Application Example

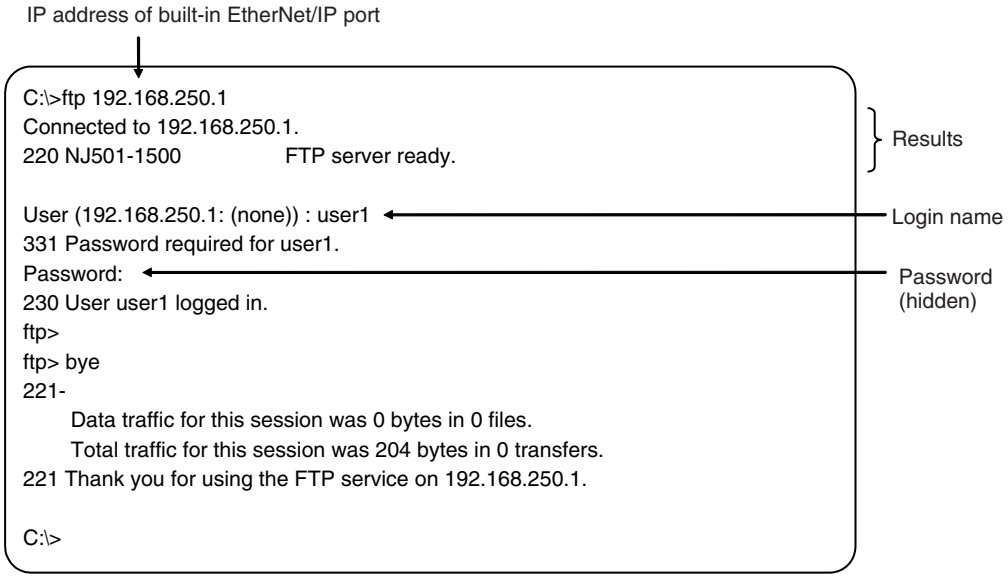
An example of using the FTP server with the login name "user1" and the password "password" is shown below.

Additional Information

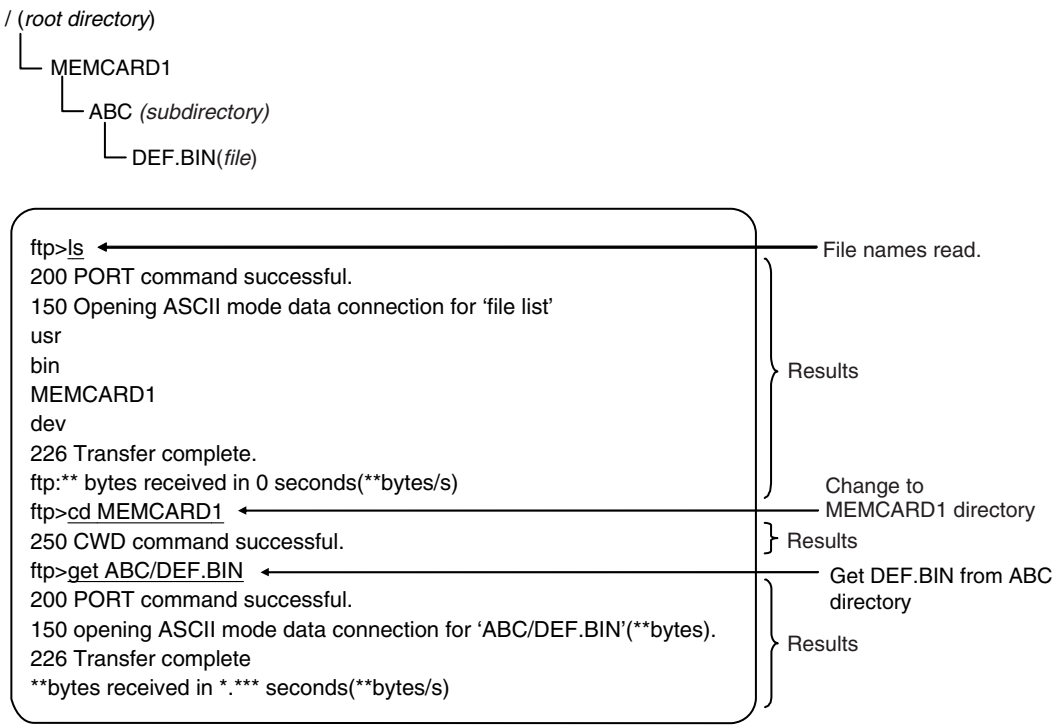
When a general-purpose FTP application is used, you can use a graphical user interface similar to Explorer to transfer and read files.

Step

1. Make sure that an SD Memory Card is inserted and turn ON the power supply to the Controller.
2. Connect to the FTP server from a computer on the Ethernet by entering the text that is underlined in the following diagram.



3. Enter FTP commands (underlined in the following diagram) to read and write files. The following directory tree is used in this example.


/ (root directory)
MEMCARD1
ABC (subdirectory)
DEF.BIN(file)
ftp>ls
File names read.
200 PORT command successful.
150 Opening ASCII mode data connection for 'file list'
usr
bin
MEMCARD1
dev
226 Transfer complete.
ftp:** bytes received in 0 seconds(**bytes/s)
Results
ftp>cd MEMCARD1
Change to MEMCARD1 directory
250 CWD command successful.
Results
ftp>get ABC/DEF.BIN
Get DEF.BIN from ABC directory
200 PORT command successful.
150 opening ASCII mode data connection for 'ABC/DEF.BIN'(**bytes).
226 Transfer complete
bytes received in *.* seconds(**bytes/s)
Results

10-5 Using FTP Commands

This section describes the FTP commands which the host computer (FTP client) can send to the FTP server of the built-in EtherNet/IP port.
There may be slight differences in the descriptions depending on the model of your workstation. Refer to your workstation's operation manuals for details.

10-5-1 Table of Commands

The FTP commands which can be sent to the built-in EtherNet/IP port are listed in the following table.

Command	Description
open	Connects the specified host FTP server.
user	Specifies a user name for the remote FTP server.
ls	Displays file names in the remote host.
mls	Displays file names in multiple remote hosts.
dir	Displays file names and details in the remote host.
mdir	Displays file names and details in multiple remote hosts.
rename	Rename a file
mkdir	Creates a new directory in the working directory on the remote host.
rmdir	Deletes a directory from the working directory on the remote host.
cd	Changes the work directory on the remote host to the specified directory.
pwd	Displays the work directory on the remote host.
type	Changes the file transfer type.
get	Transfers a specified remote file to the local host.
mget	Transfers specified multiple remote files to the local host.
put	Transfers a specified local file to the remote host.
mput	Transfers specified multiple local files to the remote host.
delete	Deletes a specified file from the remote host.
mdelete	Deletes specified multiple files from the remote host.
append	Uses the file data type that is specified by the type command to append a local file to the remote host.
close	Disconnects the FTP server.
bye	Closes the FTP client.
quit	Closes the FTP client.

Note 1. "Remote host" refers to the built-in EtherNet/IP port.
Note 2. "Remote file" refers to a file on the SD Memory Card in the CPU Unit.
Note 3. "Local host" refers to the host computer (FTP client).
Note 4. "Local file" refers to a file on the host computer (FTP client).

10-5-2 Using the Commands

open

- **Format**

  open [IP_address or host_name_of_FTP_server]

● Function

- Connects the FTP server. Normally, the FTP server IP address is specified to execute this command automatically when the FTP client is booted.

user

● Format

user [user_name]

● Function

- Specifies the user name. Specify the FTP login name set in the built-in EtherNet/IP port system setup.
- The user name is automatically requested immediately after connection to the FTP server is opened.

ls

● Format

ls [-l] [remote_file_name [local_file_name]]

● Function

- Displays the names of files on the remote host (on the SD Memory Card).
- Set the switch [-l] to display not only the file names but the creation dates and sizes as well. If the switch is not set, only the file names are displayed.
- Specify a file on the SD Memory Card for the remote_file_name.
- If the local_file_name is specified, the file information is stored in the specified file.

mls

● Format

mls remote_file_name local_file_name

● Function

- Displays a list of the names of files on multiple remote hosts (on the SD Memory Card).
- For the remote_file_name, specify a directory on the SD Memory Card in which you wish to list files contained, or a file name. Input an asterisk (*) to display a list of the current working directory.
- If the local_file_name is specified, the file information is stored in the specified file. Input a hyphen (-) to display a list of the remote hosts but not store the list of file names.

dir

Format

dir [remote_file_name [local_file_name]]

Function

- Displays the names, creation dates, and sizes of files on the remote host (on the SD Memory Card).
- It displays the same information as command [ls -l].
- Specify a file on the SD Memory Card for the remote_file_name.
- If the_local_file name is specified, the file information is stored in the specified file.

mdir

Format

mdir remote_file_name local_file_name

Function

- Displays the names of files, subdirectories, creation dates, and sizes on multiple remote hosts (on the SD Memory Card).
- For the remote_file_name, specify the directory or file name on the SD Memory Card you wish to list. Input a hyphen (-) to display a list of the current working directory.
- If the_local_file_name is specified, the file information is stored in the specified file. Input a hyphen (-) to display a list of the remote hosts and not store the file information.

rename

Format

rename current_file_name new_file_name

Function

- Changes the specified current file name to the specified new file name.
- If the new file name is already used by an existing file on the remote host (on the SD Memory Card), the existing file is overwritten by the file whose name was changed.
- rename can just change the file name. It cannot be used to move the file to a different directory.

mkdir

Format

mkdir directory_name

Function

- Creates a directory of the specified name on the remote host (on the SD Memory Card).

- An error will occur if a file or directory of the same name already exists in the working directory.

rmdir

Format

rmdir directory_name

Function

- Deletes the directory with the specified name from the remote host (from the SD Memory Card).
- The directory must be empty to be deleted.
- An error will occur if the specified directory does not exist or is not empty.

pwd

Format

pwd

Function

- Displays the work directory on the remote host.

append

Format

append local_file_name [remote_file_name]

Function

- Uses the file data type that is specified by the type command to append the local file to the remote host (on the SD Memory Card).

cd

Format

cd [directory_name]

Function

- Changes the remote host work directory to the specified remote directory.
- Files on the SD Memory Card are stored in the MEMCARD1 directory under the root directory (/).
- The root directory (/) is the directory that is used when you log onto the built-in EtherNet/IP port. The MEMCARD1 directory does not exist if an SD Memory Card is not inserted in the CPU Unit or if the SD Memory Card power indicator on the CPU Unit is not lit.

type

Format

type data_type

Function

- Specifies the file data type.
- The following data types are supported:
  ascii: Files are transferred as ASCII data.
  binary (image): Files are transferred as binary data.
  The CPU Unit handles binary files. Use the type command to specify binary transfers before you upload or download files.
- The default file type is ASCII.

get

Format

get file_name [receive_file_name]

Function

- Transfers the specified remote file from the SD Memory Card to the local host.
- You can specify the name of the file to be received on the local host by setting receive file name.

mget

Format

mget file_name

Function

- With wildcards (*) included in the file_name, transfers multiple remote files from the SD Memory Card to the local host.

put

Format

put file_name [destination_file_name]

Function

- Transfers the specified local file to the remote host (to the SD Memory Card).
- You can save the transfered file with the name you specify for the destination_file_name.
- Any existing file with the same name in the remote host (on the SD Memory Card) is overwritten by the contents of the transferred file.

mput

Format

mput file_name

Function

- With wildcards (*) included in the file_name, transfers multiple local files to the remote host (to the SD Memory Card).
- Any existing file with the same name in the remote host (on the SD Memory Card) is overwritten by the contents of the transferred file.

delete

Format

delete file_name

Function

- Deletes the specified remote file (on the SD Memory Card).

mdelete

Format

mdelete file_name

Function

- With wildcards (*) included in the file_name, deletes multiple remote files from the SD Memory Card.

close

Format

close

Function

- Disconnects the FTP server of the built-in EtherNet/IP port.

bye

Format

bye

● Function

- Ends the FTP session.

quit

● Format

quit

● Function

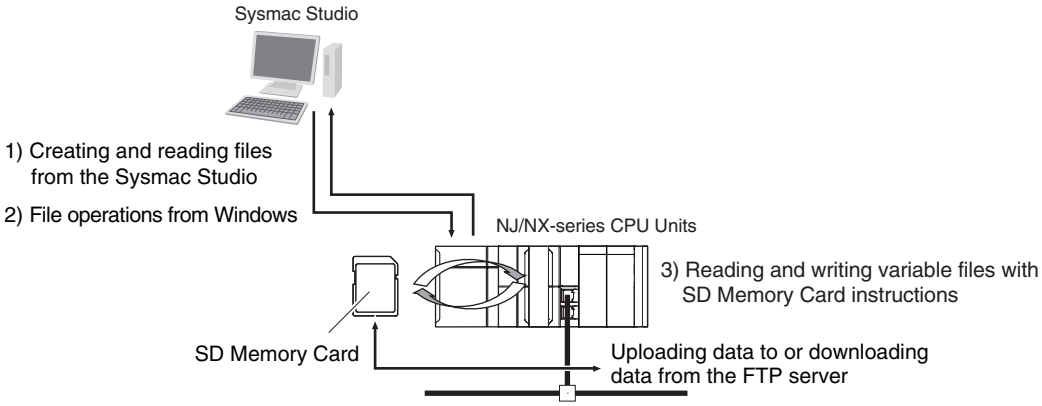
- Ends the FTP session.

10-6 Using SD Memory Card Operations

The built-in EtherNet/IP port can be used to upload and download the following data between the SD Memory Card and the FTP server.

- Variables files (binary format)

The following three methods are available when a CPU Unit saves data to and reads data from the SD Memory Card.



10-6-1 SD Memory Card Types

Refer to *Specifications of Supported SD Memory Cards, Folders, and Files* in the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for details.

10-6-2 File Types

File Names

File names and extensions are assigned to identify files.

The following characters can be used in file names and extensions. File names are not case sensitive.

A to Z, a to z, and 0 to 9

the following symbols: $ % ' - _ @ ! ‘ ( ) ~ = # & + ^ [ ] { } , . ;

The following characters cannot be used in files names.

Blanks, multi-byte characters, and the following symbols: / \ ? * ” : < >

The maximum length of a file name with the extension is 65 characters.

The first period (.) in a file name is taken as the delimiter between the file name and extension.

Extensions are determined by the file type.

Directory

You can create up to five levels of directories to store files on the SD Memory Card (count the root directory as one level).

A maximum of 65 characters can be used in a directory name.

File Names Handled by CPU Unit

The files described in the following table can be read or written by the CPU Unit.

File type	File name	Extension	Contents	Description
Variables file (binary format)	Refer to *10-6-2 File Types* on page 10-18.	.bin	Specified variables	The variables file contains the values of specified variables (which include arrays and structures) in binary format (.bin).

Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for details.

10-6-3 Initializing SD Memory Cards

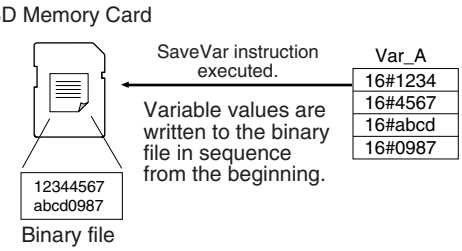
1 Insert the SD Memory Card into the CPU Unit.

2 Use the Sysmac Studio to initialize the SD Memory Card.

10-6-4 Format of Variable Data

Binary Format

This is a data format used for binary data specified by the ladder instructions, FileReadVar (Read Variables File) and FileWriteVar (Save Variables File), in the CPU Unit.
You can also read and save arrays and structures.
Data is created as shown below when the data of variable Var_A is placed in an attached file in binary format.



Additional Information

- When you handle a binary file on the NJ/NX-series CPU Unit, always specify the binary data type with the type command before you read or write the file via FTP. (Refer to *10-5-2 Using the Commands* on page 10-11.)
- For details on how to use ladder diagram instructions to process files, refer to the *NJ/NX-series Instructions Reference Manual (Cat. No. W502)*.

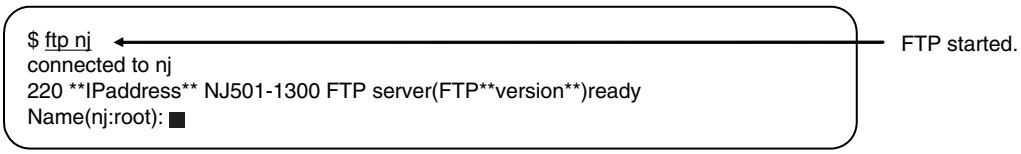
10-7 Application Example from a Host Computer

The following procedure provides an example of FTP operations from a host computer. In this example, the following assumptions are made.

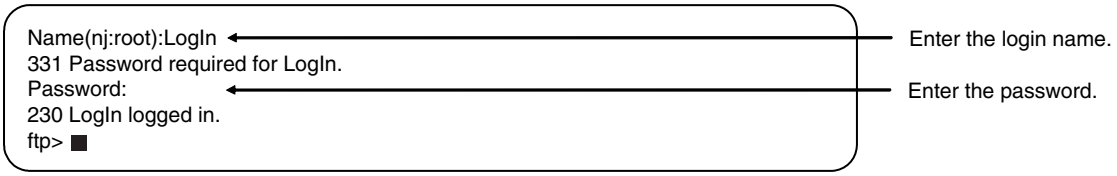
- The IP address of the built-in EtherNet/IP port is registered in the hosts as host name [nj].
- The FTP login name is "LogIn".
- Manufacturing results are stored in a file named RESULT.BIN. in the SD Memory Card in the CPU Unit.
- A manufacturing instructions data file called PLAN.BIN already exists on the workstation.

In the following procedure, the manufacturing results file (RESULT.BIN) in the SD Memory Card in the CPU Unit is transferred to a workstation, and then a manufacturing instructions file (PLAN.BIN) on the workstation is transferred to the SD Memory Card in the CPU Unit.
Underlined text is keyed in from the FTP client. The workstation prompt is indicated as $, and the cursor is indicated as ■.

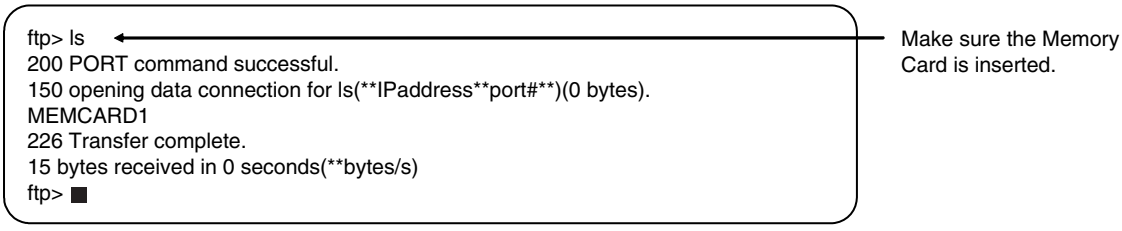
1. Start the FTP application and connect to the built-in EtherNet/IP port.



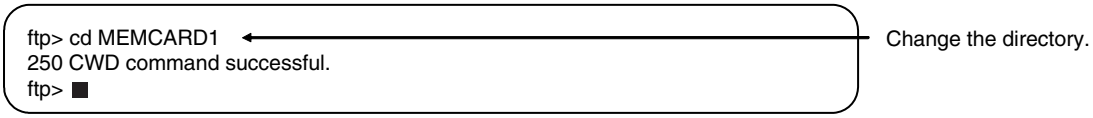
2. Enter the login name.



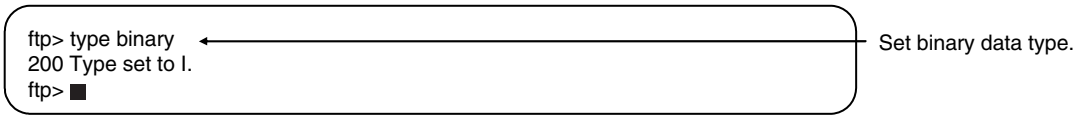
3. Make sure the Memory Card is correctly inserted. The MEMCARD1 directory is displayed if there is an SD Memory Card in the CPU Unit.



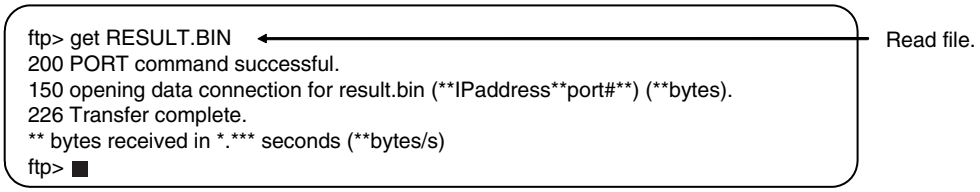
4. Change to the MEMCARD1 directory.



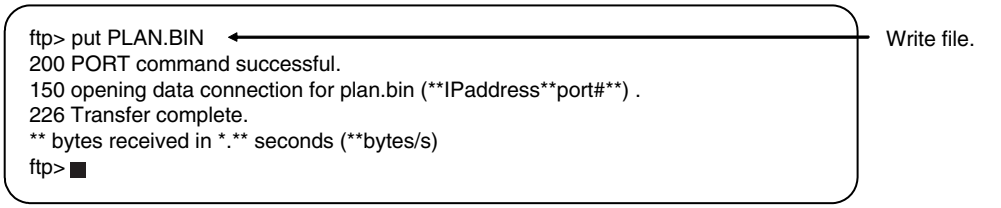
5. Change data type to binary.



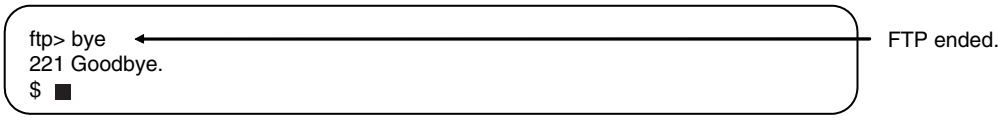
6. Read the file RESULT.BIN and transfer it to the workstation.



7. Write the file PLAN.BIN to the Memory Card.



8. End the FTP session.



11

FTP Client

11-1 Using the FTP Client to Transfer Files

You can use the FTP client to transfer files between the FTP client and an FTP server. You can transfer files in either direction: download data from the FTP server to the FTP client or upload data from the FTP client to the FTP server.

Version Information

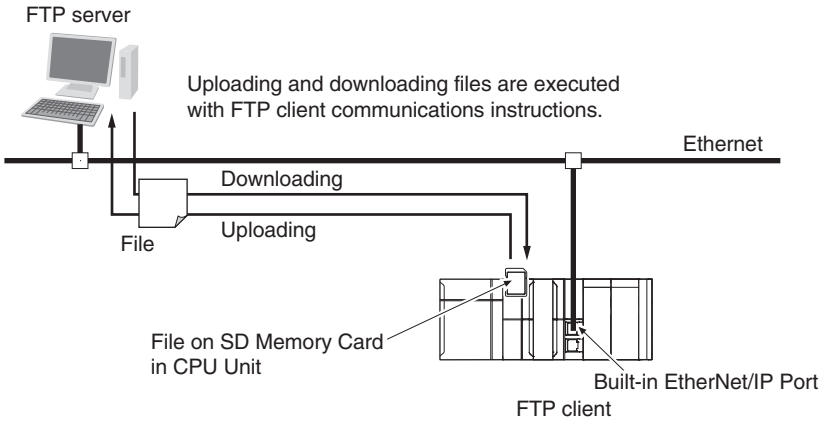
A CPU Unit with unit version 1.08 or later is required to use the FTP client.

11-1-1 Transferring Files

All file transfers that use the FTP client are executed with FTP client communications instructions in the user program. The file transfer settings are all made with the parameters of the FTP client communications instructions. No settings are required from the Sysmac Studio.
The FTP client communications instructions and their functions are given in the following table. You can execute up to three FTP client communications instructions at the same time.

Instruction	Function
FTPGetFileList	Gets a file list from the FTP server.
FTPGetFile	Downloads one or more files from the FTP server.
FTPPutFile	Uploads one or more files to the FTP server.
FTPRemoveFile	Deletes one or more files from the FTP server.
FTPRemoveDir	Deletes a directory from the FTP server.

Downloaded files are stored on the SD Memory Card. When uploading files, files that are stored on the SD Memory Card are uploaded to the FTP server. Therefore, when you upload or download files, an SD Memory Card must be inserted in the NJ-/ NX series CPU Unit.



11-1-2 Connectable FTP Servers

An NJ/NX-series CPU Unit can connect to the following FTP servers. Refer to the relative manuals for information on setting and using the FTP servers.

- Built-in EtherNet/IP port on NJ/NX-series CPU Unit
- CJ-series EtherNet/IP Unit with unit version 2.0 or later

- CJ-series CJ2 CPU Unit with Built-in EtherNet/IP
- CJ-series CJ1M CPU Unit with Ethernet Functions
- CJ-series Ethernet Unit
- Windows7: Windows Server 2008 R2 (Internet Information Services (IIS) 7.5)
- Windows8: Windows Server 2012 (IIS8.0)
- Windows10: Windows Server2016 (IIS10.0)
- Linux

11-1-3 File Transfer Options

You can use the following options for file transfers. All the options are specified in the parameters of the FTP client communications instructions.

- File transfer mode
- Open mode for data connection
- Deleting files after transfer
- Overwriting

The following sections describe each of these options.

● File Transfer Mode

There are two file transfer modes, ASCII Mode and Binary Mode, that differ in how line feeds in text data are handled. The following table describes the differences.

Transfer mode	Handling of line feeds in text data
ASCII Mode	Line feeds are converted to the line feed code of the destination system, e.g., Unix or Windows.
Binary Mode	Line feeds are transferred without conversion.

● Open Mode for Data Connection

In order to transfer files, a TCP connection between the FTP server and FTP client should be opened. TCP connections include control connections to control communications and data connections to transfer data. When a data connection is opened, the connection is assigned with either Active Mode or Passive Mode, depending on whether the connection request is issued by the FTP server or FTP client. The following table describes the differences.

Open mode	Request to establish a connection
Active Mode	The FTP server makes the connection request.
Passive Mode	The FTP client makes the connection request.

For example, if the FTP server is not on the Internet and you use Active Mode to open a data connection, a connection request from the FTP server may not be permitted due to security policies. In this case, you must set Passive Mode for the data connection and sends a connection request from the FTP client.

● File Deletion after Transfer

You can specify whether to delete the source files after the file transfer. If the file transfer fails for any reason, the source files are not deleted even if deletion is specified.

● Overwriting

You can specify whether to overwrite a file of the same name as the transferred file at the file transfer destination. If you specify not overwriting files and a file of the same name exists at the transfer destination, the source file will not be transferred.

11-1-4 Other Functions

You can also use the following two functions for file transfers.

- Retrying connection processing with the FTP server
- Using wildcards to specify the files to transfer

These functions are described in the following sections.

● Retrying Connection Processing with the FTP Server

If connection processing fails to connect with the FTP server, the connection is automatically retired up to three times. You can set the timeout time that is used to determine connection failure, the number of retries, and the retry interval.

● Using Wildcards to Specify the Files to Transfer

You can use wildcards to specify the names of files to transfer. This allows you to transfer more than one file at one time.

11-2 FTP Client Communications Instructions

FTP client communications instructions are always used to transfer files with the FTP client. The FTP client communications instructions and their functions are given in the following table.

Instruction	Function
FTPGetFileList	Gets a file list from the FTP server.
FTPGetFile	Downloads one or more files from the FTP server.
FTPPutFile	Uploads one or more files to the FTP server.
FTPRemoveFile	Deletes one or more files from the FTP server.
FTPRemoveDir	Deletes a directory from the FTP server.

For details on the FTP client communications instructions, refer to the *NJ/NX-series Instructions Reference Manual (Cat. No. W502)*.

11-2-1 Functions of the FTP Client Communications Instructions

This section describes the functions of the FTP client communications instructions.

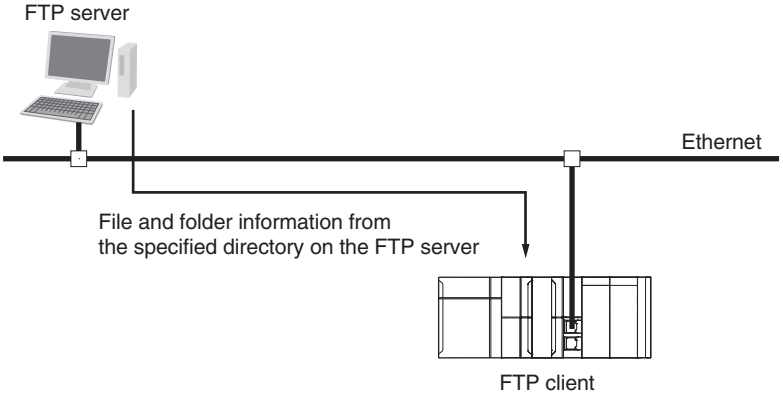
FTPGetFileList Instruction

The FTPGetFileList instruction gets a list of files and folders in a specified directory on the FTP server. The following information is obtained.

- The number of files and folders in the specified directory
- The names of the files and folders
- The last updated date and time of each file and folder
- The file sizes
- The read-only attributes of the files and folders

You can specify the following option.

- Open Mode for data connection



Additional Information

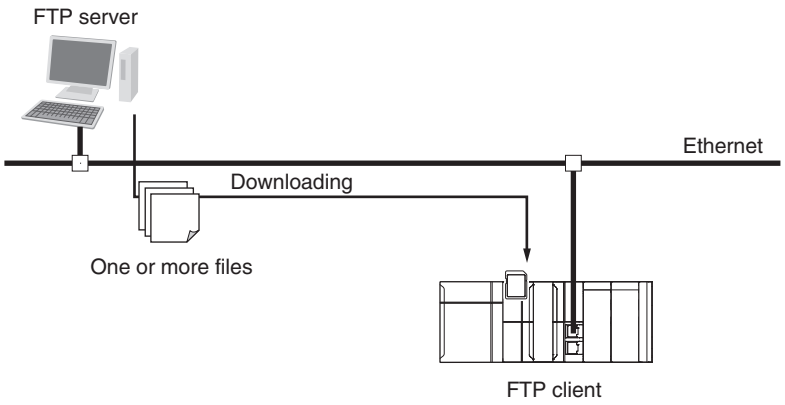
The updated dates of files at 12 am and 12 pm are improved in the CPU Unit with unit version 1.14 or later.

FTPGetFile Instruction

The FTPGetFile instruction downloads the specified file from the specified directory on the FTP server to the specified directory in the SD Memory Card.
You can use wildcards to specify the file name to allow you to download more than one file at the same time.
If the directory specified for the download does not exist in the SD Memory Card, the directory is created and the data is downloaded to it.

You can specify the following options.

- Transfer mode
- Open Mode for data connection
- Deleting files after transfer
- Overwriting

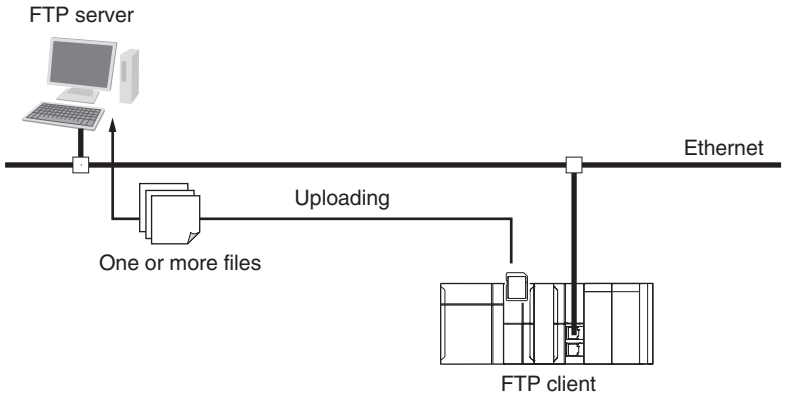


FTPPutFile Instruction

The FTPPutFile instruction uploads the specified file from the specified directory in the SD Memory Card to the specified directory on the FTP server.
You can use wildcards to specify the file name to allow you to upload more than one file at the same time.
If the directory specified for the upload does not exist on the FTP server, the directory is created and the data is uploaded to it.

You can specify the following options.

- Transfer mode
- Open Mode for data connection
- Deleting files after transfer
- Overwriting

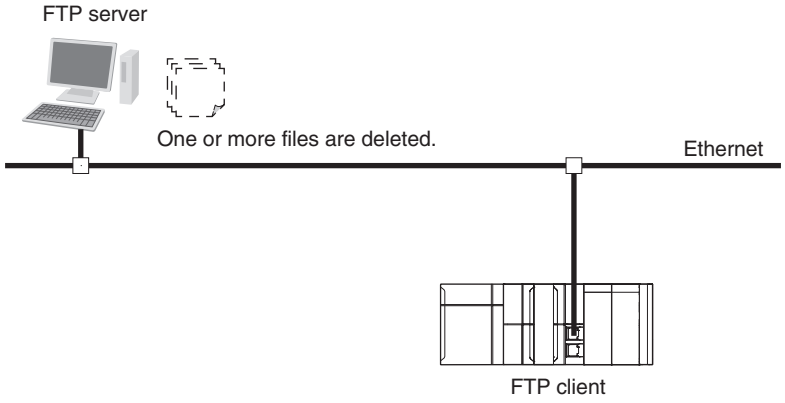


FTPRemoveFile Instruction

The FTPRemoveFile instruction deletes the specified file in the specified directory on the FTP server. You can use wildcards to specify the file name to allow you to delete more than one file at the same time.

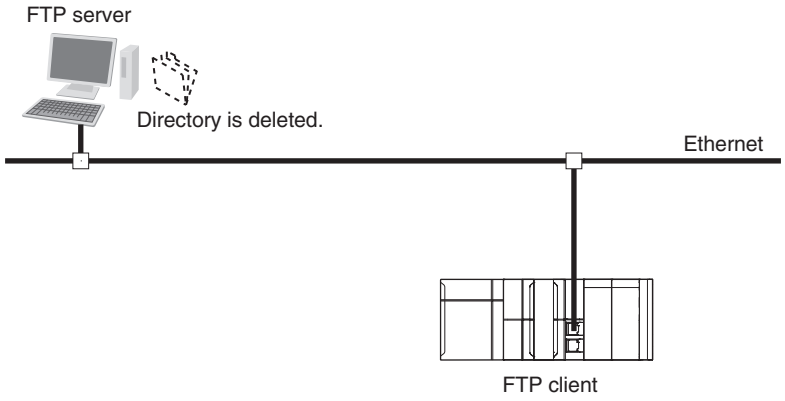
You can specify the following option.

- Open Mode for data connection



FTPRemoveDir Instruction

The FTPRemoveDir instruction deletes the specified directory from the FTP server.



11-2-2 Restrictions on the FTP Client Communications Instructions

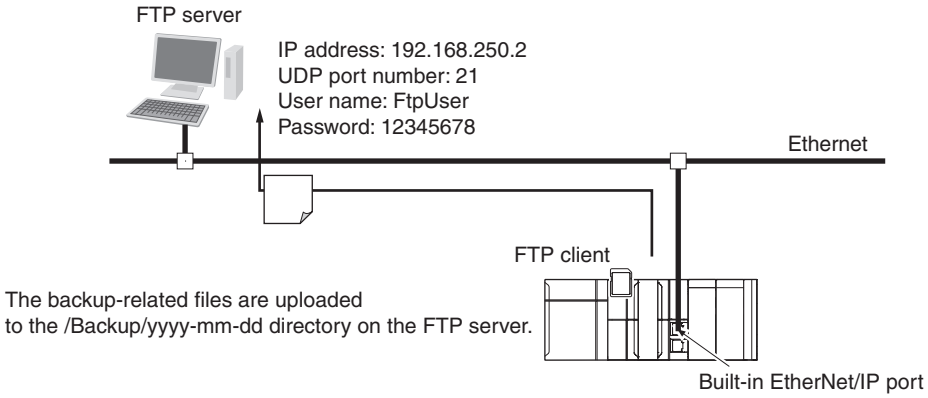
The following restrictions apply to the FTP client communications instructions. Keep in mind these restrictions when you create the user program.

- If you execute more than one FTP client communications instruction to read and write data in the SD Memory Card at the same time, unexpected operation may result, such as reading data from a file to which data is being written. Perform exclusive control of the instructions in the user program.
- If you execute an FTP client communications instruction to read or write data in the SD Memory Card at the same time as another operation to read or write data in the SD Memory Card, unexpected operation may result, such as reading data from a file to which data is being written. Perform exclusive control of the instructions in the user program. Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for information on exclusive control of access to files in the SD Memory Card.

11-3 FTP Client Application Example

FTP client functionality is executed with FTP client communications instructions. This section provides sample programming that uses the FTP client communications instructions.

This program executes an SD Memory Card backup and then uploads all of the backup-related files to the /Backup/yyyy-mm-dd directory on the FTP server.



The Controller is connected to the FTP server through an EtherNet/IP network. The settings of the parameters to connect to the FTP server are given in the following table.

Parameter	Value
IP address	192.168.250.2
UDP port number	21
User name	FtpUser
Password	12345678

The following procedure is used.

1 The BackupToMemoryCard instruction is used to save backup-related files of a NJ/NX-series Controller to the root directory on the SD Memory Card.

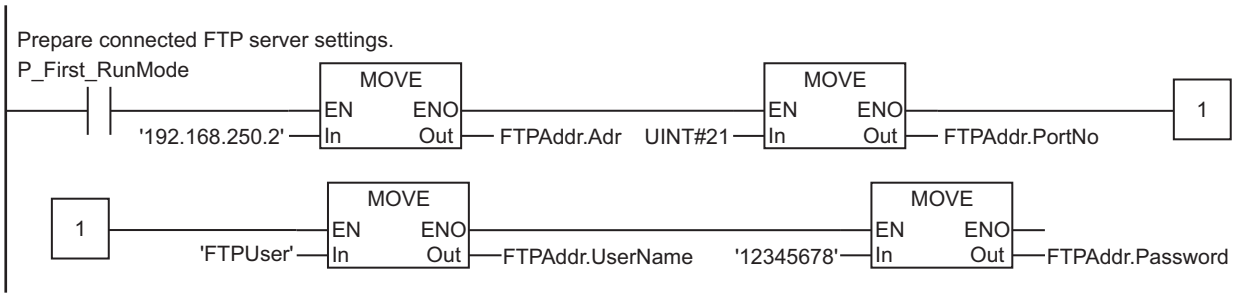
2 The FTPPutFile instruction is used to upload the backup-related files to the /Backup/yyyy-mm-dd directory on the FTP server. The wildcard specification *.* is used to specify the names of the files to transfer.

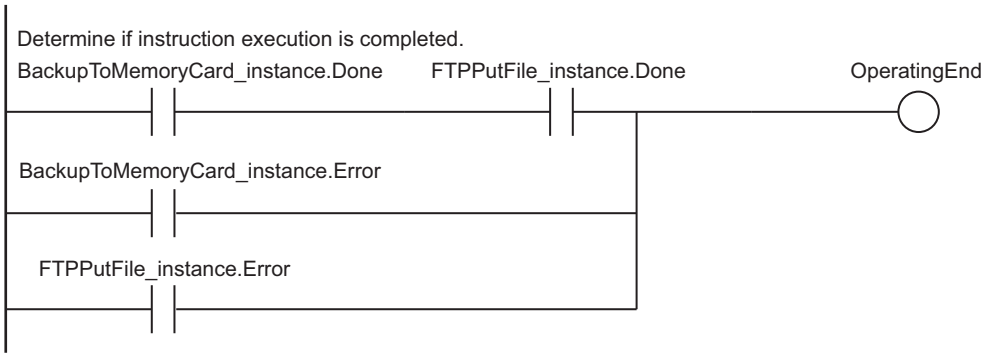
3 Normal end processing is performed if the operation is normally completed. Error end processing is performed if an error occurs.

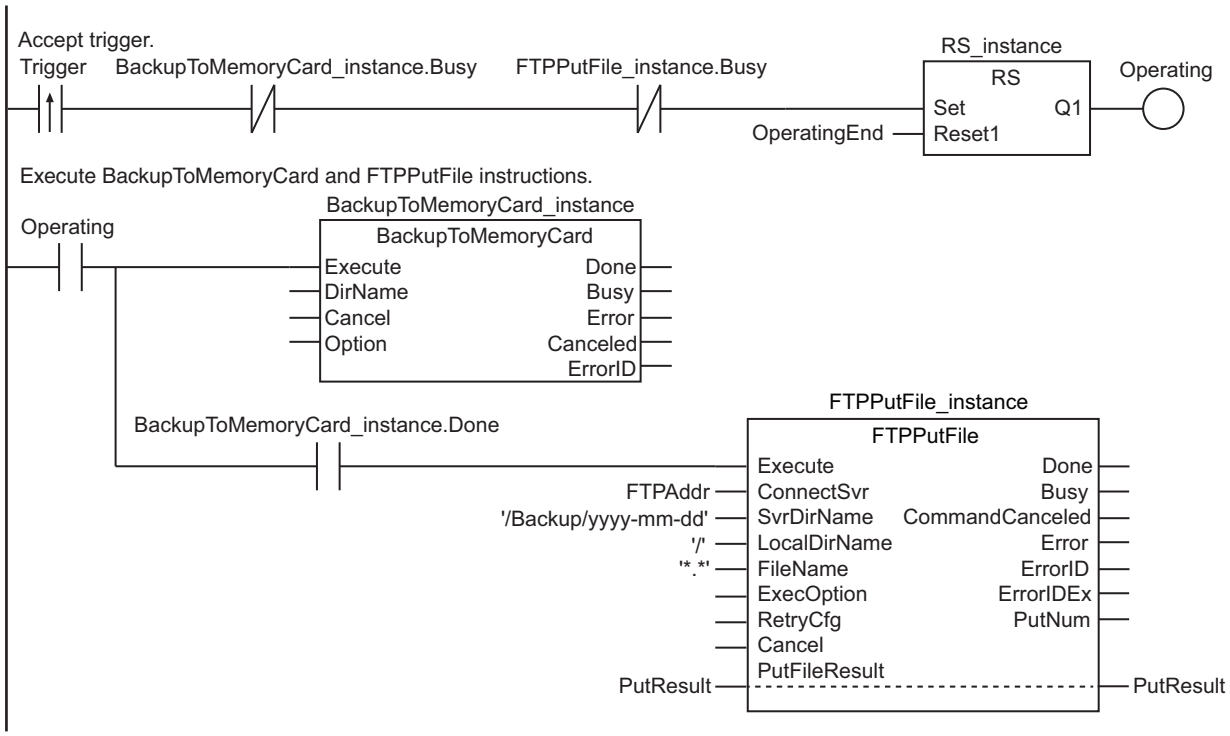
LD

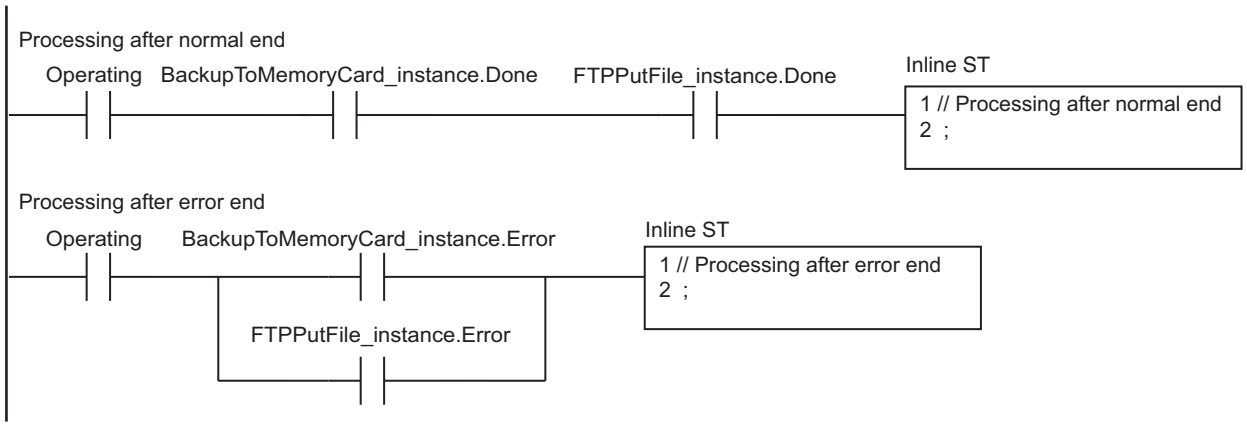
Internal variables	Variable	Data type	Initial value	Comment
	FTPPutFile_instance	FTPPutFile		Instance of FTPPutFile instruction
	FTPAddr	_sFTP_CONNECT_SVR	(Adr := '', PortNo := 0, UserName := '', Password := '')	Connected FTP server settings

Internal variables	Variable	Data type	Initial value	Comment
	PutResult	ARRAY[0..0] OF _sFTP_FILE_RESULT	[(Name := '', TxError := False, RemoveError := False, Reserved := [4(16#0)])]	Uploaded file results
	RS_instance	RS		Instance of RS instruction
	OperatingEnd	BOOL	FALSE	Processing completed
	Trigger	BOOL	FALSE	Execution condition
	Operating	BOOL	FALSE	Processing
	BackupToMemoryCard_instance	BackupToMemoryCard		Instance of BackupToMemoryCard instruction









ST

Internal variables	Variable	Data type	Initial value	Comment
	R_TRIG_instance	R_TRIG		Instance of R_TRIG instruction
	UP_Q	BOOL	FALSE	Trigger output
	FTPPutFile_instance	FTPPutFile		Instance of FTPPutFile instruction
	DoFTPTrigger	BOOL	FALSE	Execution condition for BackupToMemoryCard and FTPPutFile
	FTPAddr	_sFTP_CONNECT_SVR	(Adr := '', PortNo := 0, UserName := '', Password := '')	Connected FTP server settings

Internal variables	Variable	Data type	Initial value	Comment
	PutResult	ARRAY[0..0] OF _sFTP_FILE_RESULT	[(Name := '', TxError := False, RemoveError := False, Reserved := [4(16#0)])]	Uploaded file results
	Stage	UINT	0	Instruction execution stage
	Trigger	BOOL	FALSE	Execution condition
	BackupToMemoryCard_instance	BackupToMemoryCard		Instance of BackupToMemoryCard instruction

```
// Prepare connected FTP server settings.
IF P_First_RunMode THEN
    FTPAddr.Adr             := '192.168.250.2';             // Address
    FTPAddr.PortNo          := UINT#21;                     // Port number
    FTPAddr.UserName        := 'FtpUser';                   // User name
    FTPAddr.Password        := '12345678';                  // Password
END_IF;

// Accept trigger.
R_TRIG_instance(Trigger, UP_Q);
IF ( (UP_Q = TRUE) AND (BackupToMemoryCard_instance.Busy = FALSE) AND
    (FTPPutFile_instance.Busy = FALSE) ) THEN
    DoFTPTrigger            := TRUE;
    Stage := INT#1;
    BackupToMemoryCard_instance(                            // Initialize instan
ce.
        Execute             := FALSE) ;
    FTPPutFile_instance(                                    // Initialize instan
ce.
        Execute             := FALSE,
        ConnectSvr          := FTPAddr,
        SvrDirName          := '/Backup/yyyy-mm-dd',
        LocalDirName        := '/',
        FileName            := '*.*',
        PutFileResult       := PutResult) ;
END_IF;

IF (DoFTPTrigger = TRUE) THEN
    CASE Stage OF
        1 :                                                 // Execute BackupToM
emoryCard instruction.
            BackupToMemoryCard_instance(
                Execute     := TRUE) ;                      // Execution
            IF (BackupToMemoryCard_instance.Done = TRUE) THEN
                Stage       := INT#2;                       // To next stage
            ELSIF (BackupToMemoryCard_instance.Error = TRUE) THEN
```

```
                Stage        := INT#10;                          // Error end
            END_IF;
        2 :                                                     //Execute FTPPutFile
 instruction.
            FTPPutFile_instance(
                Execute      := TRUE,                           // Execution
                ConnectSvr   := FTPAddr,                        // Connected FTP ser
ver
                SvrDirName   := '/Backup/yyyy-mm-dd',           // FTP server direct
ory name
                LocalDirName := '/',                            // Local directory n
ame
                FileName     := '*.*',                          // File name
                PutFileResult:= PutResult) ;                    // Uploaded file res
ults
            IF (FTPPutFile_instance.Done = TRUE) THEN
                Stage        := INT#0;                          // Normal end
            ELSIF (FTPPutFile_instance.Error = TRUE) THEN
                Stage        := INT#20;                         // Error end
            END_IF;
        0 :                                                     // Processing after
normal end
            DoFTPTrigger     :=FALSE;
            Trigger          :=FALSE;
        ELSE                                                    // Processing after
error end
            DoFTPTrigger     :=FALSE;
            Trigger          :=FALSE;
    END_CASE;
END_IF;
```

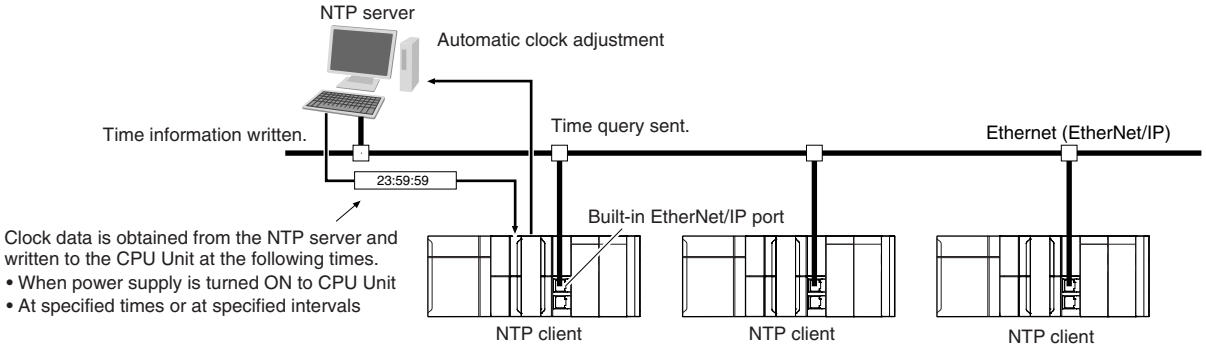
12

Automatic Clock Adjustment

12-1 Automatic Clock Adjustment

12-1-1 Overview

The built-in EtherNet/IP port reads clock information from the NTP server and updates the internal clock time in the CPU Unit at the specified time or at a specified interval after the power supply to the Controller is turned ON.

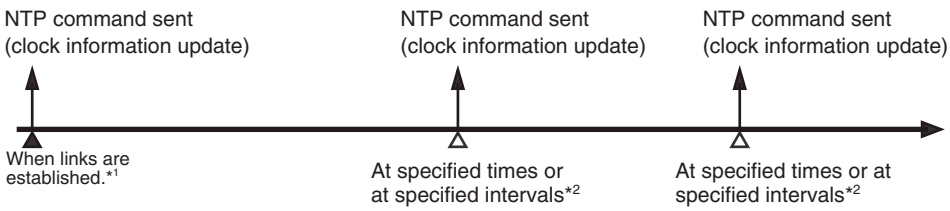


The NTP (Network Time Protocol) server is used to control the time on the LAN.

12-1-2 Specifications

Item	Specification	
Protocol	NTP	
Port No.	123 (UDP) However, you can change the port number in the Built-in EtherNet/IP Port Settings on the Sysmac Studio.	
Access to NTP server	Writes the clock information from the NTP server to the local CPU Unit.	Obtains the clock information from the NTP server set up on the Network, and applies the information obtained to the local CPU Unit.
NTP Operation Timing	Clock information is automatically updated at the following times if the NTP function is used. • After links are established when the power supply to the Controller is turned ON • At specified times or at specified intervals (according to the option selected for the **NTP operation timing**)	

Clock information is updated at the following times.



*1. This is performed when the **Get** Option is selected for the **NTP server clock information** in the **NTP Settings** Display.
*2. Depends on the option set for the **NTP operation timing** in the **NTP Settings** Display.

Additional Information

- NTP clock synchronization is normally performed as follows:
  - If the clock deviation is within 128 ms: The clock is synchronized every 0.5 ms.
  - If the clock deviation exceeds 128 ms: The clock is synchronized immediately.
- If the NTP operation timing is set for a specified time interval, the timing will not change even if the time in the CPU Unit is changed during operation.
  (For example, if the time interval is set to 60 minutes, the information is updated 60 minutes after the last time it was updated even if the time in the CPU Unit is changed.)

12-2 Procedure to Use the Automatic Clock Adjustment Function

12-2-1 Procedure

1 Make the basic settings.
Refer to *1-5 EtherNet/IP Communications Procedures* on page 1-29 for the basic operation flow.

2 Select **Controller Setup** - **Built-in EtherNet/IP Port Settings** on the Sysmac Studio.
Set the following on the **NTP Settings** Display.
- NTP server settings (required)
- NTP operation timing

3 Select **Synchronization** from the **Controller** Menu. The built-in EtherNet/IP port settings are transferred to the CPU Unit.

12-2-2 Settings Required for Automatic Clock Adjustment

The following Built-in EtherNet/IP Port Settings are made from the Sysmac Studio to use automatic clock adjustment.

Tab page	Setting		Setting conditions	Reference
NTP	NTP server clock information		Required.	page 4-13
	Port No.		Specified by user.*1	
	Server specifying method		Required	
		IP address	One of these must be set, depending on the **Server specification type** setting.	
		Host name		
	NTP operation timing		Required	
		Specify a Time	One of these must be set. (Set according to the **NTP operation timing**.)	
		Specify a time interval		
		Timeout time	Specified by user.*2	

*1. Required to change from the default value of 123.
*2. Required to change from the default value of 10 seconds.

Additional Information

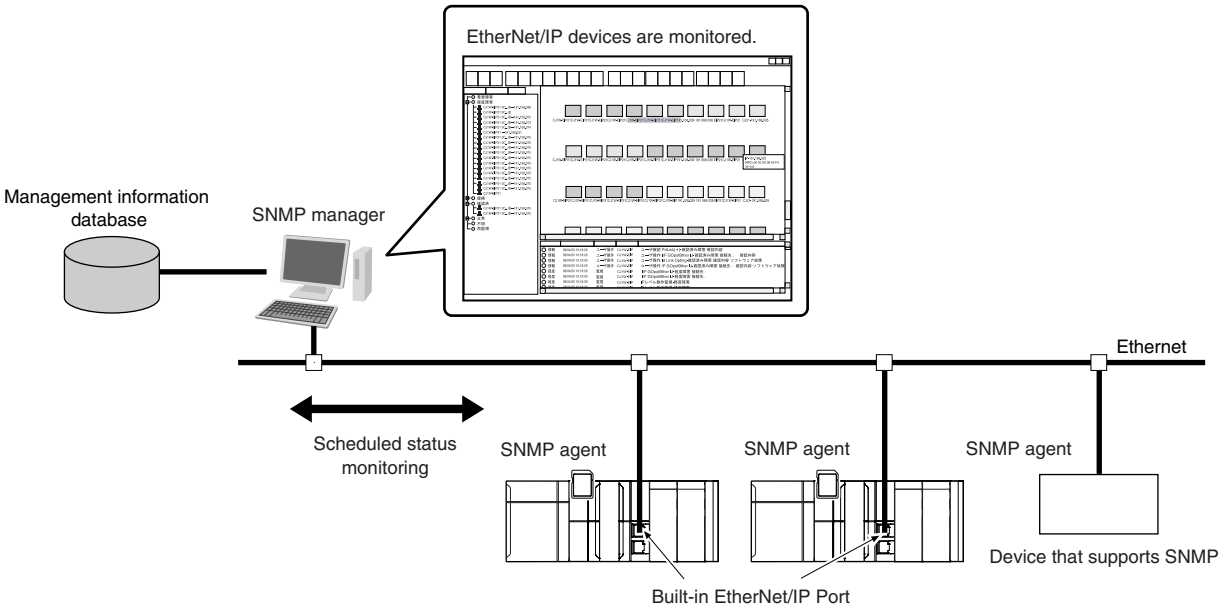
Make the settings in the **NTP Settings** Display if automatic clock adjustment is used. Refer to *4-4 **NTP Settings** Display* on page 4-13 for information on the **NTP Settings** Display.

13

SNMP Agent

13-1 SNMP Agent

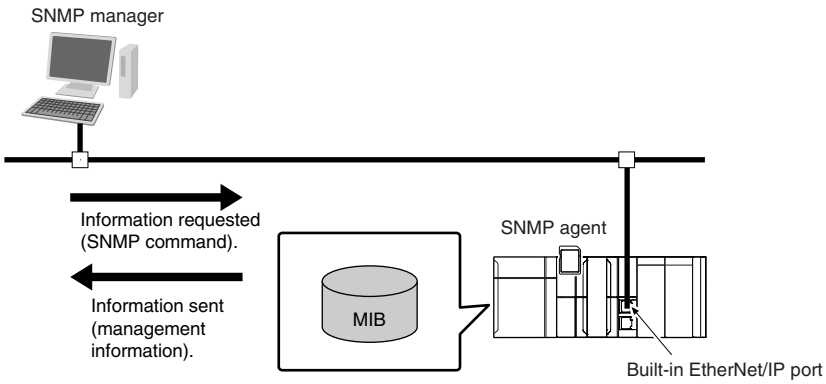
The SNMP (simple network management protocol) is a network management protocol.
You can use the SNMP to manage any network that consists of devices that support SNMP.
The server that manages the network is called the SNMP manager. The managed network devices are called SNMP agents.



13-1-1 Overview

SNMP Agent

The built-in EtherNet/IP port has its own management information called the MIB (management information base). This information can be provided to the SNMP manager.
The SNMP manager is software that gathers and processes information about devices on the SNMP network and provides that information to the network administrator.
You can use the SNMP manager to monitor the built-in EtherNet/IP port.

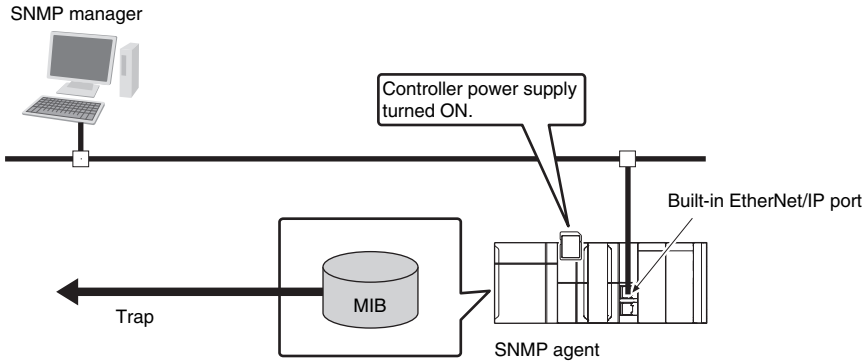


The SNMP manager has a SNMP command to request MIB information.

The built-in EtherNet/IP port SNMP agent function supports SNMPv1 (RFC1157) and SNMPv2C (RFC1901).
Use the SNMPv1 or SNMPv2C protocol to manage the built-in EtherNet/IP port with the SNMP manager. You can also use both the SNMPv1 and SNMPv2C protocols together at the same time.

SNMP Traps

When a failure or some other specific problem occurs, a status report called a trap is sent.
This enables monitoring changes in status even if the SNMP manager does not monitor the built-in EtherNet/IP port periodically.
However, traps use UDP. Therefore, you cannot check to see if the SNMP manager receives traps from the EtherNet/IP port.
Thus, depending on the network status, some traps may not reach the SNMP manager.

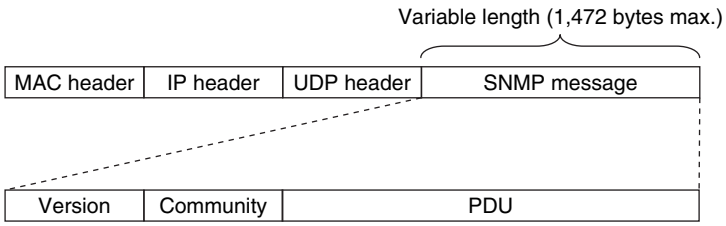


13-1-2 Specifications

Item	Specification
Protocol	SNMP
Agent	SNMPv1, SNMPv2c
MIB	MIB-II
Port No.	SNMP agent: 161 (UDP) SNMP trap: 162 (UDP) These can be changed in the Built-in EtherNet/IP Port Settings from the Sysmac Studio.
Timing of SNMP trap operation	Status reports are sent to the SNMP manager at the following times. • When the Controller is turned ON • When links are established • When an SNMP agent fails to be authorized
Supported MIB commands	GetRequest/GetNextRequest

13-1-3 SNMP Messages

The structure of SNMP messages is as follows:



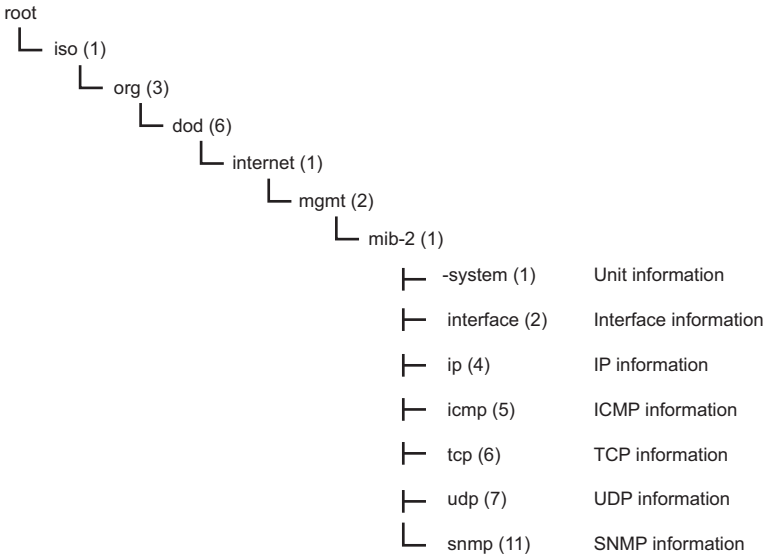
Item	Set value
Version	This value gives the SNMP version. SNMPv1: 0 SNMpv2c: 1
Community	Community name for verification
PDU	This depends on the PDU type.

13-1-4 MIB Specifications

This section describes the specifications of the MIB that is supported by the built-in EtherNet/IP port.

MIB System Diagram

The built-in EtherNet/IP port MIB consists of the following tree structure.



MIB Groups

MIB group		Stored information
Standard MIB	system group	The MIB for information related to the device.
	interfaces group	The MIB for information related to the interface.

MIB group			Stored information
	ip group	ip	The MIB for IP information.
		ipAddrTable	The MIB for addressing table information related to IP addresses.
		ipRouteTable	The MIB for information related to IP routing tables.
		ipNetToMediaTable	The MIB for information related to IP address conversion tables.
		ipForward	The MIB for information related to IP forwarding tables.
	icmp group		The MIB for ICMP information.
	tcp group	tcp	The MIB for TCP information.
	udp group	udp	The MIB for UDP information.
	snmp group	snmp	The MIB for SNMP information.

Detailed Descriptions of MIB Objects

● System Group

Subtree name	Standard [(identifier) attribute]	Support	Implementation specifications
sysDescr	[(1) RO] Device information (including hardware, OS, software names, and versions) ASCII characters only.	Supported	"OMRON Corporation" + CPU Unit model + CPU Unit version • CPU Unit model (example): NJ501-1200 • CPU Unit version (example): Version 1.0
sysObjectID	[(2) RO] Vendor OID. Tells where this device information was assigned in the private MIB.	Supported	1.3.6.1.4.1.16838.1.1025.5
sysUpTime	[(3) RO] The time elapsed since the system was started (unit: 1/100 s).	Supported	According to the standard.
sysContact	[(4) RW] How to contact the administrator and information on the administrator.	Supported	Set by the user.
sysName	[(5) RW] The name for management. Sets the full domain name of the device.	Supported	CPU Unit name
sysLocation	[(6) RW] The physical location of the device.	Supported	Set by the user.
sysServices	[(7) RO] The value of the provided service.	Supported	64

● Interfaces Group

Subtree name	Standard [(identifier) attribute]	Support	Implementation specifications
ifNumber	[(1) RO] The number of network interfaces.	Supported	• NX701 CPU Unit: 3 • NX102 CPU Unit: 3 • NX1P2 CPU Unit: 2 • NJ-series CPU Unit: 2
ifTable	[(2) NA] Interface entity table	---	---
ifEntry	[(1) NA] Row data for interface information The index is *ifIndex*.	---	---
ifIndex	[(1) RO] A number used to identify the interface.	Supported	• NX701 CPU Unit: 1 to 3 • NX102 CPU Unit: 1 to 3 • NX1P2 CPU Unit: 1 to 2 • NJ-series CPU Unit: 1 to 2
ifDescr	[(2) RO] Information related to the interface (includes manufacturer name, product name, and hardware interface version).	Supported	• NX701 CPU Unit: 10/100/1000M Gigabit Ethernet Port • NX102 CPU Unit: 10/100M Fast Ethernet Port • NX1P2 CPU Unit: 10/100M Fast Ethernet Port • NJ-series CPU Unit: 10/100M Fast Ethernet Port
ifType	[(3) RO] The type of interface classified according to the physical/link layer protocol directly under the network layer of the protocol stack.	Supported	ethernet-csmacd (6)
ifMtu	[(4) RO] MTU value The maximum size (in octets) of datagrams that can be sent and received through this interface.	Supported	1500

Subtree name	Standard [(identifier) attribute]	Support	Implementation specifications
ifSpeed	[(5) RO] Estimated bandwidth If a stable, accurate value cannot be obtained for the bandwidth, a nominal value is set instead.	Supported	• NX701 CPU Unit: 10000000/ 100000000/ 1000000000 • NX102 CPU Unit: 10000000/ 100000000 • NX1P2 CPU Unit: 10000000/ 100000000 • NJ-series CPU Unit: 10000000/ 100000000
ifPhysAddress	[(6) RO] MAC address The physical address under the network layer of the interface.	Supported	The MAC address of the EtherNet/IP port
ifAdminStatus	[(7) RW] The preferred status of the interface. You cannot send normal packets in the testing state. up (1) down (2) testing (3)	Supported	According to the standard.
ifOperStatus	[(8) RO] The current status of the interface. You cannot send normal packets in the testing state. up (1) down (2) testing (3)	Supported	According to the standard.
ifLastChange	[(9) RO] The sysUpTime (in 0.01 seconds) at the last change in ifOperStatus for this interface.	Supported	According to the standard.
ifInOctets	[(10) RO] The number of octets received through this interface. This includes framing characters.	Supported	According to the standard.
ifInUcastPkts	[(11) RO] The number of unicast packets reported to a higher level protocol.	Supported	According to the standard.
ifInNUcastPkts	[(12) RO] The number of non-unicast packets (broadcast or multicast packets) reported to a higher level protocol.	Supported	According to the standard.
ifInDiscards	[(13) RO] The number of packets that had no errors but could not be passed to a higher level protocol (i.e., the number of packets received but discarded due to a buffer overflow).	Supported	According to the standard.
ifInErrors	[(14) RO] The number of packets discarded because they contained errors.	Supported	According to the standard.

Subtree name	Standard [(identifier) attribute]	Support	Implementation specifications
ifInUnknownProtos	[(15) RO] The number of packets received, but discarded because they were of an illegal or unsupported protocol. For example, Ethernet packets did not have IP set for the field that identifies their higher level protocol.	Supported	According to the standard.
ifOutOctets	[(16) RO] The number of octets of packets sent through this interface. This includes framing characters.	Supported	According to the standard.
ifOutUcastPkts	[(17) RO] The number of unicast packets sent by higher level protocols. This includes discarded packets and unsent packets.	Supported	According to the standard.
ifOutNUcastPkts	[(18) RO] The number of non-unicast packets sent by higher level protocols. This includes discarded packets and unsent packets.	Supported	According to the standard.
ifOutDiscards	[(19) RO] The number of packets that had no errors but were discarded in the sending process (due to a send buffer overflow, etc.).	Supported	According to the standard.
ifOutErrors	[(20) RO] The number of packets that could not be sent because of an error.	Supported	According to the standard.
ifOutQLen	[(21) RO] The size of the send packet queue (i.e., the number of packets).	Supported	Always 0.
ifSpecific	[(22) RO] The object ID that represents a reference to the media-specific MIB for the interface. For example, for Ethernet, set the object ID of the MIB that defines Ethernet. If there is no information, set { 0.0 }.	Supported	0.0

● Ip Group: Ip

Subtree name	Standard [(identifier) attribute]	Support	Implementation specifications
ipForwarding	[(1) RW] Indicates if the device operates as a gateway. IP gateways can transfer datagrams, but IP hosts can perform only source routing. Some nodes take only one of these values. Therefore, if you attempt to change this object from the SNMP Manager, a badValue error is returned. forwarding (1) not-forwarding (2)	Supported	• NX701 CPU Unit: forwarding (1) • NX102 CPU Unit: forwarding (1), not-forwarding (2) • NX1P2 CPU Unit: not-forwarding (2) • NJ-series CPU Unit: not-forwarding (2) Depends on the settings in **Built-in EtherNet/IP Port Settings** - **TCP/IP Settings** - **Port Forward** on the Sysmac Studio.
IpDefaultTTL	[(2) RW] The default value set for the IP header TTL if no TTL value was given by the transport layer protocol.	Supported	64
IpInReceives	[(3) RO] The number of all IP datagrams that reached the interface, including errors.	Supported	According to the standard.
IpInHdrErrors	[(4) RO] The number of received datagrams that were discarded because of an IP header error (checksum error, version number error, format error, TTL error, IP option error, etc.).	Supported	According to the standard.
IpInAddrErrors	[(5) RO] The number of packets that were discarded because the destination address in the IP header was not valid.	Supported	According to the standard.
ipForwDatagrams	[(6) RO] The number of IP datagrams that were transferred to their final destination. If this node does not operate as an IP gateway, this is the number of datagrams that were successfully transferred through source routing.	Supported	According to the standard.
ipInUnknownProtos	[(7) RO] The number of IP datagrams that were received but discarded because they were of an unsupported or unrecognized protocol.	Supported	According to the standard.
ipInDiscards	[(8) RO] The number of IP datagrams that could have continued to be processed without any problems, but were discarded (for example, because of insufficient buffer space).	Supported	According to the standard.

Subtree name	Standard [(identifier) attribute]	Support	Implementation specifications
ipInDelivers	[(9) RO] The number of datagrams delivered to an IP user protocol (any higher level protocol, including ICMP).	Supported	According to the standard.
ipOutRequests	[(10) RO] The number of times a send request was made for an IP datagram by a local IP user protocol (any higher level protocol, including ICMP). This counter does not include ipForwDatagrams.	Supported	According to the standard.
ipOutDiscards	[(11) RO] The number of IP datagrams that could have been sent without any problems, but were discarded (for example, because of insufficient buffer space).	Supported	According to the standard.
ipOutNoRoutes	[(12) RO] The number of IP datagrams that were discarded because there was no transmission path. This counter includes datagrams that attempted to be sent through ipForwDatagrams, but were discarded because they were set with no-route. This value indicates the number of datagrams that could not be transferred because the default gateway was down.	Supported	According to the standard.
ipReasmTimeout	[(13) RO] The maximum number of seconds to wait to receive all IP datagrams for reassembly if a fragmented IP datagram is received.	Supported	60 s
ipReasmReqds	[(14) RO] The number of IP datagrams received that require reassembly. There is a flag in the IP header that indicates if the datagram is fragmented. You can use that flag to identify fragments.	Supported	According to the standard.
ipReasmOKs	[(15) RO] The number of IP datagrams received that were successfully reassembled.	Supported	According to the standard.
ipReasmFails	[(16) RO] The number of IP datagrams received that were not successfully reassembled.	Supported	According to the standard.
ipFragOKs	[(17) RO] The number of IP datagrams that were successfully fragmented.	Supported	According to the standard.
ipFragFails	[(18) RO] The number of IP datagrams that were not successfully fragmented. (For example, because the Don't Fragment flag was set for the IP datagram.)	Supported	According to the standard.
ipFragCreates	[(19) RO] The number of IP datagrams created as a result of fragmentation.	Supported	According to the standard.

Subtree name	Standard [(identifier) attribute]	Support	Implementation specifications
ipAddrTable	[(20) NA] An address information table for IP addresses.	---	---
ipAddrEntry	[(1) NA] Row data of address information for IP addresses. The index is *ipAdEntAddr*.	---	---
ipAdEntAddr	[(1) RO] The IP address.	Supported	According to the standard.
ipAdEntIfIndex	[(2) RO] The index value of the interface that this entry applies to. This is the same value as ifIndex.	Supported	According to the standard.
ipAdEntNetMask	[(3) RO] The subnet mask for the IP address of this entry.	Supported	According to the standard.
ipAdEntBcastAddr	[(4) RO] The value of the least significant bit of the address when an IP broadcast is sent. An address represented by all 1 bits is used for broadcasting as an Internet standard. In that case, this value is always 1.	Supported	According to the standard.
ipAdEntReasmMaxSize	[(5) RO] The maximum IP packet size that can be reassembled from IP fragmented input IP datagrams received through the interface.	Supported	According to the standard.
ipRouteTable	[(21) NA] The IP routing table for this entity.	---	---

Subtree name	Standard [(identifier) attribute]	Support	Implementation specifications
ipRouteEntry	[(1) NA] Route information for a specific destination. The index is *ipRouteDest*.	---	---
ipRouteDest	[(1) RW] The destination IP address for this route. A value of 0.0.0.0 for this entry indicates the default route.	Supported	According to the standard.
ipRouteIfIndex	[(2) RW] The ID number of the interface required to send to the next destination host in this route. This ID number is the same number as ifIndex, which is used to identify the interface.	Supported	According to the standard.
ipRouteMetric1	[(3) RW] The primary routing metric for this route. This value is determined based on the protocol specified in ipRouteProto. Set to -1 if you do not want to use this metric (this is also the same for ipRouteMetric 2 through 4).	Supported	According to the standard.
ipRouteMetric2	[(4) RW] The alternative routing metric for this route.	Supported	According to the standard.
ipRouteMetric3	[(5) RW] The alternative routing metric for this route.	Supported	According to the standard.
ipRouteMetric4	[(6) RW] The alternative routing metric for this route.	Supported	According to the standard.
ipRouteNextHop	[(7) RW] The IP address of the next hop in this route (for routes connected by a broadcast or media, this is the agent address or address of that interface).	Supported	According to the standard.
ipRouteType	[(8) RW] The type of route. other (1): Not any of the following types. invalid (2): An invalid route. direct (3): A direct connection. indirect (4): An indirect connection (not connected to LOCAL).	Supported	According to the standard.

Subtree name			Standard [(identifier) attribute]	Support	Implementation specifications
		ipRouteProto	[(9) RO] This is the routing mechanism used to determine routes. Some values correspond to gateway routing protocols, but be aware that the host may not support those protocols. other (1): Other than the following items. local (2): A route set on the local machine. netmgmt (3): A route set by network management. icmp (4): A route set by an ICMP redirect or some other ICMP function. egp (5): EGP The following are gateway protocols: ggp (6): GGP hello (7): HELLO rip (8): RIP is-is (9) es-is (10) ciscolgrp (11) bbnSpflgp (12) ospf (13): OSPF bgp (14)	Supported	According to the standard.
		ipRouteAge	[(10) RW] The elapsed time since this route was updated (in seconds).	Supported	Always 0.
		ipRouteMask	[(11) RW] The subnet mask value in relation to ipRouteDest. On systems that do not support a custom subnet mask value, this value is based on the address class of the ipRouteDest field. If ipRouteDest is 0.0.0.0, this value is also 0.0.0.0.	Supported	According to the standard.
		ipRouteMetric5	[(12) RW] The alternative routing metric.	Supported	According to the standard.
		ipRouteInfo	[(13) RO] The MIB object ID for the routing protocol used by this route. If not defined, set to {0.0}.	Supported	0.0
	ipNetToMediaTable		[(22) NA] The IP address conversion table used to map IP addresses to physical addresses.	---	---

Subtree name	Standard [(identifier) attribute]	Support	Implementation specifications
ipNetToMediaEntry	[(1) NA] Row data for the conversion table. The indices are *ipNetToMediaIfIndex* and *ipNetToMediaNetAddress*.	---	---
ipNetToMediaIfIndex	[(1) RW] The interface ID number for this entry. The value of ifIndex is used for this value.	Supported	According to the standard.
ipNetToMediaPhysAddress	[(2) RW] The media-dependent physical address.	Supported	According to the standard.
ipNetToMediaNetAddress	[(3) RW] The IP address that corresponds to the media-dependent physical address.	Supported	According to the standard.
ipNetToMediaType	[(4) RW] The address conversion method. other (1): A method other than the following items. invalid (2): An invalid value. dynamic (3): Dynamic conversion. static (4): Static conversion.	Supported	According to the standard.
ipRoutingDiscards	[(23) RO] The number of routing entries that were valid but discarded. For example, if there was not enough buffer space because of other routing entries.	Supported	According to the standard.

● Ip Group: Icmp

Name	Standard [(identifier) attribute]	Support	Implementation specifications
icmpInMsgs	[(1) RO] The total number of received ICMP messages. This includes messages counted by icmpInErrors.	Supported	According to the standard.
icmpInErrors	[(2) RO] The number of received ICMP message errors. (Checksum errors, frame length errors, etc.)	Supported	According to the standard.
icmpInDestUnreachs	[(3) RO] The number of Destination Unreachable messages received.	Supported	According to the standard.
icmpInTimeExcds	[(4) RO] The number of Time Exceed messages received.	Supported	According to the standard.
icmpInParmProbs	[(5) RO] The number of Parameter Problem messages received.	Supported	According to the standard.
icmpInSrcQuenchs	[(6) RO] The number of Source Quench messages received.	Supported	According to the standard.
icmpInRedirects	[(7) RO] The number of Redirect messages received.	Supported	According to the standard.

Name	Standard [(identifier) attribute]	Support	Implementation specifications
icmpInEchos	[(8) RO] The number of Echo (request) messages received.	Supported	According to the standard.
icmpInEchoReps	[(9) RO] The number of Echo Reply messages received.	Supported	According to the standard.
icmpInTimestamps	[(10) RO] The number of Timestamp messages received.	Supported	According to the standard.
icmpInTimestampReps	[(11) RO] The number of Timestamp Reply messages received.	Supported	According to the standard.
icmpInAddrMasks	[(12) RO] The number of Address Mask Request messages received.	Supported	According to the standard.
icmpInAddrMaskReps	[(13) RO] The number of Address Mask Reply messages received.	Supported	According to the standard.
icmpOutMsgs	[(14) RO] The total number of ICMP messages sent. This includes messages counted by icmpOutErrors.	Supported	According to the standard.
icmpOutErrors	[(15) RO] The number of ICMP messages that could not be sent because of an error.	Supported	According to the standard.
icmpOutDestUnreachs	[(16) RO] The number of Destination Unreachable messages sent.	Supported	According to the standard.
icmpOutTimeExcds	[(17) RO] The number of Time Exceed messages sent.	Supported	According to the standard.
icmpOutParmProbs	[(18) RO] The number of Parameter Problem messages sent.	Supported	According to the standard.
icmpOutSrcQuenchs	[(19) RO] The number of Source Quench messages sent.	Supported	According to the standard.
icmpOutRedirects	[(20) RO] The number of Redirect messages sent.	Supported	According to the standard.
icmpOutEchos	[(21) RO] The number of Echo (request) messages sent.	Supported	According to the standard.
icmpOutEchoReps	[(22) RO] The number of Echo Reply messages sent.	Supported	According to the standard.
icmpOutTimestamps	[(23) RO] The number of Timestamp messages sent.	Supported	According to the standard.
icmpOutTimestampReps	[(24) RO] The number of Timestamp Reply messages sent.	Supported	According to the standard.
icmpOutAddrMasks	[(25) RO] The number of Address Mask Request messages sent.	Supported	According to the standard.

Name	Standard [(identifier) attribute]	Support	Implementation specifications
icmpOutAddrMaskReps	[(26) RO] The number of Address Mask Reply messages sent.	Supported	According to the standard.

● Ip Group: Tcp

Name	Standard [(identifier) attribute]	Support	Implementation specifications
tcpRtoAlgorithm	[(1) RO] The algorithm used to determine the timeout value for resending. other (1): Other than the following items. constant (2): A constant RTO value. rsre (3): The algorithm specified by the MIL-STD-1778 standard. vanj (4): The Van Jacobson algorithm.	Supported	According to the standard.
tcpRtoMin	[(2) RO] The minimum resend timeout value (in 0.01 s). This value depends on the algorithm used to determine the resend timeout value.	Supported	According to the standard.
tcpRtoMax	[(3) RO] The maximum resend timeout value (in 0.01 s). This value depends on the algorithm used to determine the resend timeout value.	Supported	According to the standard.
tcpMaxConn	[(4) RO] The total number of supported TCP connections. If the maximum number of connections is dynamic, this value is -1.	Supported	According to the standard.
tcpActiveOpens	[(5) RO] The number of times the TCP connection changed from the CLOSE state directly to the SYN-SENT state. (Active connection establishment.)	Supported	According to the standard.
tcpPassiveOpens	[(6) RO] The number of times the TCP connection changed from the LISTEN state directly to the SYN-RCVD state. (Passive connection establishment.)	Supported	According to the standard.
tcpAttemptFails	[(7) RO] The total number of times the TCP connection changed from the SYN-SENT or SYN-RCVD state directly to the CLOSE state and from the SYN-RCVD state directly to the LISTEN state.	Supported	According to the standard.
tcpEstabResets	[(8) RO] The number of times the TCP connection changed from the ESTABLISHED or the CLOSE-WAIT state directly to the CLOSE state.	Supported	According to the standard.

Name	Standard [(identifier) attribute]	Support	Implementation specifications
tcpCurrEstab	[(9) RO] The total number of TCP connections currently in the ESTABLISHED or the CLOSE-WAIT state.	Supported	According to the standard.
tcpInSegs	[(10) RO] The total number of received segments. This includes the number of error segments.	Supported	According to the standard.
tcpOutSegs	[(11) RO] The total number of sent segments. This includes the number of segments for the current connection, but does not include the number of segments for resent data only.	Supported	According to the standard.
tcpRetransSegs	[(12) RO] The total number of resent segments.	Supported	According to the standard.
tcpConnTable	[(13) NA] The information table specific to the TCP connection.	---	---
tcpConnEntry	[(1) NA] Entry information related to a specific TCP connection. This value is deleted if the connection changes to the CLOSE state. The indices are *tcpConnLocalAddress*, *tcpConnLocalPort*, *tcpConnRemAddress*, and *tcpConnRemPort*.	---	---
tcpConnState	[(1) RW] The status of the TCP connection. closed (1) listen (2) synSent (3) synReceived (4) established (5) finWait1 (6) finWait2 (7) closeWait (8) lastAck (9) closing (10) timeWait (11)	Supported	According to the standard.
tcpConnLocalAddress	[(2) RO] The local IP address of this TCP connection. A value of 0.0.0.0 is used for connections in the LISTEN state that accept connections from any IP interface related to the node.	Supported	According to the standard.
tcpConnLocalPort	[(3) RO] The local port number for this TCP connection.	Supported	According to the standard.
tcpConnRemAddress	[(4) RO] The remote IP address for this TCP connection.	Supported	According to the standard.
tcpConnRemPort	[(5) RO] The remote port number for this TCP connection.	Supported	According to the standard.

Name	Standard [(identifier) attribute]	Support	Implementation specifications
tcpInErrs	[(14) RO] The total number of error segments received (TCP checksum errors, etc.).	Supported	According to the standard.
tcpOutRsts	[(15) RO] The number of segments sent with the RST flag (the number of times the TCP connection was reset).	Supported	According to the standard.

● Ip Group: Udp

Name	Standard [(identifier) attribute]	Support	Implementation specifications
udpInDatagrams	[(1) RO] The total number of UDP datagrams (i.e., the number of packets) sent to the UDP user.	Supported	According to the standard.
udpNoPorts	[(2) RO] The number of UDP datagrams that were received but did not start an application at the destination port.	Supported	According to the standard.
udpInErrors	[(3) RO] The number of UDP datagrams that were not sent to a higher level protocol for a reason other than udpNoPorts.	Supported	According to the standard.
udpOutDatagrams	[(4) RO] The total number of sent UDP datagrams.	Supported	According to the standard.
udpTable	[(5) NA] The information table for the UDP listener.	---	---
udpEntry	[(1) NA] An entry related to a specific UDP listener. The indices are *udpLocalAddress* and *udpLocalPort*.	---	---
udpLocalAddress	[(1) RO] The local IP address of this UDP listener. A value of 0.0.0.0 is used for UDP listeners that accept datagrams from any IP interface related to the node.	Supported	According to the standard.
udpLocalPort	[(2) RO] The local port number for this UDP listener.	Supported	According to the standard.

● Ip Group: Snmp

Name	Standard [(identifier) attribute]	Support	Implementation specifications
snmpInPkts	[(1) RO] The total number of SNMP messages received.	Supported	According to the standard.
snmpOutPkts	[(2) RO] The total number of SNMP messages sent.	Supported	According to the standard.
snmpInBadVersions	[(3) RO] The total number of messages received of an unsupported version.	Supported	According to the standard.

Name	Standard [(identifier) attribute]	Support	Implementation specifications
snmpInBadCommunityNames	[(4) RO] The total number of messages received from an unregistered community.	Supported	According to the standard.
snmpInBadCommunityUses	[(5) RO] The total number of messages received that specify an operation that is not allowed by that community.	Supported	According to the standard.
snmpInASNParseErrs	[(6) RO] The total number of messages received that resulted in an ASN.1 error or BER error during decoding.	Supported	According to the standard.
snmpInTooBigs	[(8) RO] The total number of PDUs received with an error status of tooBig.	Supported	According to the standard.
snmpInNoSuchNames	[(9) RO] The total number of PDUs received with an error status of noSuchName.	Supported	According to the standard.
snmpInBadValues	[(10) RO] The total number of PDUs received with an error status of badValue.	Supported	According to the standard.
snmpInReadOnlys	[(11) RO] The total number of PDUs received with an error status of readOnly.	Supported	According to the standard.
snmpInGenErrs	[(12) RO] The total number of PDUs received with an error status of genErr.	Supported	According to the standard.
snmpInTotalReqVars	[(13) RO] The total number of MIB objects read normally after receiving GetRequest or GetNextRequest.	Supported	According to the standard.
snmpInTotalSetVars	[(14) RO] The total number of MIB objects updated normally after receiving SetRequest.	Supported	According to the standard.
snmpInGetRequests	[(15) RO] The total number of GetRequest PDUs received.	Supported	According to the standard.
snmpInGetNexts	[(16) RO] The total number of GetNextRequest PDUs received.	Supported	According to the standard.
snmpInSetRequests	[(17) RO] The total number of SetRequest PDUs received.	Supported	According to the standard.
snmpInGetResponses	[(18) RO] The total number of GetResponse PDUs received.	Supported	According to the standard.
snmpInTraps	[(19) RO] The total number of trap PDUs received.	Supported	According to the standard.
snmpOutTooBigs	[(20) RO] The total number of PDUs sent with an error status of tooBig.	Supported	According to the standard.

Name	Standard [(identifier) attribute]	Support	Implementation specifications
snmpOutNoSuch-Names	[(21) RO] The total number of PDUs sent with an error status of noSuchName.	Supported	According to the standard.
snmpOutBadValues	[(22) RO] The total number of PDUs sent with an error status of badValue.	Supported	According to the standard.
snmpOutGenErrs	[(24) RO] The total number of PDUs sent with an error status of genErr.	Supported	According to the standard.
snmpOutGetRequests	[(25) RO] The total number of GetRequest PDUs sent.	Supported	According to the standard.
snmpOutGetNexts	[(26) RO] The total number of GetNextRequest PDUs sent.	Supported	According to the standard.
snmpOutSetRequests	[(27) RO] The total number of SetRequest PDUs sent.	Supported	According to the standard.
snmpOutGetResponses	[(28) RO] The total number of GetResponse PDUs sent.	Supported	According to the standard.
snmpOutTraps	[(29) RO] The total number of trap PDUs sent.	Supported	According to the standard.
snmpEnableAuthen-Traps	[(30) RW] Determines if the agent generates verification failed traps. enabled (1) disabled (2)	Supported	According to the standard.

13-2 Procedure to Use the SNMP Agent

13-2-1 Procedures

1. Make the basic settings.
   Refer to *1-5 EtherNet/IP Communications Procedures* on page 1-29 for the basic operation flow.
2. Select **Controller Setup** - **Built-in EtherNet/IP Port Settings** on the Sysmac Studio.
   Make the following settings on the **SNMP Settings** Display or the **SNMP Trap Settings** Display.
   - SNMP Service
   - Recognition 1
   - Recognition 2
3. Select **Transfer to Controller** from the **Controller** Menu and click the **Yes** Button. The built-in EtherNet/IP port settings are transferred to the CPU Unit.

Precautions for Correct Use

If the **Use** Option is selected for Packet Filter of the built-in EtherNet/IP port, allow packets from the SNMP manager. If they are not permitted, communication with SNMP manager is not possible. For the details on the settings, refer to *1-5 EtherNet/IP Communications Procedures* on page 1-29.

13-2-2 Settings Required for the SNMP Agent

The following Built-in EtherNet/IP Port Settings are made from the Sysmac Studio to use the SNMP agent.

Tab page	Setting		Setting conditions	Reference
SNMP Settings	SNMP service		Required.	page 4-15
	Port No.		Specified by user. Required to change from the default value of 161.	
	Contact, location		Specified by user.	
	Send a recognition trap		Specified by user. Select this check box to send a recognition trap if there is access from an SNMP manager that is not specified (Access other than Recognition 1 and 2).	
	Recognition 1 and Recognition 2		Specified by user. Make these settings to permit access by only certain SNMP managers.	page 4-16
		IP address		
		Host name		
		Community name		

Tab page	Setting		Setting conditions	Reference
SNMP Trap Settings	SNMP trap		Required	page 4-17
	Port No.		Specified by user. Required to change from the default value of 162.	
	Trap 1 and trap 2			page 4-17
		IP address	Required Set an IP address or a host name as the SNMP trap destination.	
		Host name		
		Community name	Specified by user.	
		Version	Required Set the version of the SNMP manager.	

Additional Information

Make the settings in the **SNMP Settings** Display and the **SNMP Trap Settings** Display if the SNMP agent is used.
Refer to *4-5 **SNMP Settings** Display* on page 4-15 for information on the **SNMP Settings** Dialog Box. Refer to *4-6 **SNMP Trap Settings** Display* on page 4-17 for information on the **SNMP Trap** Dialog Box.

14

Communications Performance and Communications Load

14-1 Communications System

14-1-1 Tag Data Link Communications Method

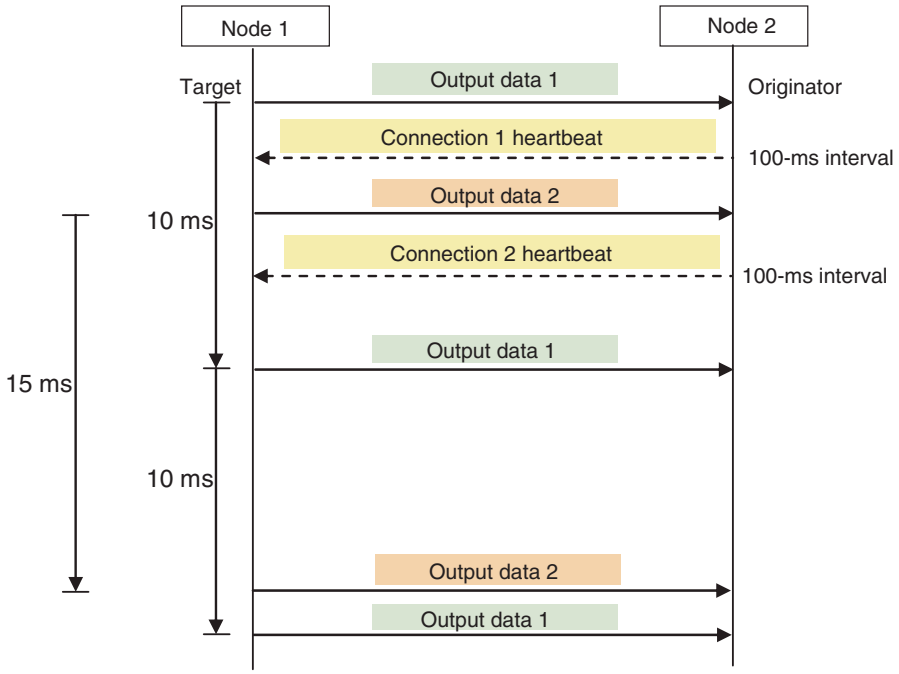
Requested Packet Interval (RPI) Settings

In tag data links for the built-in EtherNet/IP port, the data transmission period is set for each connection as the RPI.
The target device sends data (i.e., output tags) based on the specified RPI, regardless of the number of nodes.
Also, the heartbeat frame is sent from the originator to the target device for each connection. The target device uses the heartbeat to check if any errors have occurred in the connection with the originator. The data transmission period of the heartbeat frame depends on the RPI settings.

Heartbeat Frame Transmission Period

- If packet interval is shorter than 100ms, the heartbeat frame transmission period is 100ms.
- If packet interval is equal to or larger than 100ms, the heartbeat frame transmission period is the same as the RPI.

Example)
In this example, two tag data link connections are set for node 2 (the originator) and node 1 (the target).
The RPI for output data 1 is set to 10ms.
The RPI for output data 2 is set to 15 ms.
In this case, output data 1 is sent from node 1 to node 2 every 10 ms, and output data 2 is sent from node 1 to node 2 every 15 ms, as shown in the following diagram.
Also, data is sent from node 2 (the originator) to node 1 (the target) with a heartbeat of 100 ms for connection 1 and a heartbeat of 100 ms for connection 2.



Requested Packet Interval (RPI) and Bandwidth Usage (PPS)

The number of packets transferred each second is called the used bandwidth, or PPS (packets per second).
The PPS is calculated from the RPI and heartbeat for each connection as follows:

PPS for a connection (pps)
= (1,000/RPI (ms)) + (1,000/Heartbeat transmission period (ms))

Use the following equation to calculate the total number of packets transferred by each built-in EtherNet/IP port (Unit) in 1 second.
Total PPS for the built-in EtherNet/IP port = Total PPS of originator connections + Total PPS of target connections (*)

* Connections set as target connections must be added, too.

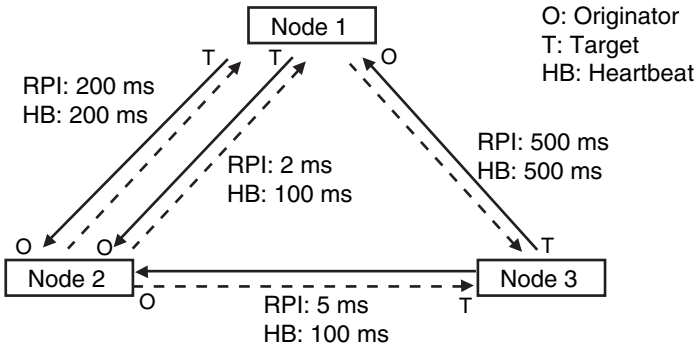
The following shows the maximum number of packets that each CPU Unit can send and receive per second via the built-in EtherNet/IP port through tag data links (i.e., the allowed communications bandwidth per Unit). You need to consider these values when configuring connections.

- NX701 CPU Unit: 40,000 pps
- NX102 CPU Unit: 12,000 pps
- NX1P2 CPU Unit: 3,000 pps
- NJ-series CPU Unit: 3,000 pps (*)

*Note that the bandwidth allowed for NJ-series CPU Units with unit version 1.00 to 1.02 is 1,000 pps.

Example)
Node 1 has an originator connection with the receive RPI of 500 ms, and two target connections with the send RPIs of 200 ms and 2 ms.
Node 2 has three originator connections with the receive RPIs of 200 ms, 2 ms, and 5 ms.

Node 3 has two target connections with the send RPIs of 5 ms and 1 ms.



The total PPS of each node is calculated as follows:

- Total PPS of the Unit Node 1
  = 1,000/200 ms + 1,000/2 ms+ 1,000/500 ms (for data)
  + 1,000/200 ms + 1,000/100 ms + 1,000/500 ms (for heartbeat)
  = 524 pps
- Total PPS of the Unit Node 2
  = 1,000/200 ms + 1,000/2 ms + 1,000/5 ms (for data)
  + 1,000/200 ms + 1,000/100 ms + 1,000/100 ms (for heartbeat)
  = 730 pps
- Total PPS of the Unit Node 3
  = 1,000/5 ms + 1,000/500 ms (for data)
  + 1,000/100 ms + 1,000/500 ms (for heartbeat)
  = 214 pps

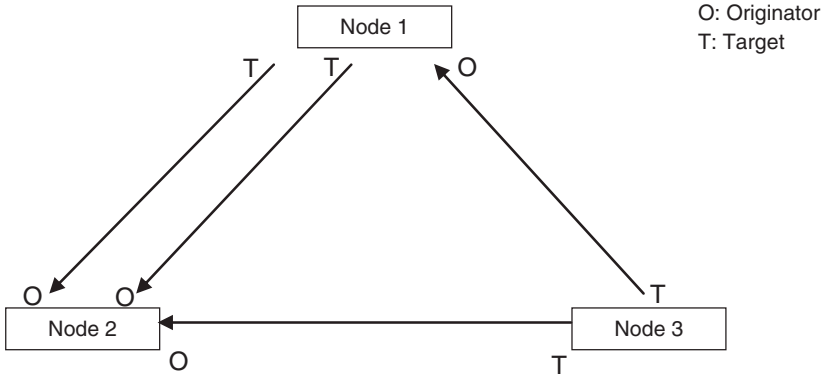
In this example, the total PPS of each Unit is below the maximum bandwidth allowed for the Unit, so data transmission can be successfully performed.

14-1-2 Calculating the Number of Connections

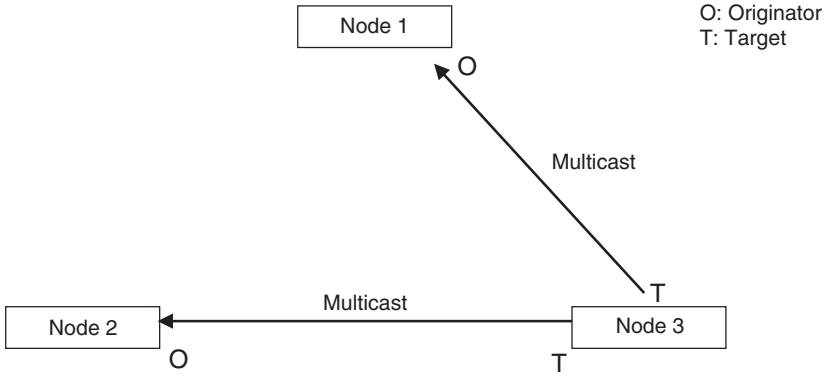
The maximum number of connections per built-in EtherNet/IP port on a CPU Unit is as follows.

- NX701 CPU Unit: 256
- NX102 CPU Unit: 32
- NX1P2 CPU Unit: 32
- NJ-series CPU Unit: 32

The maximum number of connections for a Unit should not be exceeded by the total number of originator connections, which the Unit opens, and target connections, which other nodes open to the Unit.
Example)
Node 1 has two target connections with Node 2, and opens one originator connection to Node 3. So, Node 1 has three connections in total.
Node 2 opens two originator connections to Node 1, and one originator connection to Node 3. So, Node 2 has three connections in total.
Node 3 has one target connection with Node 1, and one target connection with Node 2. So, Node 3 has two connections in total.
In either case, the connections can be successfully opened since the total number of connections is below the maximum number for a built-in EtherNet/IP port, as shown above.



If multicast is specified for data transmission and the node sends out just one multicast packet to other nodes, it requires respective connections for them.
Example)
Node 3 sends out one multicast packet to Node 1 and Node 2. Node 3 has one target connection with Node 1, and one target connection with Node 2, requiring two connections in total.
You need to keep in mind that the number of required connections is the same, whether multicast or unitcast is specified for the communications.

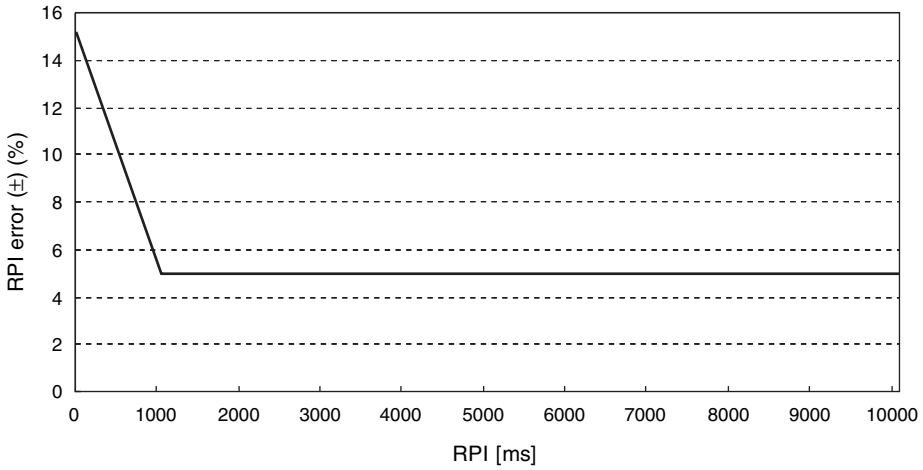


14-1-3 Packet Interval (RPI) Accuracy

A send processing delay occurs in a built-in EtherNet/IP port when data packets are sent based on a packet interval (RPI).
This delay varies within the RPI error margin as shown below, so the send processing may be delayed for the maximum value for each RPI.

Packet interval (RPI)	RPI error margin (±) (%)
0.5 to 1,000 ms (NX701 CPU Unit) 1 to 1,000 ms (NX102 CPU Unit) 2 to 1,000 ms (NX1P2 CPU Unit) 1 to 1,000 ms (NJ-series CPU Unit)*1	15 − (RPI [ms]/100)
1,000 to 10,000 ms	5% of the RPI

*1. Note that the RPI for a NJ-series CPU Unit with unit version 1.00 to 1.02 is between 10 ms to 1,000 ms.


16
14
12
10
8
6
4
2
0
RPI error (±) (%)
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
10000
RPI [ms]

14-2 Adjusting the Communications Load

In an Ethernet network using an Ethernet switch, the network bandwidth is not shared by all of the nodes; independent transmission paths are established between individual nodes through the Ethernet switch.
A dedicated communications buffer is established in the Ethernet switch for communications between the nodes, and full-duplex communications (simultaneous transmission and reception) are performed asynchronously with other transmission paths. The communications load in other transmission paths does not affect communications, so packet collisions do not occur and stable, high-speed communications can be performed.
The Ethernet switch functions shown in the following table determine the performance of tag data links.

Item	Description
Buffer capacity	This is the amount of data that can be buffered when packets accumulate at the Ethernet switch.
Multicast filtering	This function transfers multicast packets to specific nodes only.
QoS function	This function performs priority control on packet transfers.

The following table shows the setting ranges of the tag data link settings that can be made for a built-in EtherNet/IP port.

Item	Description	NX-series CPU Unit			NJ-series CPU Unit	
		NX701	NX102	NX1P2	Unit version 1.00 to 1.02	Unit version 1.03 or later
Network bandwidth	Physical Ethernet baud rate	1,000 Mbps	100 Mbps or 10 Mbps			
Allowed tag data link communications bandwidth	Maximum number of tag data link packets that can be processed in 1 second (pps: packets per second)	40,000 pps max. (total of 40,000 pps with two ports)	12,000 pps max. (total of 12,000 pps with two ports)	3,000 pps max.	1,000 pps max.	3,000 pps max.
Connection resources	Number of connections that can be established	256 max. (total of 512 with two ports)	32 max. (total of 64 with two ports)	32 max.		
Packet interval (RPI: Requested Packet Interval)	Refresh period for tag data	0.5 to 10,000 ms in 0.5-ms increments	1 to 10,000 ms in 1-ms increments	2 to 10,000 ms in 1-ms increments	10 to 10,000 ms in 1-ms increments	1 to 10,000 ms in 1-ms increments

When the tag data link settings exceed the capabilities of the Ethernet switch to be used, increase the packet interval (RPI) value for adjustment.
Particularly when you configure the settings with an Ethernet switch that does not support multicast filtering, you need to consider that multicast packets will be sent to all the nodes on the network without setting the connections.

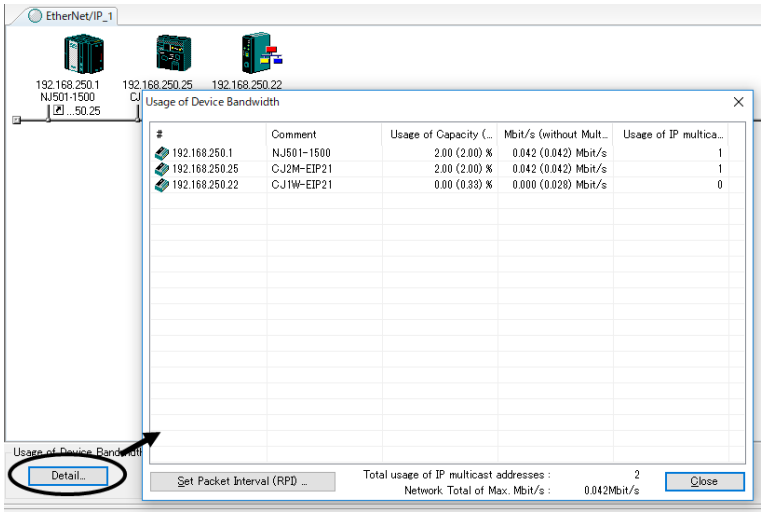
Additional Information

If you select **Multi-cast Connection** for the connection type in the connection settings on the Network Configurator, multicast packets are used. If you select **Point to Point Connection** for the connection type, multicast packets are not used.

If required tag data link performance cannot be achieved with the Ethernet switch, re-evaluate the overall network configuration and take necessary measures such as selecting a different Ethernet switch or splitting the network.
The following sections show how to check the device bandwidth used by the tag data links in the designed network, and how to set appropriate values.

14-2-1 Checking Bandwidth Usage for Tag Data Links

The Network Configurator can display the bandwidth to be actually used for tag data links at each built-in EtherNet/IP port, based on the connections set in the network configuration.
The device bandwidth used for tag data links can be checked by clicking the **Detail** Button in the **Usage of Device Bandwidth** Area at the bottom of the Network Configuration Window.



Item	Description
#	The IP address of the device.
Comment	A description of the device. The comment is displayed below the device icon. The model number of the device is displayed by default.
Usage of Capacity (without Multicast Filter)	The usage rate of allowable tag data link bandwidth for the device is given. Bandwidth used/Allowable tag data link bandwidth The values outside parentheses are for when multicast filtering is used. The values inside parentheses are for when multicast filtering is not used.
Mbit/s (without Multicast Filter)	The network bandwidth used by the device for tag data link communications is given. The values outside parentheses are for when multicast filtering is used. The values inside parentheses are for when multicast filtering is not used.
Usage of IP multicast addresses	The number of multicast IP addresses actually used by the device for communications is given.
Total usage of IP multicast addresses	The number of multicast IP addresses used in the entire network is given. This value is used to estimate the number of multicast filters required for a switch.

Item	Description
Network Total of Max. Mbit/s	The total bandwidth used for tag data link communications in the entire network is given. Tag data links will not normally operate if the bandwidth allowed for the network is exceeded.

● Checking the Usage of Capacity and Network Bandwidth for Tag Data Links

The usage rate of allowable tag data link bandwidth for each built-in EtherNet/IP port is given in the **Usage of Capacity (without Multicst Filter)** column, and the network bandwidth usage for tag data link communications is given in the **Mbit/s (without Multicast Filter)** column.
The usage rate and the network bandwidth usage of tag data link communications for which multicast filtering is not supported by the Ethernet switch are given in parentheses in each corresponding column. These values include bandwidth usage for multicast packets since they are sent to all the nodes without connection settings.
These values can be adjusted as described in *14-2-4 Changing the RPI* on page 14-11.

● Checking the Total Number of Multicast IP Addresses in the Network

When using an Ethernet switch that supports multicast filtering, there must be sufficient multicast filters for the network. Based on the setting of connections, the Network Configurator indicates the number of multicast IP addresses to be used in the entire network.
Make sure that the number of multicast IP addresses to be used in the entire network does not exceed the number of multicast filters supported by the Ethernet switch. If necessary, replace the Ethernet switch with another one with sufficient multicast filters, or adjust the usage rate and network bandwidth usage with the values given for an Ethernet switch without multicast filtering (i.e., the values in parentheses). These values can be adjusted as described in *14-2-4 Changing the RPI* on page 14-11.

● Checking the Total Maximum Network Bandwidth

The Network Configurator displays the total maximum bandwidth to be used for the entire network. This value indicates the maximum possible bandwidth for a transmission path which connects Ethernet switches in cascade. If this value exceeds the bandwidth for each cascade connection in the actual network (e.g., 1,000 Mbps for an NX-series CPU Unit, or 100 Mbps for an NJ-series CPU Unit), the bandwidth for some transmission paths may be exceeded depending on the network wiring, and the tag data links may not operate normally.
If this occurs, calculate the bandwidth usage of each transmission path and make sure that the bandwidth for any cascade connection is not exceeded, or adjust the bandwidth to ensure that the value of **Network Total of Max. Mbit/s** does not exceed the bandwidth for any cascade connection. These values can be adjusted as described in *14-2-4 Changing the RPI* on page 14-11.

14-2-2 Tag Data Link Bandwidth Usage and RPI

The usage rate of allowable tag data link bandwidth as given in the **Usage of Capacity (without Multicast Filter)** column can be adjusted by changing the packet interval (RPI) setting.
If the RPI is set shorter, the **Usage of Capacity (without Multicast Filter)** will increase.
If the RPI is set longer, the **Usage of Capacity (without Multicast Filter)** will decrease.

The RPI can be set in one of the following ways.

- Setting the same PRI for all the connections
- Setting a PRI for connections of a particular device
- Setting a PRI for a particular connection

When the same RPI is set for all the connections, the **Usage of Capacity (without Multicast Filter)** will basically increase proportionally as the RPI is set shorter.
Example: If the **Usage of Capacity (without Multicast Filter)** is 40% with the PRI set to 50 ms for all the connections, the **Usage of Capacity (without Multicast Filter)** may increase to 80% when the RPI is changed to 25ms for all the connections.

Precautions for Correct Use

If the **Usage of Capacity (without Multicast Filter)** is between 80% and 100%, some operation with the Network Configurator which may cause load on the network, such as monitoring, or message communications with some user application may temporarily cause excessive load on the network and result in timeouts. If timeouts occur, increase one or all of the RPI values and reduce the usage of capacity.

14-2-3 Adjusting Device Bandwidth Usage

This section provides methods for adjusting the device bandwidth usage for tag data links.

Precautions for Correct Use

The Ethernet switch should be able to support the maximum network bandwidth for each CPU Unit. The maximum network bandwidth for each CPU Unit model is as follows.
- NX701 CPU Unit: 1,000 Mbit/s
- NX102 CPU Unit: 100 Mbit/s
- NX1P2 CPU Unit: 100 Mbit/s
- NJ-series CPU Unit: 100 Mbit/s

Ethernet Switches without Multicast Filtering

- Is the **Mbit/s (without Multicast Filter)** value for each node below the maximum network bandwidth?
  If any node exceeds the maximum network bandwidth, change the connection settings, such as the RPI.
- Is the value of **Usage of Capacity (without Multicast Filter)** for each node below 100%?
  If any node exceeds 100%, change the connections settings, such as the RPI.
- Is the value of **Network Total of Max. Mbit/s** below the maximum network bandwidth?
  If the value exceeds the maximum network bandwidth, the bandwidth for some transmission paths (e.g., an Ethernet switch or media converter) may be exceeded depending on the network wiring (e.g., cascade connection of Ethernet switches), and the tag data links may not operate normally. Check if the bandwidth of the transmission path in each cascade connection is not exceeded. If the bandwidth is exceeded, rewire the network or increase the bandwidth between Ethernet switches (e.g., increase to 1 Gbps). If these countermeasures are not possible, change the connection settings such as the RPI settings, and adjust the bandwidth to ensure that the value of **Network Total of Max. Mbit/s** does not exceed the bandwidth for any cascade connection..

Ethernet Switches with Multicast Filtering

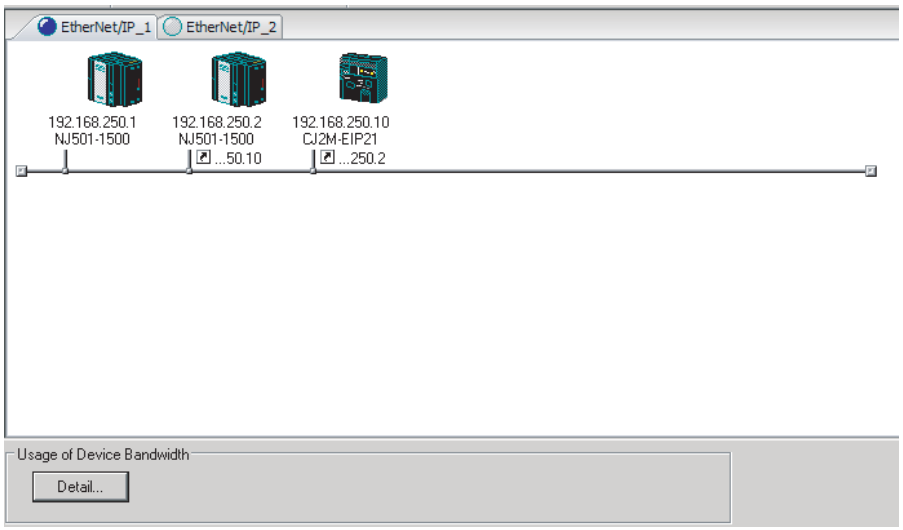
- Is the **Mbit/s** value for each node below the maximum network bandwidth?
  If any node exceeds the maximum network bandwidth, change the connection settings, such as the RPI.
- Is the **Usage of Capacity** value for each node below 100%?
  If any node exceeds the maximum network bandwidth, change the connection settings, such as the RPI.
- Is the **Network Total of Max. Mbit/s** value below the maximum network bandwidth?
  If the value exceeds the maximum network bandwidth, the bandwidth for some transmission paths (e.g., an Ethernet switch or media converter) may be exceeded due to the network wiring (e.g., cascade connection of Ethernet switches), and the tag data links may not operate normally. Check if the bandwidth of the transmission path in each cascade connection is not exceeded. If the bandwidth is exceeded, rewire the network or increase the bandwidth between Ethernet switches (e.g., to 1 Gbps). If these countermeasures are not possible, change the connection settings such as the RPI settings, and adjust the bandwidth to ensure that the value of **Network Total of Max. Mbit/s** does not exceed the bandwidth for any cascade connection.
- Is the **Mbit/s (without Multicast Filter)** value for each node below the maximum network bandwidth? Or, is the value of **Usage of Capacity (without Multicast Filter)** for each node below 100%?
  If any node exceeds either of them, check whether the multicast filtering on the relevant Ethernet switch is functioning correctly. If the number of multicast filters on the Ethernet switch is less than the number of **Total usage of IP multicast addresses**, the bandwidth for some transmission paths may be exceeded depending on the network wiring (e.g., cascade connection of Ethernet switches), and the tag data links may not operate normally. Calculate the number of multicast filters required for each Ethernet switch on the network, and check if the resulting number is below the number of multicast filters provided by the Ethernet switch. If the Ethernet switch does not have a sufficient number of multicast filters, replace it with another one which has sufficient multicast filters, or change the connection settings, such as the RPI settings.

14-2-4 Changing the RPI

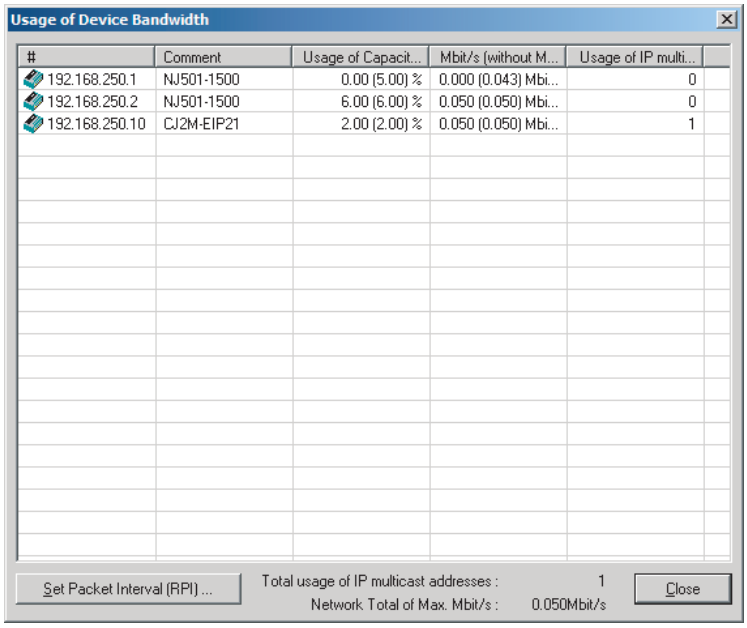
You can check **Usage of Capacity (without Multicast Filter)** values offline for the usage rate of allowable tag data link bandwidth if you follow the procedure provided in *14-2-1 Checking Bandwidth Usage for Tag Data Links* on page 14-8.
You can adjust **Usage of Capacity (without Multicast Filter)** values by changing packet interval (RPI) values.
If required communications performance cannot be achieved after the adjustment, re-evaluate the network configuration.

1 Make required settings in the Network Configuration Window on the Network Configurator.

2 Click the **Detail** Button in the **Usage of Device Bandwidth** Area at the bottom of the Network Configuration Window.



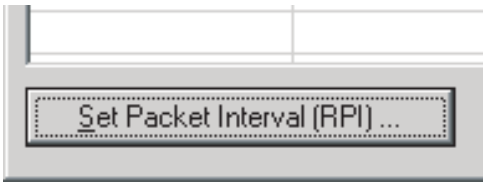
The **Usage of Device Bandwidth** Dialog Box is displayed.



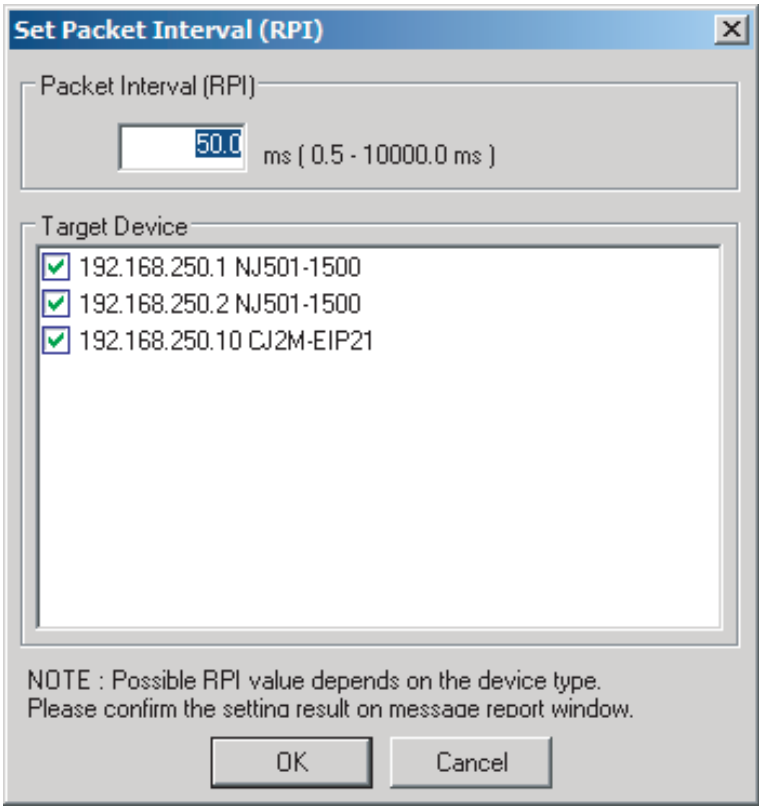
The **Usage of Capacity (without Multicast Filter)** column shows the usage rate of allowable tag data link bandwidth, and the **Mbit/s (without Multicast Filter)** column shows the network bandwidth usage.

3 You can adjust the **Usage of Capacity (without Multicast Filter)** value by changing the packet interval (RPI) for the relevant device.
There are three methods for changing the RPI as shown below.

- Method 1: Set the Same RPI for All the Connections
  You can adjust the **Usage of Capacity (without Multicast Filter)** value by changing the packet interval (RPI) values for all the connections at the same time.
  1) Click the **Set Packet Interval (RPI)** Button in the **Usage of Device Bandwidth** Dialog Box.



2) The **Set Packet Interval (RPI)** Dialog Box is displayed. Input a new RPI value, and click the **OK** Button.

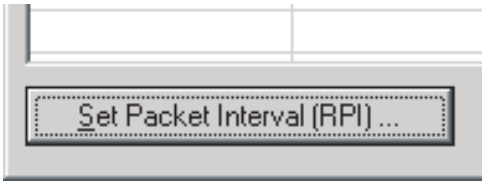


- Method 2: Change the RPI for a Specific Device

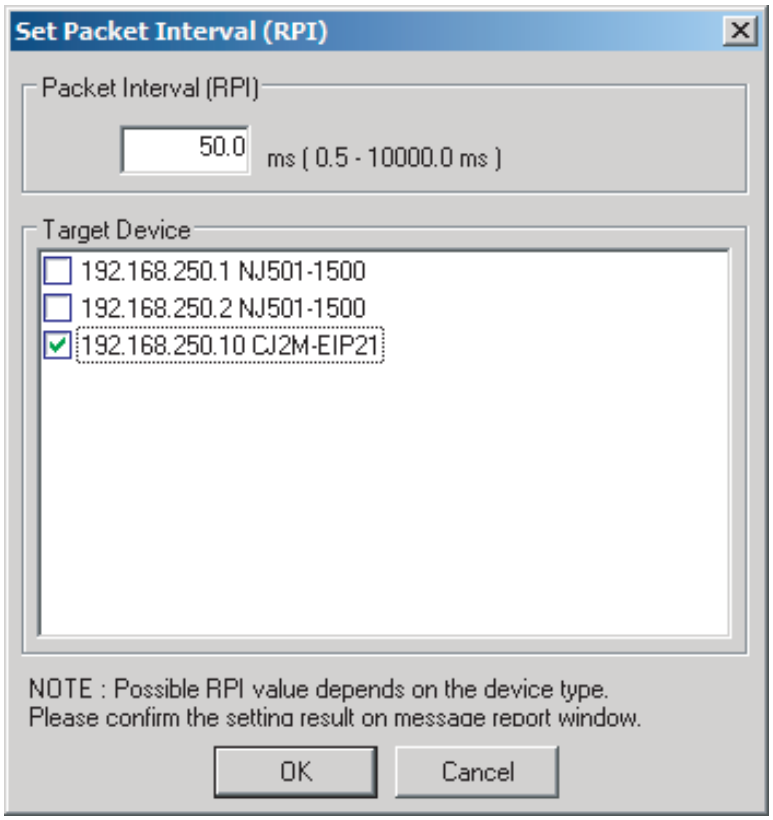
You can adjust the **Usage of Capacity (without Multicast Filter)** value by changing the RPI for all the connections of a specific device.

Note that the **Usage of Capacity (without Multicast Filter)** values for the target devices of the connections are also changed.

1) Click the **Set Packet Interval (RPI)** Button in the **Usage of Device Bandwidth** Dialog Box.



2) The **Set Packet Interval (RPI)** Dialog Box is displayed. In the **Target Device** Area, clear the check boxes for devices to which this RPI setting change is not applied.



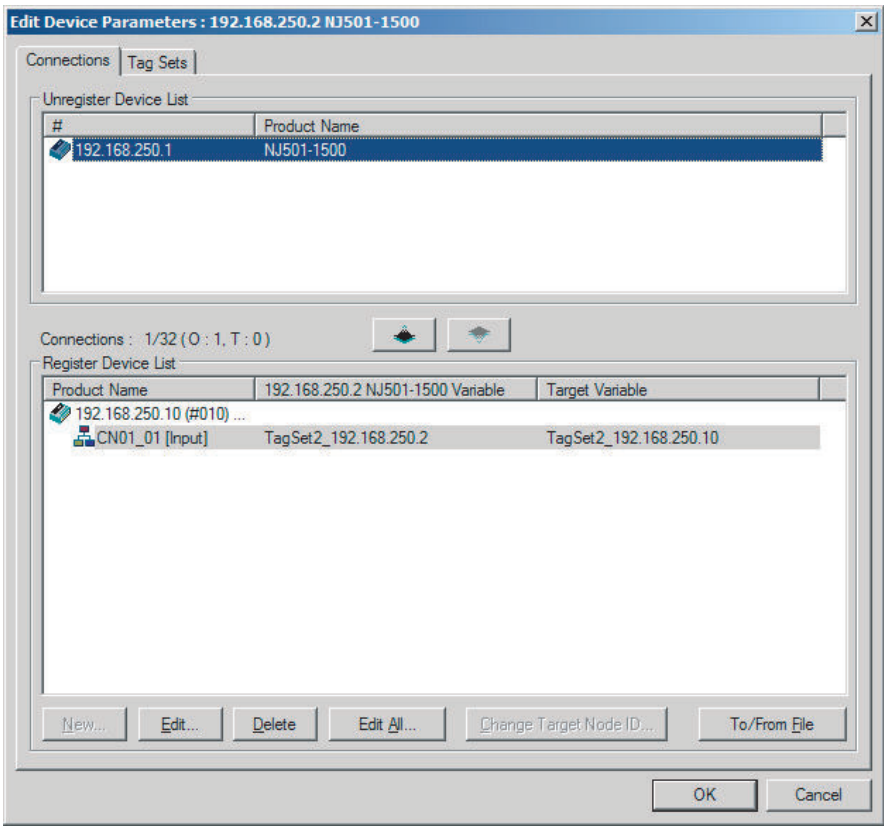
3) Input a new RPI value, and click the **OK** Button.

- Method 3: Change the RPI for a Specific Connection

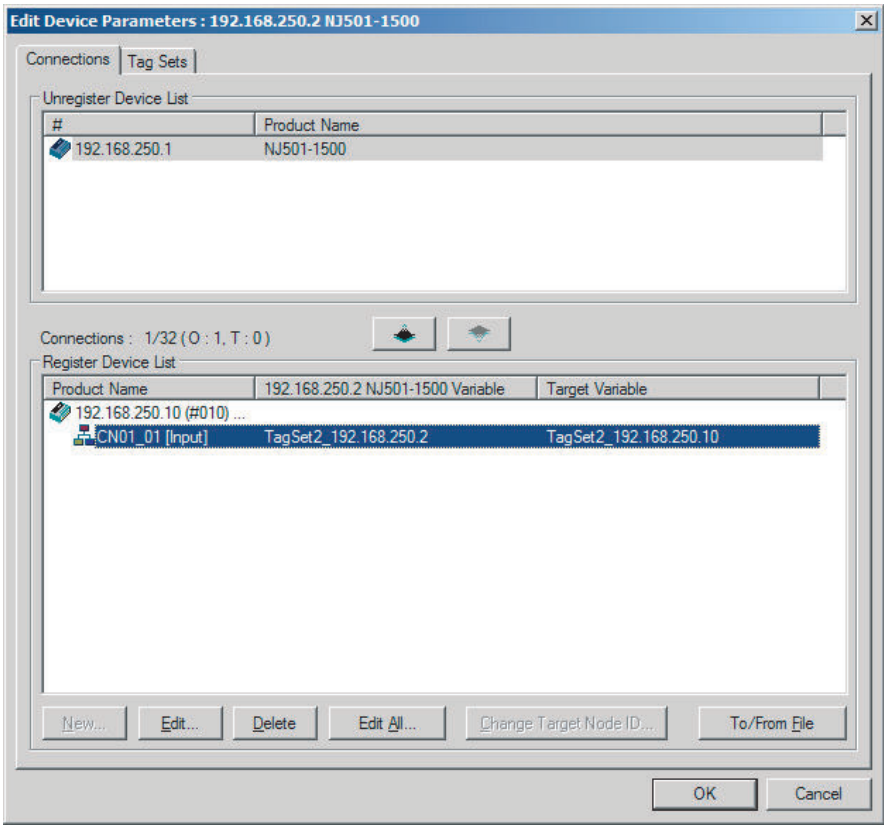
You can adjust the **Usage of Capacity (without Multicast Filter)** value by changing the RPI for a specific connection.

Note that the **Usage of Capacity (without Multicast Filter)** value for the target device of the connection are also changed.

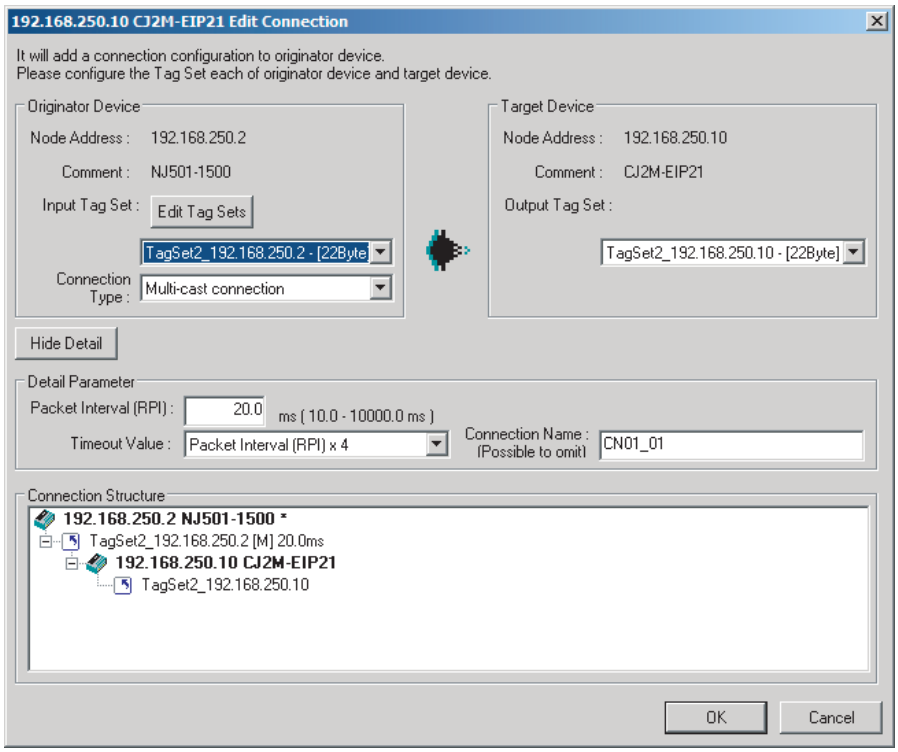
1) Click the **Close** Button in the **Usage of Device Bandwidth** Dialog Box.
2) Double-click the device that is set as the originator of the connection. The **Edit Device Parameters** Dialog Box is displayed.



3) In the **Register Device List** Area, select the connection for which you want to change the RPI, and click the **Edit** Button.



4) The Edit Connection Dialog Box for the device is displayed. Input a new packet interval (RPI) value, and click the **OK** Button.



4 If the bandwidth usage rate is not set as desired even after the above operation, re-evaluate the network configuration,considering the following points. (Refer to *14-2-3 Adjusting Device Bandwidth Usage* on page 14-10.)

- Reduce the number of nodes and connections
- Split the network

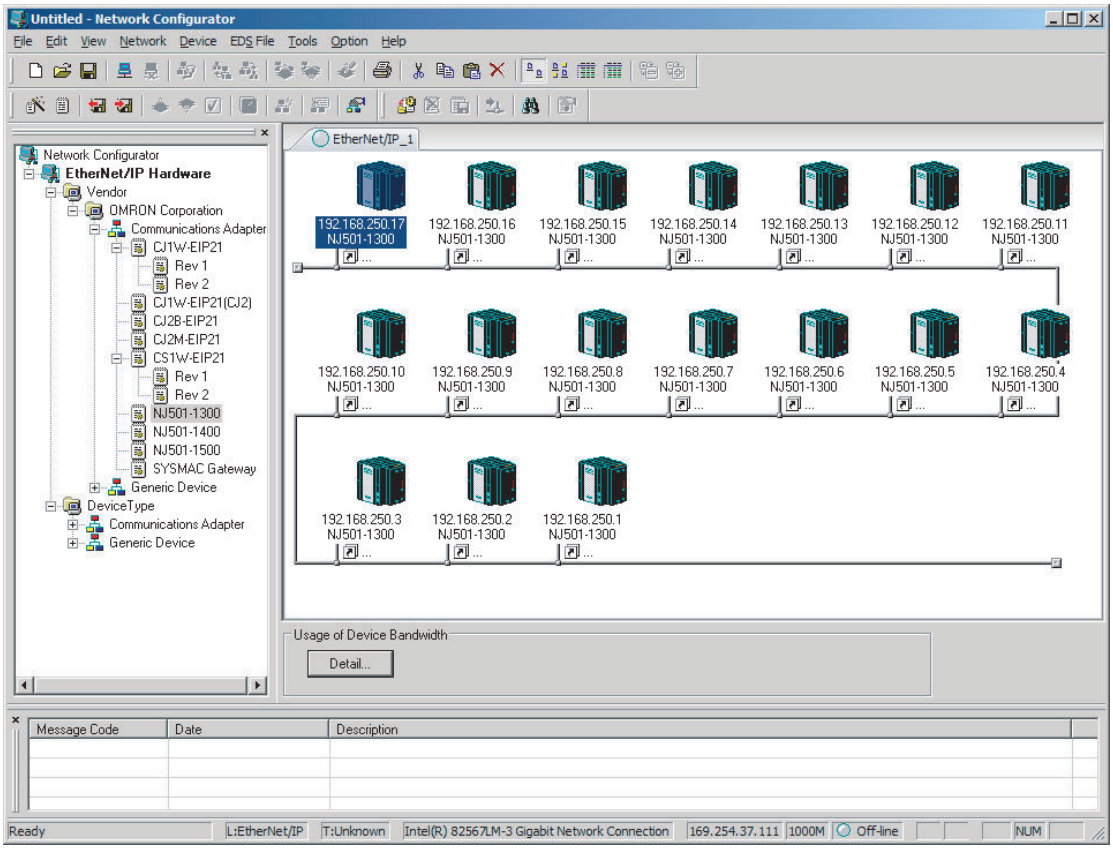
5 Check the bandwidth usage rate again.
After you change the connection settings, click the **Detail** Button in the **Usage of Device Bandwidth** Area at the bottom of the Network Configuration Window to check the bandwidth usage as described in *14-2-1 Checking Bandwidth Usage for Tag Data Links* on page 14-8.
It is important to check the bandwidth usage particularly after you change the RPI values for individual connections, instead of setting the same RPI for all the connections.

6 Run user tests to verify that there are no problems with the new values.

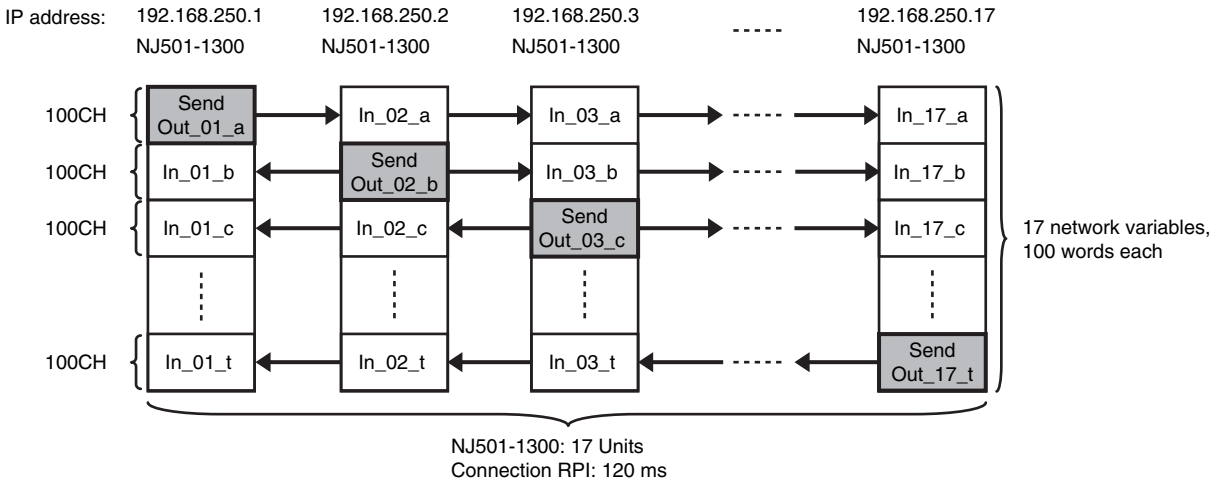
14-2-5 RPI Setting Examples

The following examples explain how to calculate the packet intervals (RPIs) in the following network configuration.

Conditions

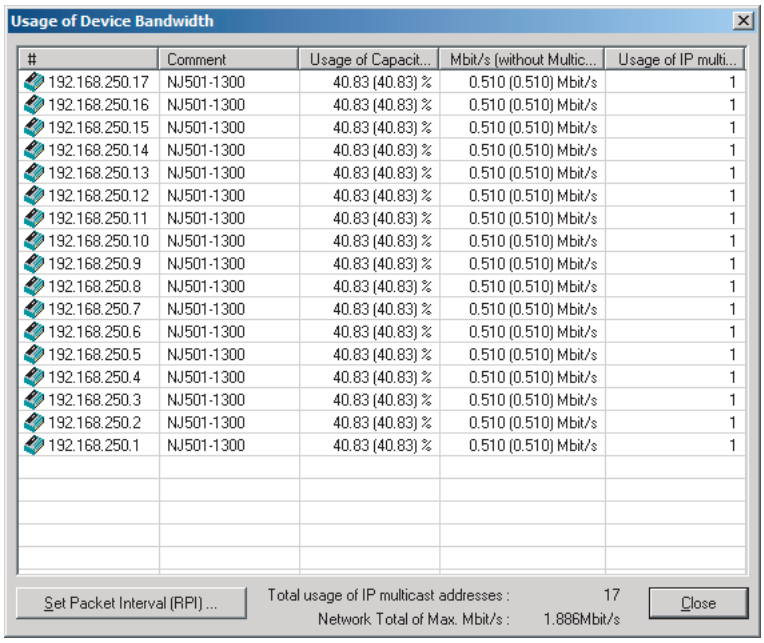


- Connections:
  Example) Seventeen NJ501-1300 Units are connected to the network.
  Each device has one 100-word tag for sending and sixteen 100-word tags for receiving, and exchanges data with each other.
  The packet interval (RPI) for all the connections is set to 120 ms.
  The IP addresses of the devices range from 192.168.250.1 to 192.168.250.17.



Checking the Device Bandwidth Usage

When you click the **Detail** Button in the Usage of Device Bandwidth Area, the window shows that the usage rate of the tag data link bandwidth for each device is 40.83%, as given in the Usage of Capacity column in the following window.



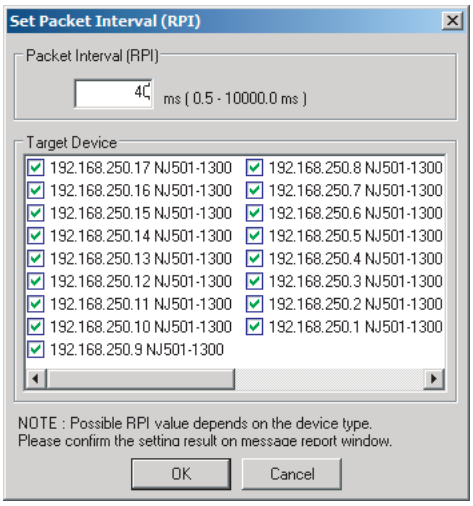
Changing Settings

Method 1: Setting the Same RPI for All the Connections

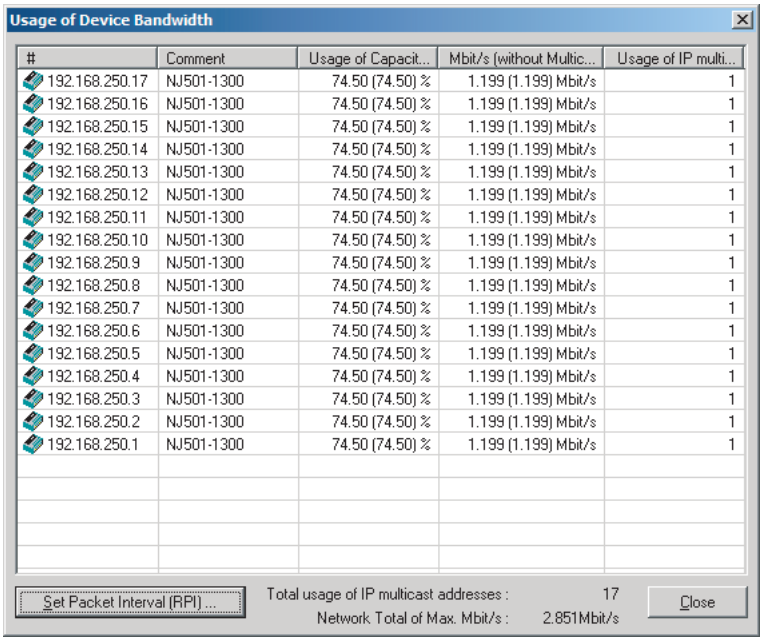
In the previous example, the usage rate of allowable tag data link bandwidth is 40.83% for all the devices as given in the Usage of Capacity column, and the RPI is set to 120 ms for all the connections. In the next example, change the RPI to 40 ms so as to increase the usage rate of allowable tag data link bandwidth up to 80% or less.

Click the **Set Packet Interval (RPI)** Button in the **Usage of Device Bandwidth** Dialog Box to display the **Set Packet Interval (RPI)** Dialog Box.

Input 40 ms as the new RPI value, and click the **OK** Button.



If you set the same packet interval (RPI) for all the connections, the table shows that the usage rate of allowable tag data link bandwidth is 74.50% for all the device as shown in the Usage of Capacity column, and this indicates that the shortest packet interval is 40 ms.

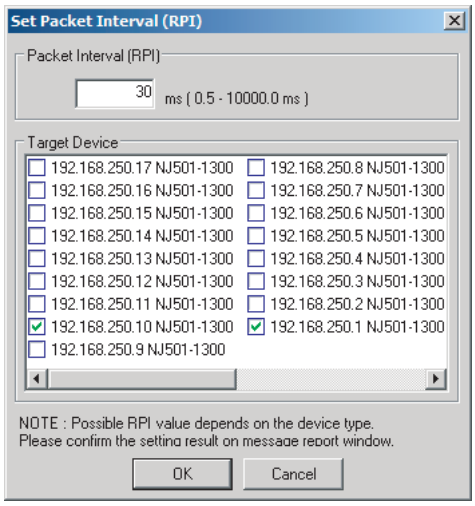


Method 2: Changing the Packet Intervals (RPIs) of Specific Devices

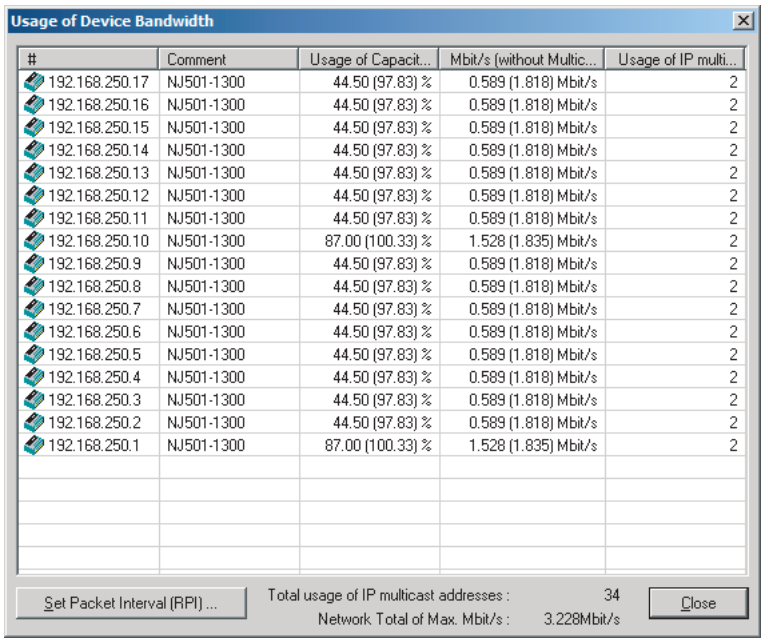
In this example, set faster tag data links for specific two devices: 192.168.250.1 and 192.168.250.10.

Click the **Set Packet Interval (RPI)** Button in the **Usage of Device Bandwidth** Dialog Box to display the **Set Packet Interval (RPI)** Dialog Box.

In the **Target Device** Area, clear the check boxes for devices to which this RPI change is not applied (all the devices except 192.168.250.1 and 192.168.250.10). Input 30 ms as the new RPI value, and click the **OK** Button.



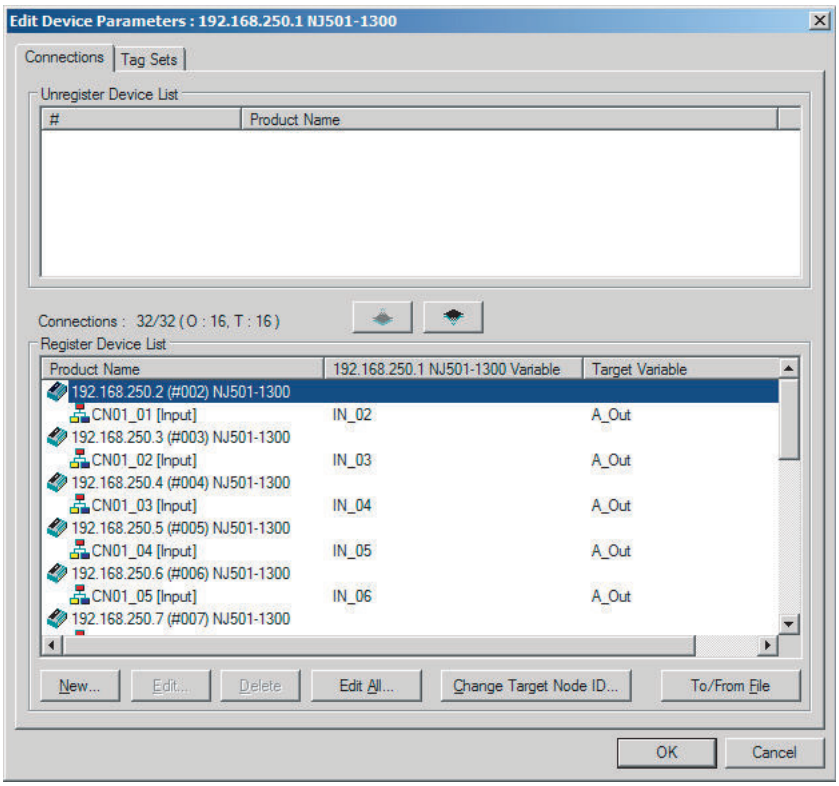
The usage rate of allowable tag data link bandwidth for each of the two devices, 192.168.250.1 and 192.168.250.10, increases to 87.00% as shown in the Usage of Capacity column, and this indicates that the shorter RPI is set for the connections of these devices.
Note that the usage rate of allowable tag data link bandwidth for all the other devices is also increased from 40.83% to 44.50% since they are connected with the two devices, 192.168.250.1 and 192.168.250.10.



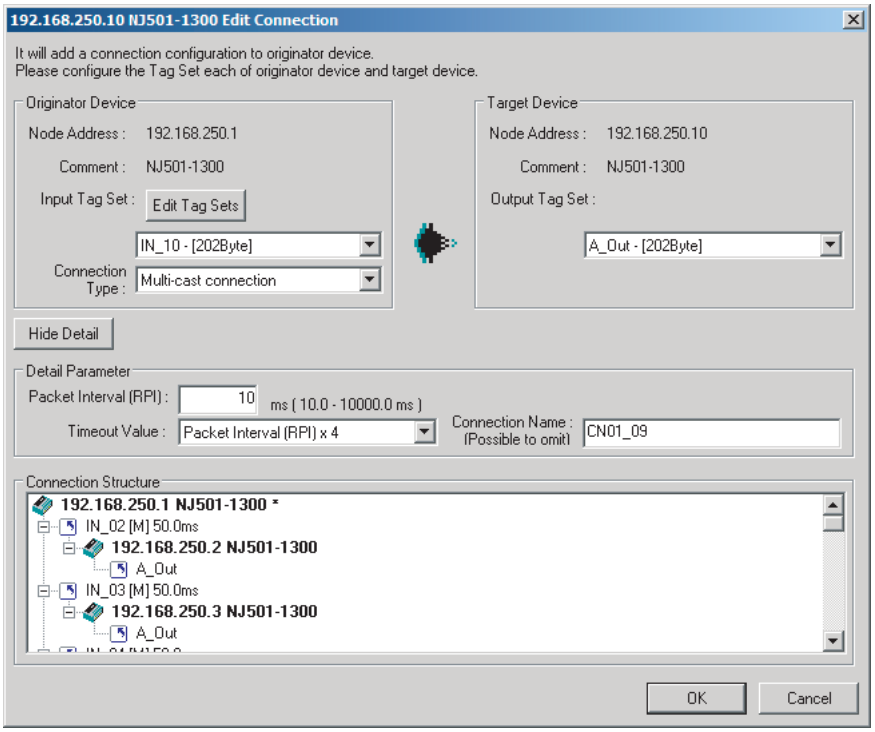
In this case, if the Ethernet switch has no multicast filter, the Usage of Capacity value would be 100.33% and communications errors might occur due to traffic overload at the built-in EtherNet/IP port.

Method 3: Changing the Packet Interval (RPI) of a Specific Connection

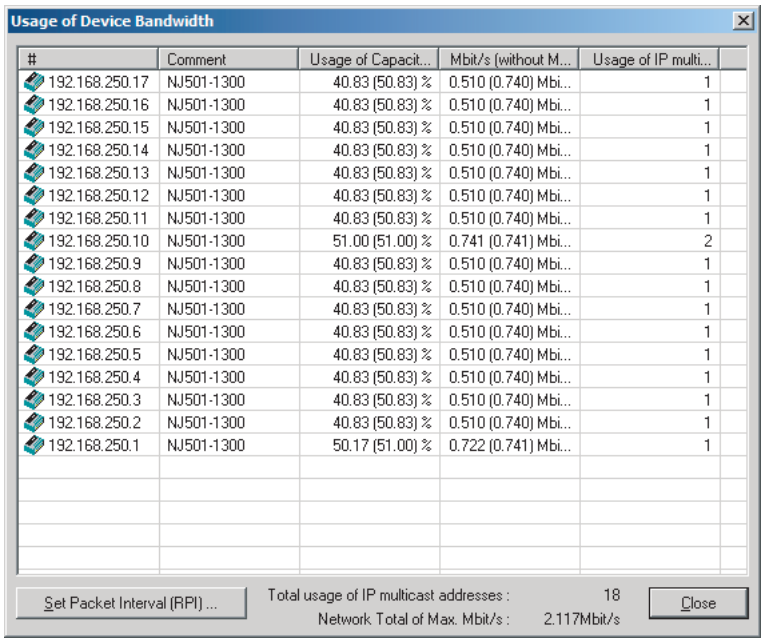
In this example, set faster tag data links for a specific connection of a device, 192.168.250.1. Double-click the device, 192.168.250.1, in the Network Configuration Window.



Since the Register Device List shows a list of devices connected with 192.168.250.1, double-click a device, 192.168.250.10, in the list.



Input 10 ms as the new RPI value in the Edit Connection Dialog Box, and click the **OK** Button. The usage rate of allowable tag data link bandwidth for the device 192.168.250.1 increases to 50.17% as shown in the Usage of Capacity column, and this indicates that the RPI for the specific connection is set shorter.



Note that the usage rate of allowable tag data link bandwidth for the device, 192.168.250.10, is also increased from 40.83% to 51.00%.

14-3 I/O Response Time in Tag Data Links

Additional Information

This section describes built-in EtherNet/IP ports on the NX and NJ-series CPU Units. Compared to those built-in EtherNet/IP ports, EtherNet/IP Units, and built-in EtherNet/IP ports on CJ2H CPU Units (CJ2H-CPU6□-EIP) and CJ2M CPU Units (CJ2M-CPU3□) support different data processing performance. For details, refer to *6-4 Tag Data Links with Other Models* on page 6-87.

As explained in *6-1-7 Concurrency of Tag Data Link Data* on page 6-12, the tag (network variable) with a refreshing task is refreshed when the refreshing task is executed in the user program. By setting the refreshing task, you can calculate the I/O response time that is not affected by the system service.

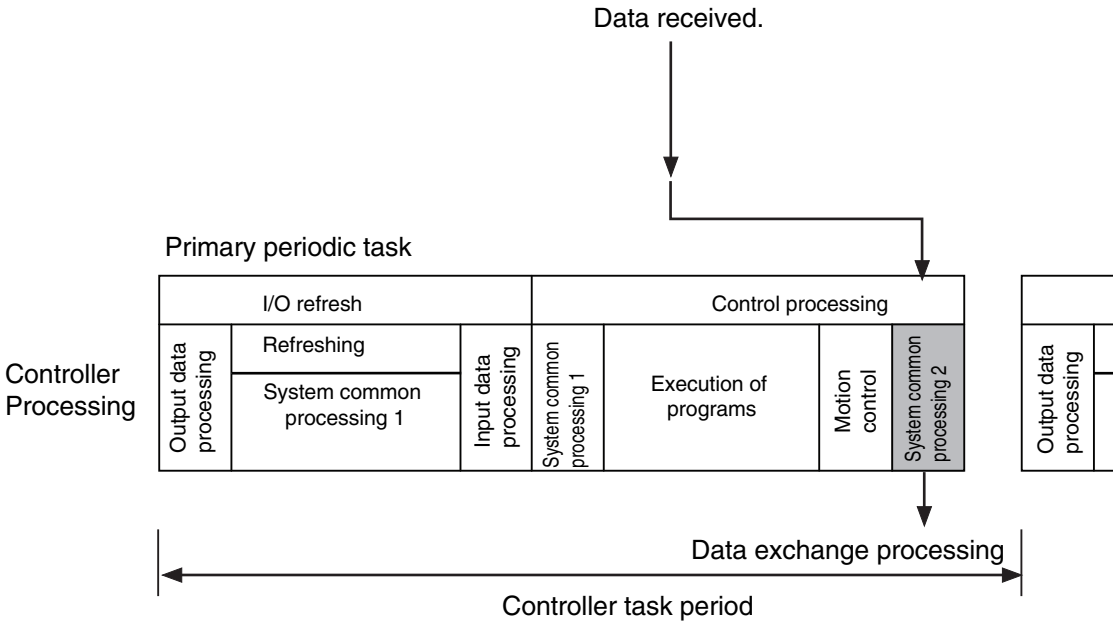
Precautions for Correct Use

The refreshing task must be set to all tags (network variables). If both tags (network variable) with a refreshing task and without it exist in a configuration, system service may affect the operation and I/O response time described in this section may not be maintained.

This section describes the I/O response time when refreshing tasks are set properly.

14-3-1 Timing of Data Transmissions

The following figure shows the timing of transmitting data for tag data link between a built-in EtherNet/IP port and a CPU Unit.
Data is transmitted at the timing of executing the system common processing 2 of the refreshing task.



You can specify either of the following task types for a refreshing task.

- Primary periodic task
  The primary periodic task has the highest execution priority. It is executed with high speed and high precision.

- Periodic task
  A periodic task is executed during the interval between executions of a primary periodic task.

You do not need to specify a refreshing task for tags (variables) with AT specifications; the tag data is transmitted in a primary periodic task. (This applies to NX102, NX1P2 and NJ-series CPU Units)
Specify a task type for each tag for tag data link processing.
On the Sysmac Studio, set a refreshing task for each variable assigned as a tag.
Refer to the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for details on setting refreshing tasks.

14-3-2 Built-in EtherNet/IP Port Data Processing Time

This section describes the data processing time required to transfer data between the built-in EtherNet/IP port and the CPU Unit.

Data Processing Time Overview

The time required for data processing consists of the following three elements.

1. **Variable Access Time**
   Calculate the time required to transfer tag data, which is regarded as the time required to access the variable.
   This calculation is performed for each task. Therefore, if multiple tag sets are set for the same refreshing task, use the total for all tag values in the tag sets.
   Use the following equation for calculating the variable access time.
   Variable access time [μs] = total size of variables [bytes] × a + number of variables × b + number of accesses × c + d
   Number of accesses: equal to the number of tag sets
   a to d: Constant values as given below

CPU Unit model	Constant value (μs)			
	a	b	c	d
NX701-□□□□	0.0005	0.033	2.67	7.22
NX102-□□□□	0.0040	0.240	3.27	25.21
NX1P2-□□□□	0.0040	0.240	3.27	25.21
NJ501-□□□□	0.0010	0.490*1	1.41	6.68
NJ301-□□□□	0.0015*2	0.560*3	2.15	7.52
NJ101-□□□□	0.0015	0.560	3.83	7.52

*1. The value is 0.58 for CPU Units with unit version 1.02 or earlier.
*2. The value is 0.0009 for CPU Units with unit version 1.02 or earlier.
*3. The value is 1.03 for CPU Units with unit version 1.02 or earlier.

2. **Number of Data Transfers**
   Tag data transfer is executed as part of the task processing.
   If the time required to process the data transfer is greater than the variable access time (*2), the entire data cannot be sent in one task period and needs to be split and sent over multiple times instead.

Number of data transfers = Time required to send the entire data (*1) / Variable access time (*2) set for the task

*1. This is the variable access time as calculated in step 1 above.
*2. The variable access time is the maximum processing time for accessing the variable. Double-click **Task Settings** under **Configurations and Setup** on the Sysmac Studio to display the **Task Settings** Tab Page, and configure the settings for each task.

Precautions for Correct Use

The maximum number of tag data link words that can be transferred through a built-in EtherNet/IP port is 184,832 words on an NX701 CPU Unit (total of 369,664 words with the two ports), 9,600 words on an NX102 CPU Unit (total of 19,200 words with the two ports), and 9,600 words on an NX1P2 CPU Unit or NJ-series CPU Unit.
If the number of tag data link words exceeds the number of words that can be exchanged with the CPU Unit at one time, the data is divided and transferred over multiple times

3. **Actual Time Required for Data Transfer**
You can use the task period of the refreshing task and the number of data transfers as calculated in (2) above to calculate the actual time required to transfer the data.

Task period × Number of data transfers

Data Processing Time Calculation Example

The following shows an example to explain how to calculate the time required for tag data transfer.

- CPU Unit model
  NJ501-□□□□
- Refreshing task
  Primary periodic task
  Task period: 500μs (variable access time: 3%)
- Settings of tag sets

Tag set	Refreshing task	Number of variables	Total size of variables
Tag set A	Primary periodic task	8	600 bytes
Tag set B	Primary periodic task	4	200 bytes
Tag set C	Primary periodic task	10	1,000 bytes

1 Calculate the variable access time as shown below.
[(600 + 200 + 1,000) bytes × 0.001 μs] + [(8 + 4 + 10) variables × 0.49 μs] + 3 × 1.41 μs + 6.68 μs = 23.49 μs

2 Calculate the number of data transfers.

Time required for data transfer:	Variable access time in step1 = 23.49 μs
Variable access time set for the task:	500 μs × 0.03 = 15 μs
Number of data transfers	23.49 μs ÷ 15 μs = 1.6 times

Thus, approximately two data transfers are required.

3 Calculate the actual time required for the entire data transfer.

500 μs × 2 times = 1,000 μs

14-3-3 Relationship between Task Periods and Packet Intervals (RPIs)

Effect of Tag Data Links on Task Periods

Tag data transfer is executed as part of the task processing.
Therefore, the tag data transfer process is added to the task processing for tasks set as a tag's refreshing task. This requires you to make adjustments to the variable access time and task period in the Task Settings Tab Page so that these processes are completed within a single task period.

1 Calculate the time required for the data transfer and set the result as the variable access time(*).
For the formula for calculating the time required for data transfer, refer to *Data Processing Time Overview* on page 14-24.
* If the same refreshing task is set for multiple tag sets, calculate the total time required for all tags in tag sets.

2 Set the variable access time in the Task Setup to a value equal to or greater than the value calculated in step 1 above.
Adjust the task period time after adding in the time calculated in step 1. Use the Sysmac Studio to set the variable access time and task period settings.
Refer to *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for details.

Adjusting Packet Intervals (RPIs) According to the Task Period

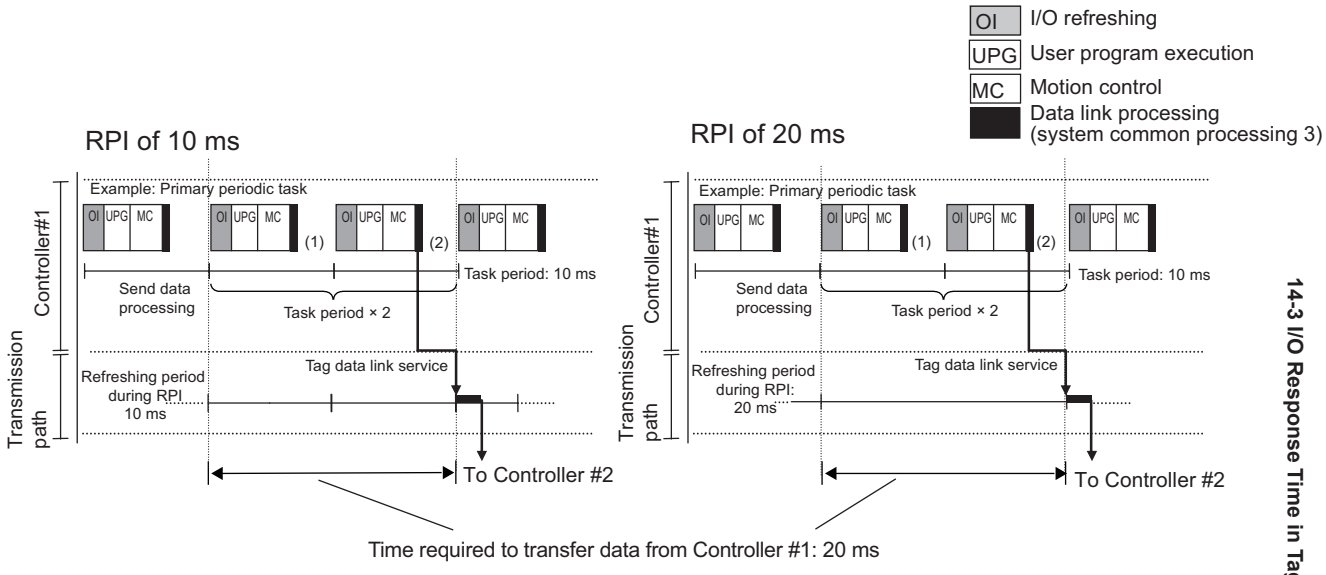
Tag data is transferred based on the actual time required for the transfer (task period × number of data transfers), regardless of the packet interval (RPI) setting.
Set the packet interval (RPI) as below.

Actual time required for data transfer (Task period × Number of data transfers) < RPI

For details on the actual time required for data transfer, refer to *14-3-2 Built-in EtherNet/IP Port Data Processing Time* on page 14-24.

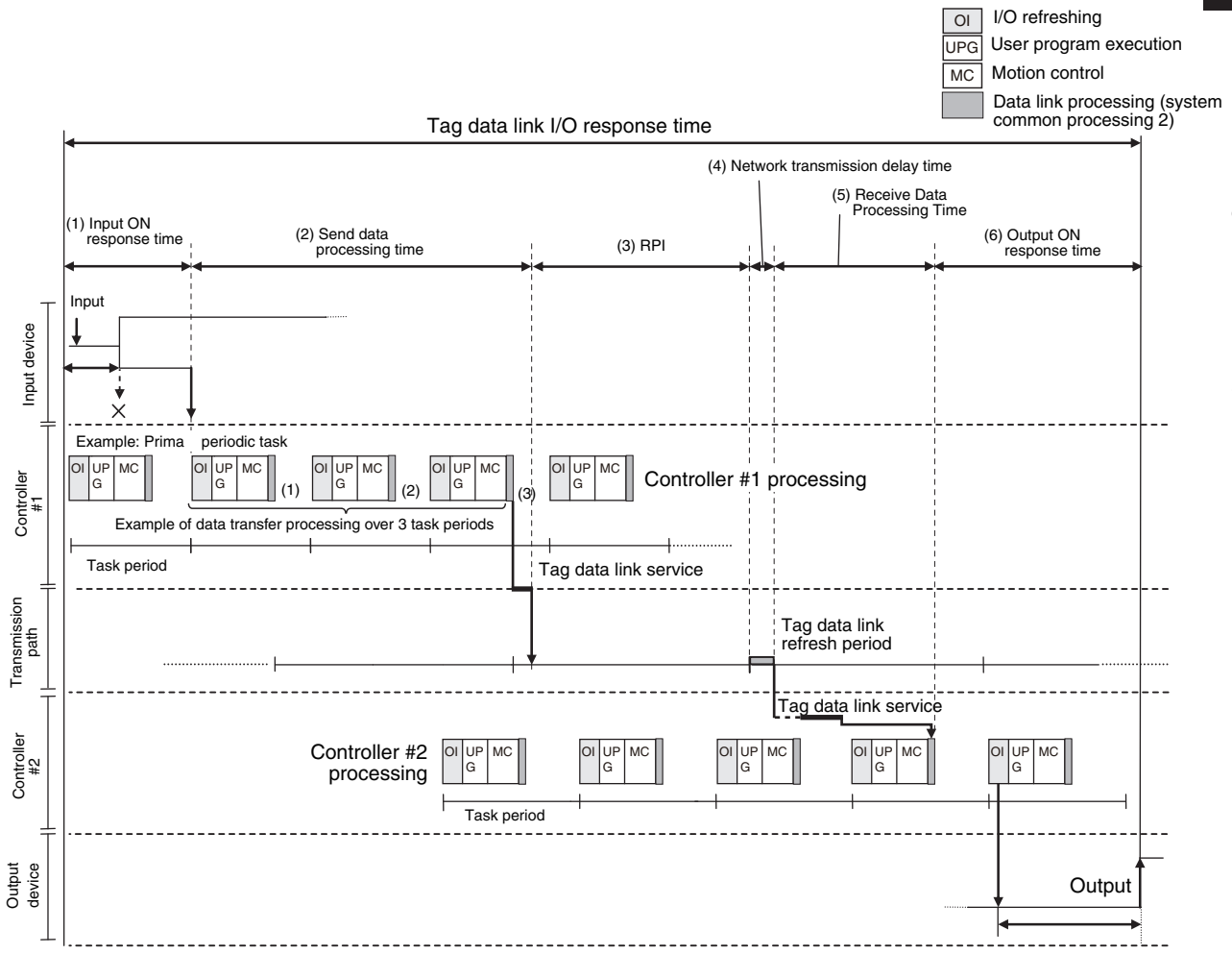
Example: Relationship between the RPI Setting and the Time Required for Data Transfer
- Task period: 10 ms
- Number of data transfers: 2 times
- Actual time required for data transfer: 10 ms × 2 times = 20 ms

Regardless of the RPI value, the time required for the data transfer is 20 ms.



14-3-4 Maximum Tag Data Link I/O Response Time

You can calculate the maximum I/O response time by adding up the time of (1) to (6) in the following figure.



Additional Information

- For CPU Units with unit version 1.03 or later, tag data link is processed in the tag data link service.
- For CPU Units with unit version 1.00 to 1.02, tag data link is processed in the system service. If a tag data link timeout occurs, reconsider the execution time of the system service.

1. **Input ON Response Time**
   The input ON response time contains the delay time for the external input device from when the input occurs until the switch actually changes to ON and the time until the input data is stored in the memory area of the CPU Unit. Refer to the input delay information of the device for input switch delay time.
   One task period is required until the input data is stored in the memory area of the CPU Unit. Accordingly, the input ON response time is calculated as below.

   Input ON response time = Input device delay time + Task period

2. **Send Data Processing Time**
   This is the time required to transfer a variable from a CPU Unit to the built-in EtherNet/IP port. Since data transfer is executed as part of task processing, the send data processing time is as long as the task period.
   If the data is larger than the allowable data size to send in a single task process (which can be set with **Variable Access Time** of the task), the data will be transferred over more than one task period, requiring additional time equivalent to the task period multiplied by the number of transfers.
   For details on the send data processing time, refer to *14-3-2 Built-in EtherNet/IP Port Data Processing Time* on page 14-24.

3. **Packet Interval (RPI)**
   This is the communications refresh period which can be specified on the Network Configurator.

4. **Network Transmission Delay Time**
   The transmission delay on an Ethernet line is 50 μs or less. This delay time can be ignored.

5. **Receive Data Processing Time**
   This is the time required to transfer data that is received on the built-in EtherNet/IP port to a variable in the CPU Unit.
   Since data receive is executed as part of task processing, the receive data processing time is as long as the task period.
   If the data is larger than the allowable data size to receive in a single task process (which can be set with **Variable Access Time** of the task), the data will be transferred over more than one task period, requiring additional time equivalent to the task period multiplied by the number of transfers.
   For details on the receive data processing time, refer to *14-3-2 Built-in EtherNet/IP Port Data Processing Time* on page 14-24.
   Data transfer is executed once every task period. If another input data is received just after the data transfer in the current task period, the transfer of the received data will be delayed by one Controller task period.

Additional Information

If the Unit has connections with multiple nodes, the total amount of data to be exchanged will increase, and the Unit may send or receive data larger than the data size allowed per transfer. In this case, the number of data transfers increases.

6. **Output ON Response Time**

This is the delay time from when an output command is issued by the Controller until the output is executed on the external output device.

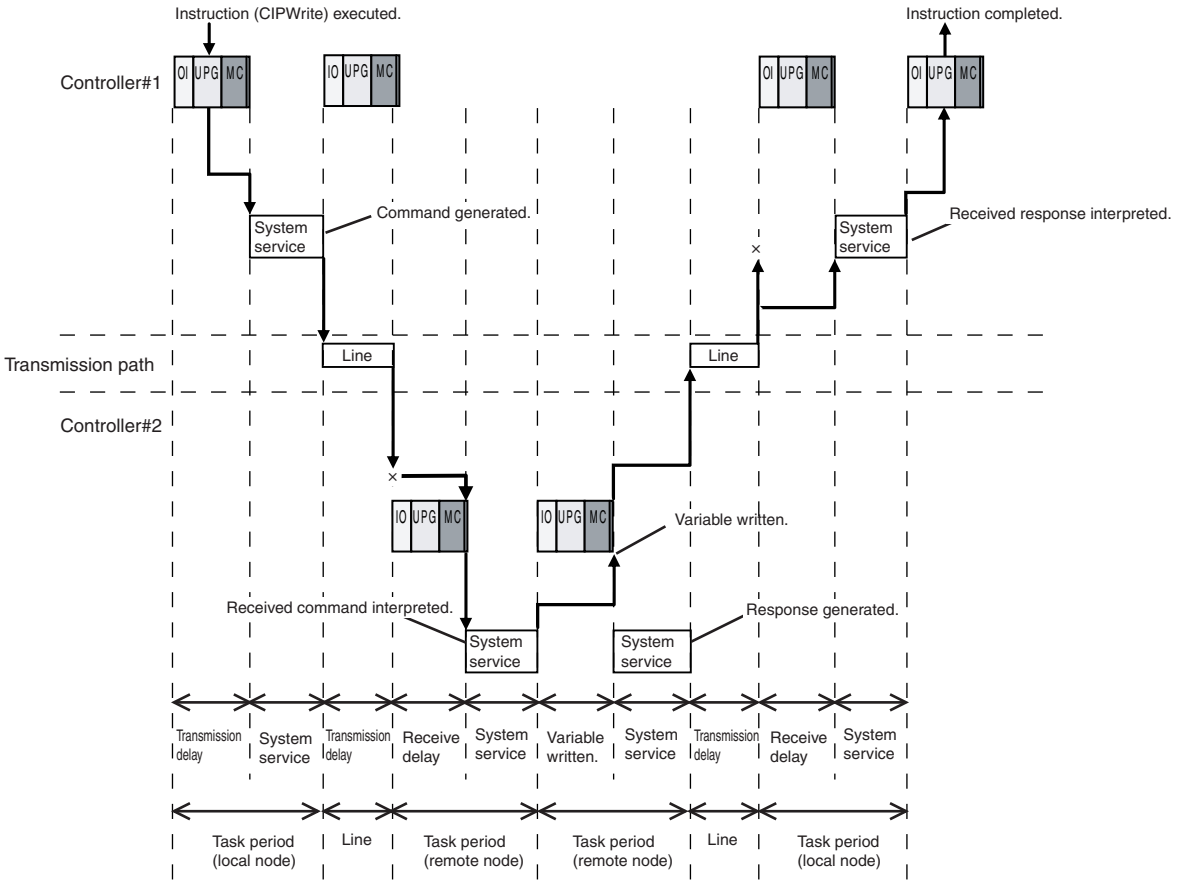
Output ON response time = Output device delay time + CPU task period

Additional Information

The I/O response time may be longer due to noise, or other causes.

14-4 Message Service Transmission Delay

This section describes delay time in the service processing of a CIP communications instruction (CIP-Write).



Maximum transmission delay time = Send delay + System service execution time + Transmission delay + Receive delay + System service execution time

(Local node task period) (Remote node task period)

+ Variable write time + System service execution time + Transmission delay + Receive delay + System service execution time

(Remote node task period) (Local node task period)

Processes with delay time are processed within the task period of each node as shown in the above diagram.
Delay time related to transmission lines is as below.

● Transmission Delay

The transmission delay on an Ethernet line is 50 μs or less. This delay time can be ignored.

Additional Information

Depending on the actual operating environment, the transmission time may be longer than the one calculated with the equations given above.
The following factors can cause longer transmission time: the load rate of the network (the degree of network congestion), the window size of each network node, traffic load on the built-in EtherNet/IP port (e.g., simultaneous tag data link communications), and the system configuration.

CIP communications instructions are executed in the system service process.
If a timeout occurs for a CIP communications instruction, reconsider the execution time for the system service.

15

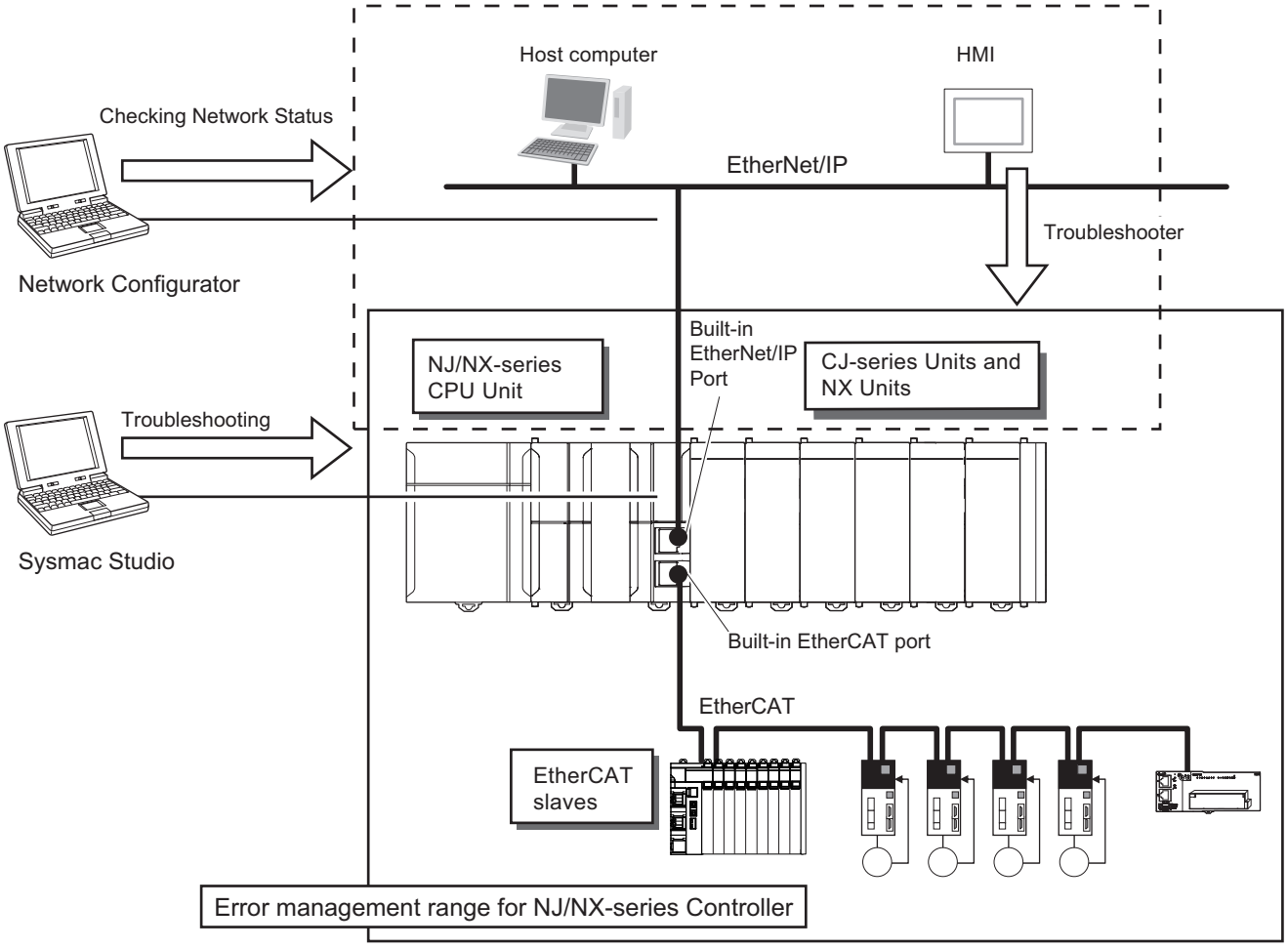
Troubleshooting

This section explains how to detect errors, how to check the communication status of the EtherNet/IP network with the Network Configurator, and how to identify and troubleshoot errors which may occur due to the tag data link connection status.

15-1 Overview of Troubleshooting

You manage all of the errors that occur on the NJ/NX-series Controller as events.
This allows you to see what errors have occurred and find corrections for them with the same methods for the entire range of errors that is managed (i.e., CPU Unit, NX Units, NX-series Slave Terminals, EtherCAT slaves,*1 and CJ-series Units).

*1. Only Sysmac devices are supported.



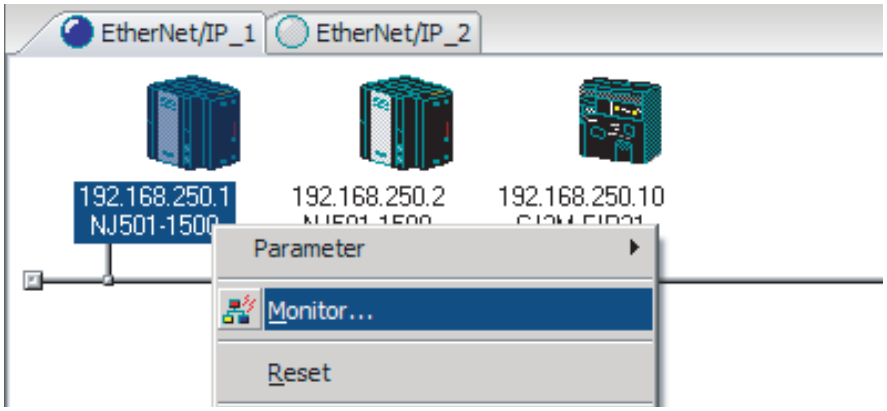
You can use the troubleshooting functions of the Sysmac Studio or the Troubleshooter on an HMI to quickly check for errors that have occurred and find corrections for them.

Refer to the *NJ/NX-series Troubleshooting Manual (Cat. No. W503)* for error types and details, specific corrections when errors occur, and troubleshooting information on the entire NJ/NX-series Controllers.

15-2 Checking Status with the Network Configurator

15-2-1 The Network Configurator's Device Monitor Function

Connect the Network Configurator online, select the device to be checked, right-click to display the pop-up menu, and select **Monitor**.



The **Monitor Device** Dialog Box will be displayed.

Precautions for Correct Use

Monitoring may not be performed if the following settings are configured on the NJ/NX-series Controller on the connection route or on the destination NJ/NX-series Controller. If monitoring is not performed, check the following settings. Refer to *CIP Message Server* on page 4-19, and*Packet Filter* on page 4-7 for details on the settings.

- The **Do not use** Option is selected for the CIP message server.
- The **Use** Option is selected for Packet Filter.

Additional Information

If a communications error occurs during monitoring, the dialog box will continue to show the last information that was collected.
To start monitoring again, close the **Monitor Device** Dialog Box, and then open the **Monitor Device** Dialog Box again.

You cannot monitor the CIP Safety communications status with Network Configurator. Refer to the *NX-series Safety Control Unit User's Manual (Cat. No. Z930)* for details on confirming CIP Safety communications status.

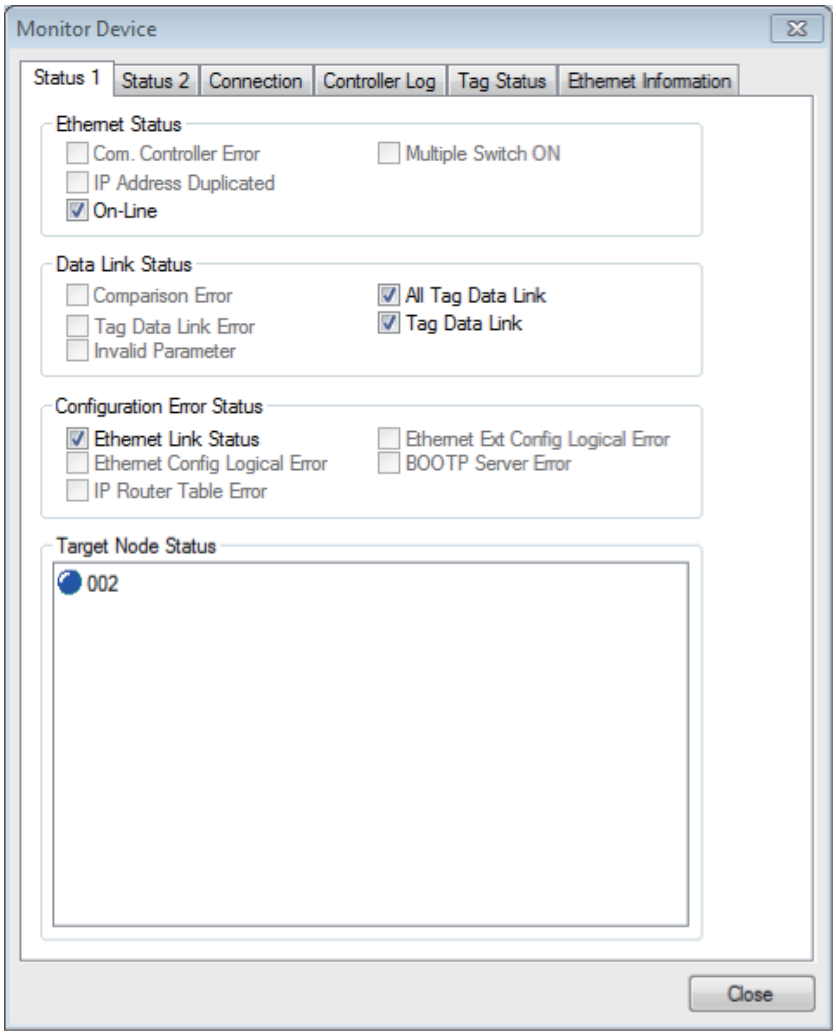
● Status 1 Tab Page

The following check boxes are displayed for the status. If a check box is checked with ☑, the status is TRUE.

Classification	Item	TRUE status description
Ethernet Status	Com. Controller Error	An error occurred in the communications controller.
	IP Address Duplicated	The same IP address is assigned to more than one node.
	On-Line	The Unit is online. (The EtherNet/IP Unit can perform communications processing.)
	Multiple Switch ON	More than one data link start/stop switch changed to TRUE at the same time.
Data Link Status	Comparison Error	The remote node information in the tag data link parameters was different from the actual node information. Main causes: • The specified target does not exist. • The variable name does not match. • The connection size is different. • Connection resources are not sufficient.
Data Link Status	Tag Data Link Error	There were two or more errors in a connection as an originator.
	Invalid Parameter	An error was found in the parameters for tag data links that are saved in non-volatile memory.
	All Tag Data Link	Tag data links are communicating in all connections as the originator.
	Tag Data Link	Tag data links are communicating in one or more connections as the originator.
Configuration Error Status	Ethernet Link Status	A link is established with the Ethernet switch.
	Ethernet Basic Settings Logic Error	The following settings are incorrect: • TCP/IP settings (IP address, subnet mask, or link settings)
	IP Router Table Error	There is a mistake in the IP router table information.
	Ethernet Ext Config Logical Error	Always FALSE.
	BOOTP Server Error	One of the following errors occurred when using the BOOTP server: • The IP address received from the BOOTP server is incorrect. • A communications timeout occurred with the server.

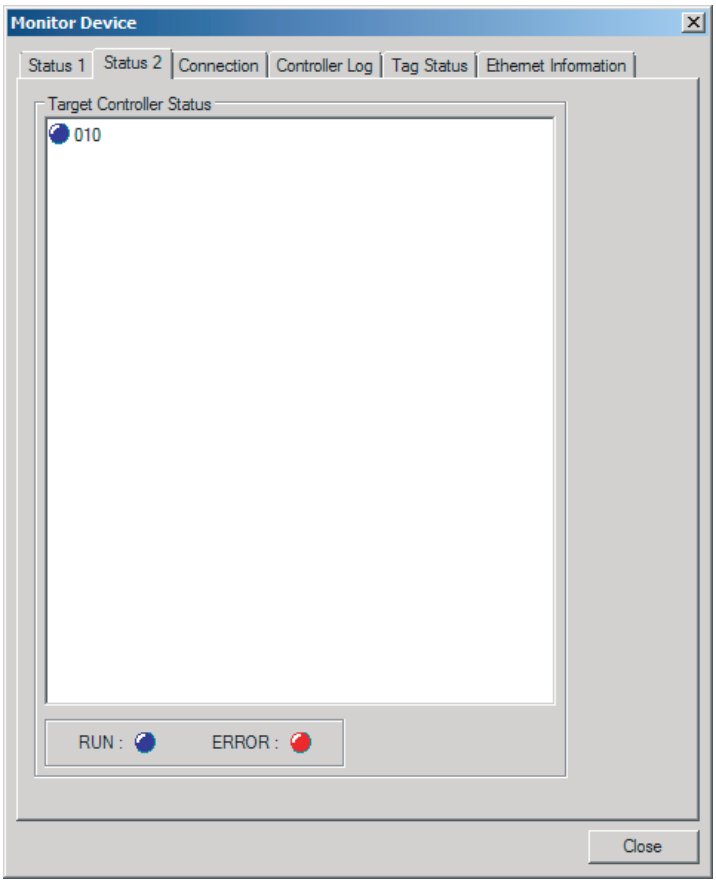
In the **Target Node Status**, information about the target node that acts as the originator is displayed.

If all tag data link connections to the node are established and normal, this information is displayed in blue. However, if any connection is broken it is displayed in red.


Monitor Device
Status 1
Status 2
Connection
Controller Log
Tag Status
Ethernet Information
Ethernet Status
Com. Controller Error
Multiple Switch ON
IP Address Duplicated
On-Line
Data Link Status
Comparison Error
All Tag Data Link
Tag Data Link Error
Tag Data Link
Invalid Parameter
Configuration Error Status
Ethernet Link Status
Ethernet Ext Config Logical Error
Ethernet Config Logical Error
BOOTP Server Error
IP Router Table Error
Target Node Status
002
Close

Status 2 Tab Page

This tab page displays information on nodes with tag data link originator settings. This information is in blue if the connection is normal, or red if an error occurred.

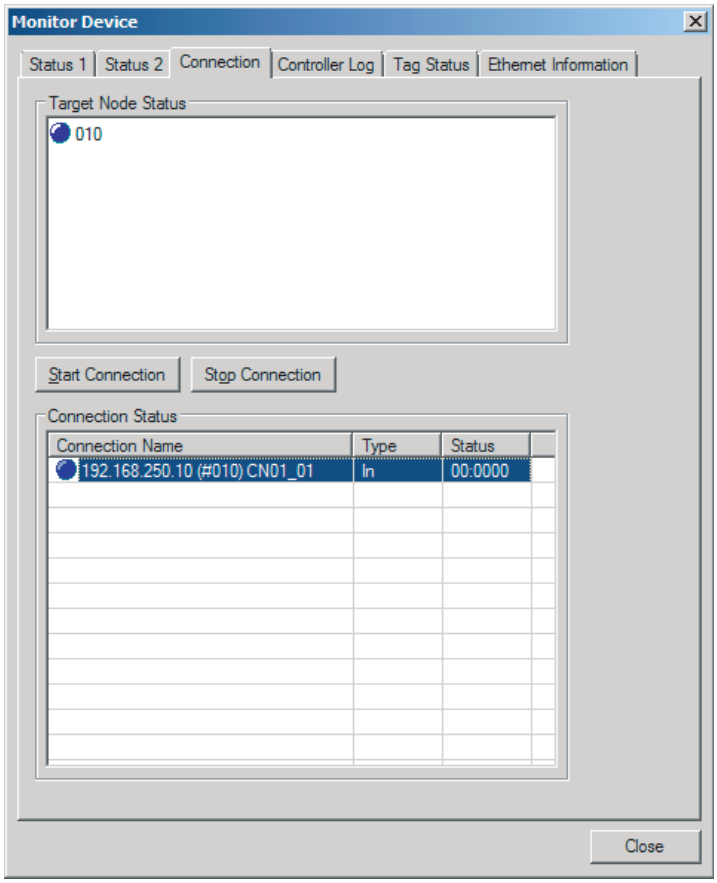


Additional Information

The target Controller status can be used when the Controller status is set to **Included** for all the target sets for both originator and target connections.
If it is set to **Not included**, it is grayed out on the display.

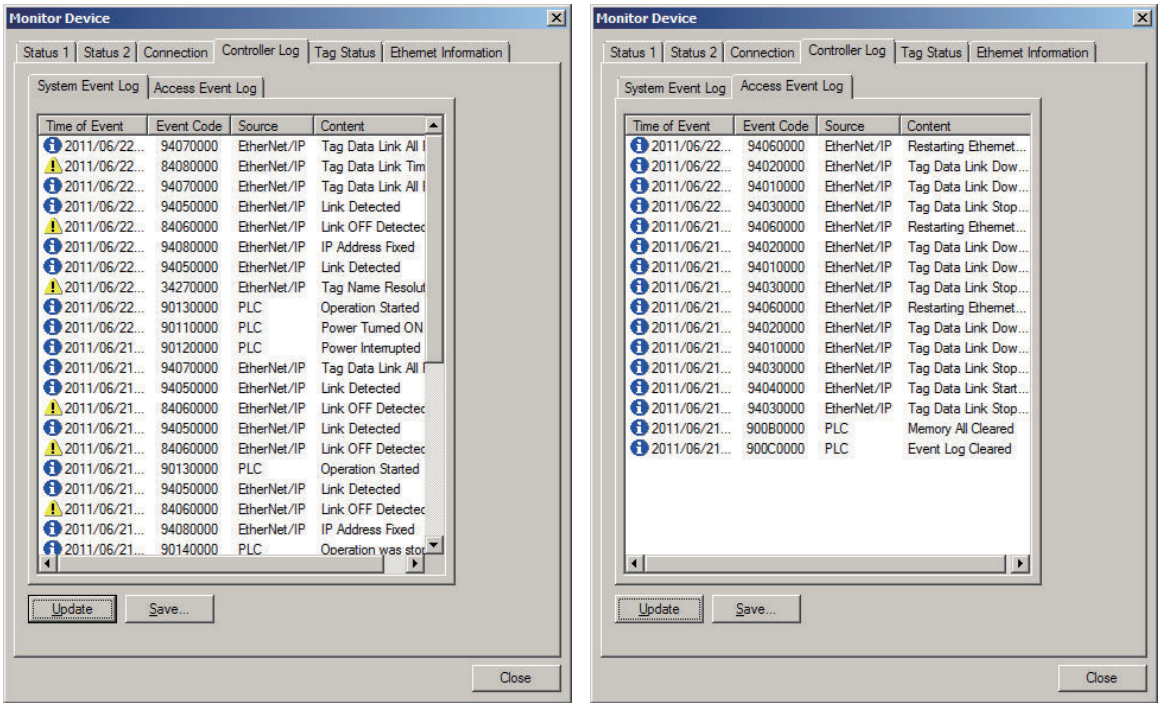
Connection Tab Page

- Target Node Status
  Information about the target node that acts as the originator is displayed.
  If all tag data link connections to the node are established and normal, this information is displayed in blue. However, if any connection is broken it is displayed in red.
  However, this information is displayed in gray if the connection to the node is stopped.
- Connection Status
  The **Status** Column of the connection status shows the status of each connection that is set as the originator. The connection status can be used to identify the cause of tag data link errors.
  Refer to *15-2-2 Connection Status Codes and Troubleshooting* on page 15-11 for details on the connection status.



● Controller Log Tab Page

This tab page displays the Controller event log that is stored in the CPU Unit.
The error history shows errors that have occurred. It can be saved in a file in the computer.
Refer to the operation manual of the CPU Unit for details on error information.



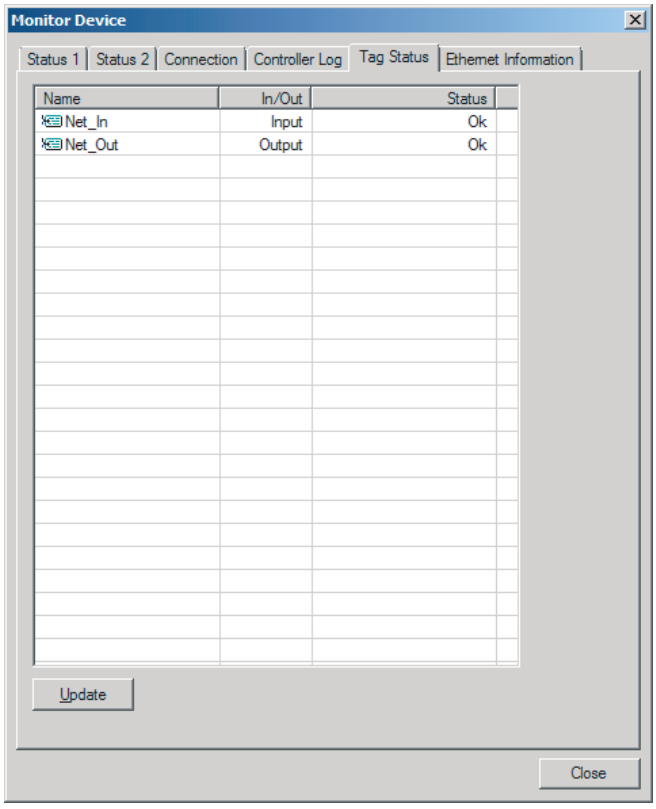
Tag Status Tab Page

This tab page displays if the tag settings for each tag for tag data links are set so that data can be exchanged with the CPU Unit.

The following status is displayed depending on the status that is set.

Normally resolved:	Normal data exchange is possible.
Resolving:	The variables with tags are being resolved. When the resolution is completed normally, a connection will be established and the data exchange will start.
Different sizes:	Different sizes are set for the network variables and the tag settings. A connection will not be established for a tag for which this error occurs.
No tag:	A network variable is not set in the variable table in the CPU Unit for the specified tag setting. Or, instead of a member of union variable, unions are specified. A connection will not be established for a tag for which this error occurs.
Attribute error:	1. Writing is not possible for Read Only and Constant attributes. 2. The I/O direction that is set in the tag data link settings does not agree with the I/O direction of the variable in the CPU Unit. There is an error in the setting of a Network Publish attribute for a CPU Unit variable. A connection will not be established for a tag for which this error occurs.

If the status is not “Normal resolution completed,” check the tag data link settings or the network variable settings in the variable table in the NJ/NX-series CPU Unit.

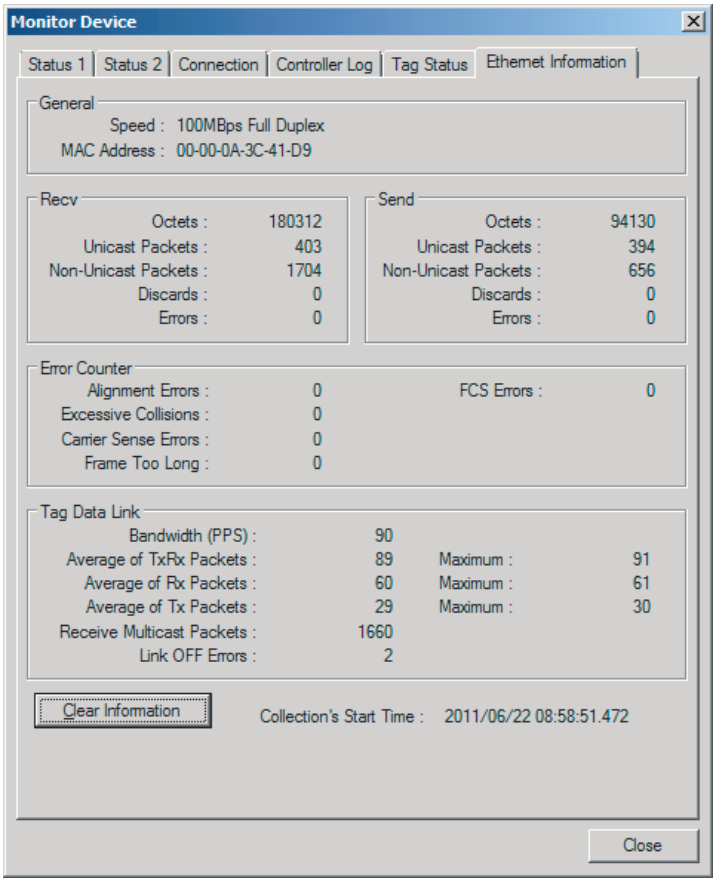


● Ethernet Information Tab Page

This tab page displays the communications status at the communications driver level of the built-in Ethernet/IP port.

The error counter information can be used to confirm whether communications problems have occurred.

The tag data link information can be used to confirm characteristics such as the Bandwidth (pps).



15-2-2 Connection Status Codes and Troubleshooting

This section explains how to identify and correct errors based on the tag data link's connection status. The connection status can be read using the **Connection** Tab Page of Monitor Device Window with the Network Configurator. Refer to *15-2-1 The Network Configurator's Device Monitor Function* on page 15-3 for details.

Additional Information

The connection status has the same meaning as the Connection Manager's General and Additional error response codes, as defined in the CIP specifications.

The following table shows the likely causes of the errors causes for each configuration and connection status (code).

	Originator	Target
Configuration 1		
Configuration 2		Products from other manufacturers
Configuration 3	Products from other manufacturers	

Connection status		Source of error	Handling		
General Status (hex)	Additional Status (hex)		Configuration 1	Configuration 2	Configuration 3
00	0000	Normal status code: The connection has been opened and the tag data link is communicating normally.	---	---	---
01	0100	Error code returned from target: Attempted to open multiple connections for the same connection.	This error does not occur.	Depends on the target's specifications. (This error should not occur. If it does, contact the target device's manufacturer.)	Depends on the originator's specifications. (This error should not occur. If it does, contact the originator device's manufacturer.)
01	0103	Error code returned from target: Attempted to open a connection with an unsupported transport class.	This error does not occur.	Confirm that the target supports Class 1.	Confirm that the originator supports Class 1.
01	0106	Duplicate consumers: Attempted to open multiple connections for single-consumer data.	If the tag data link is stopped or started, this error may occur according to the timing, but the system will recover automatically.	Depends on the target's specifications. (Contact the target device's manufacturer.)	If the tag data link is stopped or started, this error may occur according to the timing, but the system will recover automatically.

Connection status		Source of error	Handling		
General Status (hex)	Additional Status (hex)		Configuration 1	Configuration 2	Configuration 3
01	0107	Error code returned from target: Attempted to close a connection, but that connection was already closed.	This error does not occur.	This error does not occur.	This is not an error because the connection is already closed.
01	0108	Error code returned from target: Attempted to open a connection with an unsupported connection type.	This error does not occur.	Check which connection types can be used by the target. (Contact the manufacturer.) Only multicast and point-to-point connections can be set.	Check which connection types can be used by the originator. (An error will occur if a connection other than a multicast or point-to-point connection is set.)
01	0109	Error code returned from target: The connection size settings are different in the originator and target.	Check the connection (sizes) set in the originator and target.		
01	0110	Error code returned from target: The target was unable to open the connection, because of its operating status, such as downloading settings.	Check whether the tag data link is stopped at the target. (Restart the tag data link communications with the software switch.)	Depends on the target's specifications. (Contact the target device's manufacturer.)	Check whether the tag data link is stopped at the target. (Restart the tag data link communications with the software switch.)
01	0111	Error code returned from target: The RPI was set to a value that exceeds the specifications.	This error does not occur.	Check the target's RPI setting specifications.	Set the originator's RPI setting to 10 seconds or less.
01	0113	Error code generated by originator or returned from target: Attempted to open more connections than allowed by the specifications (32).	Check the connection settings (number of connections) at the originator and target.	Check the connection settings (number of connections) at the originator and target. Check the connection specifications for devices from other manufacturers.	Check the connection settings (number of connections) at the originator and target. Check the connection specifications for devices from other manufacturers.

Connection status		Source of error	Handling		
General Status (hex)	Additional Status (hex)		Configuration 1	Configuration 2	Configuration 3
01	0114	Error code returned from target: The Vendor ID and Product Code did not match when opening connection.	This error does not occur.	Depends on the target's specifications. (Contact the target device's manufacturer.) Check that the target device's EDS file is correct.	Check the originator's connection settings.
01	0115	Error code returned from target: The Product Type did not match when opening connection.	This error does not occur.	Depends on the target's specifications. (Contact the target device's manufacturer.) Check that the target device's EDS file is correct.	Check the originator's connection settings.
01	0116	Error code returned from target: The Major/Minor Revisions did not match when opening connection.	Check the major and minor revisions set for the target device and connection. If necessary, obtain the most recent EDS file and set it again.	Depends on the target's specifications. (Contact the target device's manufacturer.) Check that the target device's EDS file is correct.	Check the originator's connection settings.
01	0117	Error code returned from target: The tag set specified in the connection's target variables does not exist.	Check whether the originator and target tag sets and tags are set correctly.	Depends on the target's specifications. (Contact the target device's manufacturer.)	Check the originator's connection settings. Check whether the target tag sets and tags are set correctly.

Connection status		Source of error	Handling		
General Status (hex)	Additional Status (hex)		Configuration 1	Configuration 2	Configuration 3
01	011A	Error code generated by originator: Connection could not be established because the buffer was full due to high traffic.	Unexpected network traffic may have been received. Use the Ethernet Information Tab Page of the Network Configurator's device monitor to check the bandwidth usage, and correct the load. If there are places where broadcast storms occur, such as loop connections in the network connection format, then correct them.	Unexpected network traffic may have been received. Use the Ethernet Information Tab Page of the Network Configurator's device monitor to check the bandwidth usage, and correct the load. If there are places where broadcast storms occur, such as loop connections in the network connection format, then correct them.	Depends on the target's specifications. (Contact the target device's manufacturer.)
01	011B	Error code returned from target: The RPI was set to a value that is below the specifications.	This error does not occur.	Depends on the target's specifications. (Contact the target device's manufacturer.)	Set the originator's RPI setting to 1 ms or greater.
01	0203	Error code generated by originator: The connection timed out.	Tag data link communications from the target timed out. Check the power supply and cable wiring of the devices in the communications path, including the target and switches. If performance has dropped due to heavy traffic, change the performance settings. For example, increase the timeout time or RPI setting. Also, check whether CIP message communications are stopped or CIP communications are permitted by the originator and Packet Filter function of the device on the route.		
01	0204	Error code generated by originator: The connection open process timed out.	There was no response from the target. Check the power supply and cable wiring of the devices in the communications path, including the target and switches. Also, check whether the CIP message communications of the target or originator are stopped and whether the CIP communications are permitted by Packet Filter function of the target device or the device on the route.		

Connection status		Source of error	Handling		
General Status (hex)	Additional Status (hex)		Configuration 1	Configuration 2	Configuration 3
01	0205	Error code returned from target: There was a parameter error in the frame used to open the connection.	This error does not occur.	Depends on the target's specifications. (Contact the target device's manufacturer.)	Depends on the originator's specifications. (Contact the originator device's manufacturer.)
01	0301	Error code generated by originator or returned from target: Total number of tag sets that are set to the product was exceeded.	Check the total number of the tag sets that are set to the product and set the tag sets so that the total number does not exceed the maximum of the allowable number.	Check the total number of the tag sets that are set to the product and set the tag sets so that the total number does not exceed the maximum of the allowable number.	Check the total number of the tag sets that are set to the product and set the tag sets so that the total number does not exceed the maximum of the allowable number.
01	0302	Error code generated by originator or returned from target: The tag data link's allowable bandwidth (pps) was exceeded.	Check the connection settings (number of connections and RPI) at the originator and target.	Check the target's connection settings (number of connections and RPI). Check the connection settings (number of connections and RPI) at the originator and target.	Check the connection settings (number of connections and RPI) at the originator and target.
01	0311	Error code returned from target: There was a parameter error in the frame used to open the connection.	This error does not occur.	Depends on the target's specifications. (Contact the target device's manufacturer.)	Depends on the originator's specifications. (Contact the originator device's manufacturer.)
01	0312	Error code returned from target: There was a parameter error in the frame used to open the connection.	This error does not occur.	Depends on the target's specifications. (Contact the target device's manufacturer.)	Depends on the originator's specifications. (Contact the originator device's manufacturer.)

Connection status		Source of error	Handling		
General Status (hex)	Additional Status (hex)		Configuration 1	Configuration 2	Configuration 3
01	0315	Error code returned from target: There was a parameter error in the frame used to open the connection.	This error does not occur.	Depends on the target's specifications. (Contact the target device's manufacturer.)	Depends on the originator's specifications. (Contact the originator device's manufacturer.)
01	0316	Error code returned from target: There was a parameter error in the frame used to close the connection.	This error does not occur.	Depends on the target's specifications. (Contact the target device's manufacturer.)	Depends on the originator's specifications. (Contact the originator device's manufacturer.)
01	031C	Error code generated by originator: Some other error occurred.	This error does not occur.	The originator generates this code when an unsupported response code is returned from the target in reply to an open request.	Depends on the originator's specifications. (Contact the originator device's manufacturer.)
08	---	Error code returned from target: There is no Forward Open or Large Forward Open service in the target device.	This error does not occur.	Depends on the target's specifications. (Contact the target device's manufacturer.)	Depends on the originator's specifications. (Contact the originator device's manufacturer.)

Connection status		Source of error	Handling		
General Status (hex)	Additional Status (hex)		Configuration 1	Configuration 2	Configuration 3
D0	0001	Error code generated by originator: The connection operation is stopped.	The connection was stopped because the Tag Data Link Stop Bit was turned ON, or the settings data is being downloaded. Either turn ON the Tag Data Link Start Switch, or wait until the settings data has been downloaded. This code includes fatal Controller errors and Unit failure. To handle these errors, refer to the NJ/NX-series Troubleshooting Manual (Cat. No. W503).	The meaning of this error code is defined by each vendor, so it depends on the target's specifications. (Contact the target device's manufacturer.)	Depends on the originator's specifications. (Contact the originator device's manufacturer.)
D0	0002	Error code generated by originator: The connection is being opened (opening processing in progress).	Wait until the opening processing is completed.	The meaning of this error code is defined by each vendor, so it depends on the target's specifications. (Contact the target device's manufacturer.)	Depends on the originator's specifications. (Contact the originator device's manufacturer.)

Connection status		Source of error	Handling		
General Status (hex)	Additional Status (hex)		Configuration 1	Configuration 2	Configuration 3
OMRON error code					
01	0810	Error code returned from target: The latest data cannot be retrieved from the CPU Unit after a connection was opened. (Automatic recovery by connection open retry)	It occurs when the CPU Unit task period is too long after a connection was opened or when the controller system stopped due to an error on the controller. If it occurred due to a long task period, the error will be recovered automatically. If it was caused by stoppage of the controller system, the cause of the error will be identified from the error information of the CPU Unit.	The meaning of this error code is defined by each vendor, so it depends on the target's specifications. (Contact the target device's manufacturer.)	The meaning of this error code is defined by each vendor, so it depends on the originator's specifications. (Contact the originator device's manufacturer.)
01	0811	Error code generated by originator: The latest data cannot be retrieved from the CPU Unit after a connection was opened. (Automatic recovery by connection open retry)	It occurs when the CPU Unit task period is too long after a connection was opened. If the task period was too long, operation recovers automatically.	The meaning of this error code is defined by each vendor, so it depends on the target's specifications. (Contact the target device's manufacturer.)	The meaning of this error code is defined by each vendor, so it depends on the originator's specifications. (Contact the originator device's manufacturer.)

A

Appendices

A-1 Functional Comparison of EtherNet/IP Ports on NJ/NX-series CPU Units and Other Series

OK: Supported, ---: Not supported

Item	Built-in EtherNet/IP port on NX701 CPU Unit	Built-in EtherNet/IP port on NX102 CPU Unit	Built-in EtherNet/IP port on NX1P2 CPU Unit	Built-in EtherNet/IP port on NJ-series CPU Unit	CJ-series Ethernet Unit	EtherNet/IP Unit (built-in port on CJ2 CPU Unit)		
						Unit version 1.0	Unit version 2.0	Unit version 2.1
Tag data link communications service	OK	OK	OK	OK	---	OK	OK	OK
CIP message communications service	OK	OK	OK	OK	---	OK	OK	OK
IP routing	OK	OK	---	---	---	---	---	---
Socket service	OK	OK	OK	OK	OK	---	---	---
FTP server	OK	OK	OK	OK	OK	---	OK	OK
FTP client	OK	OK	OK	OK	---	---	---	---
Mail send/receive	---	---	---	---	OK	---	---	---
Web functions	---	---	---	---	OK	---	---	---
Automatic adjustment of PLC/Controller's internal clock	OK	OK	OK	OK	OK	---	OK	OK
Error history	OK*1	OK*1	OK*1	OK*1	OK	OK	OK	OK
Response to PING command	OK	OK	OK	OK	OK	OK	OK	OK
SNMP/SNMP trap	OK	OK	OK	OK	---	---	OK	OK
CIDR function for IP addresses	OK	OK	OK	OK	---	---	OK	OK
Online connection via EtherNet/IP using CX-One	---	---	---	---	OK	---	OK	OK
Online connection via EtherNet/IP using Network Configurator	OK	OK	OK	OK	---	OK	OK	OK
Mounting in an NJ-series CPU Unit	---	---	---	---	---	---	---	OK*2
Connection settings using the Sysmac Studio	OK	OK	OK	OK	---	---	---	OK

*1. This is equivalent to the event log in the built-in EtherNet/IP of an NJ-series Controller.

*2. You cannot use the following functions if you connect to the CPU Unit through an EtherNet/IP Unit.

- Placing the Sysmac Studio online with the CPU Unit (However, you can place the Network Configurator online)
- Using the Troubleshooter of an NS-series PT

A-2 Use the Sysmac Studio to Set the Tag Data Links (EtherNet/IP Connections)

A-2-1 Overview of the Tag Data Links (EtherNet/IP Connections) Settings with the Sysmac Studio

You can use the Sysmac Studio to set the settings required for creating tag data links (EtherNet/IP connections)*1 between NJ/NX-series Controllers.

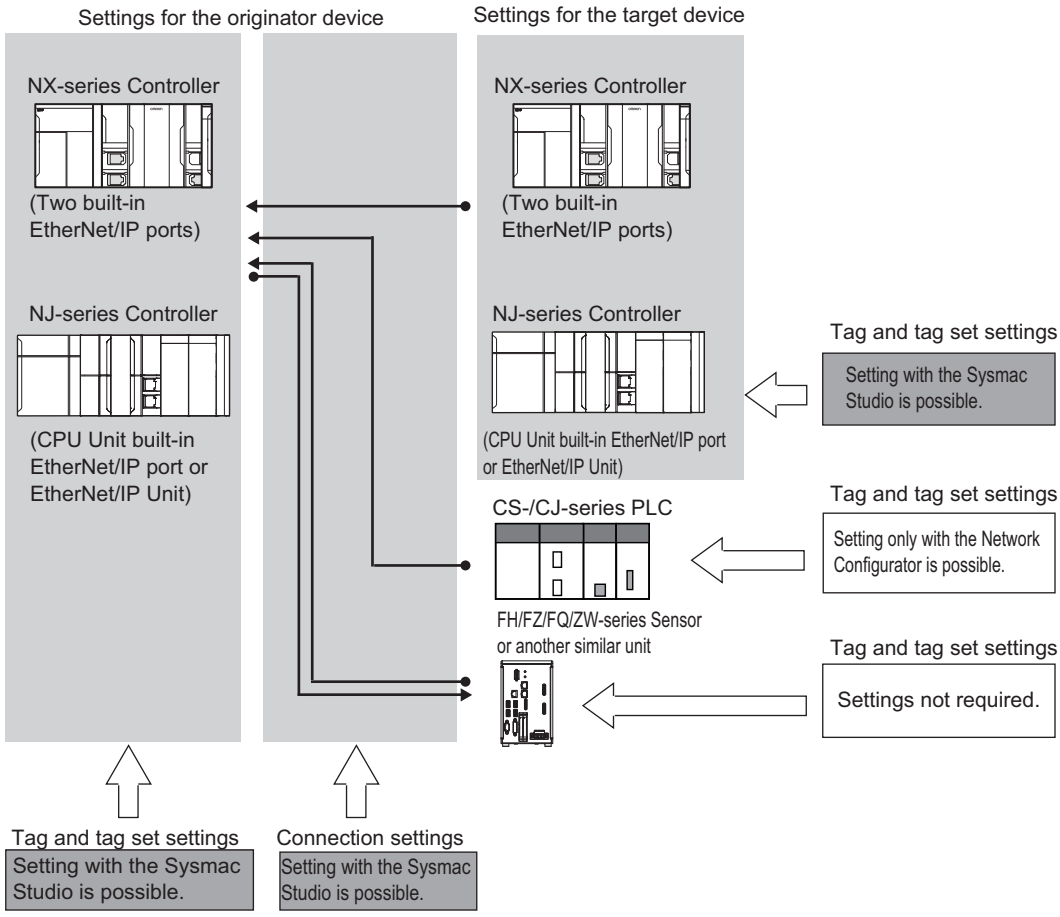
*1. The tag data links and EtherNet/IP connections enable cyclic tag data exchanges on an EtherNet/IP network between Controllers or between Controllers and other devices. Here, "EtherNet/IP connection" refers to both the tag data links and the EtherNet/IP connections.

Version Information

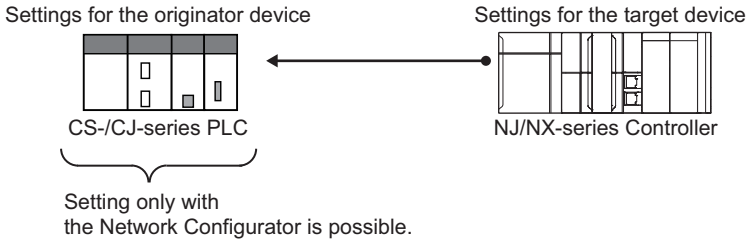
Sysmac Studio version 1.10 or higher is required to use the Tag Data Link (EtherNet/IP Connection) Settings.

Acceptable System Configuration Conditions for Setting the EtherNet/IP Connection Settings on the Sysmac Studio

If an NJ/NX-series Controller operates as the originator device, you can use the Sysmac Studio to set the originator device settings for the EtherNet/IP connections.
Similarly, if an NJ/NX-series Controller operates as the target device, you can use the Sysmac Studio to set the tags and tag sets of the target device.



Use the Network Configurator if a CS/CJ-series PLC operates as the originator device.



A-2-2 Procedure to Make the EtherNet/IP Connection Settings with the Sysmac Studio

1 Registering devices			• Main Window
↓	Register devices with which the EtherNet/IP connections are established to the project.		Setup Window
2 Creating network variables[*1]	---	Refer to *Registering the Network Variable for the Originator Device* on page A-11.	Global Variable Table on the Sysmac Studio
↓			
3 Registering tags and tag sets	---	Refer to *Registering the Tag and Tag Set* on page A-12.	EtherNet/IP Connection Settings (Tag Set Display)

↓	Register the network variables that are set in step 2 as tags and tag sets.		
4 Setting Connections	---	Refer to *Setting Connections for the Originator Device* on page A-15.	EtherNet/IP Connection Settings (Connection Display)
↓	Specify devices (i.e., target devices and originator devices) and tag sets to communicate with using the EtherNet/IP connections.		
5 Going online from the Sysmac Studio	---	Refer to *Transferring the Connection Settings Data* on page A-27.	• Main Window
↓			
6 Downloading EtherNet/IP connection settings Note Connections automatically start after the download.	---	Refer to *Transferring the Connection Settings Data* on page A-27.	• Synchronization Window/ Transfer to Controller Dialog Box • EtherNet/IP Connection Settings
↓			
7 Checking operation Stopping and starting connections	---	Refer to *A-2-5 Checking Communications Status with the Sysmac Studio and Troubleshooting* on page A-31.	EtherNet/IP Connection Monitor Tab Page

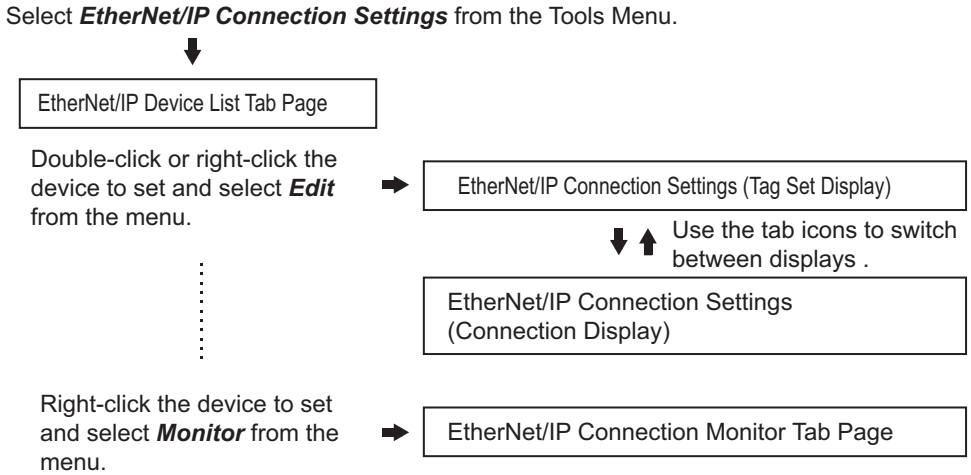
*1. Variables with its Network Publish attribute set to **Output** or **Input** in the Global Variable Table are called network variables.

A-2-3 EtherNet/IP Connection Settings

This section describes the screen configuration for EtherNet/IP connection settings.

Screen Transitions in the EtherNet/IP Connection Settings

- Connection Settings



- Transferring connection settings to the Controller from the computer

Select ***Synchronization*** from the Controller Menu.

↓

Synchronization Window

Select ***Transfer - To Controller*** from the Controller Menu.

↓

Transfer to Controller Dialog Box

Select ***EtherNet/IP Connection Settings*** from the Tools Menu to display the device list. Right-click the device and select ***Edit*** from the menu.

↓

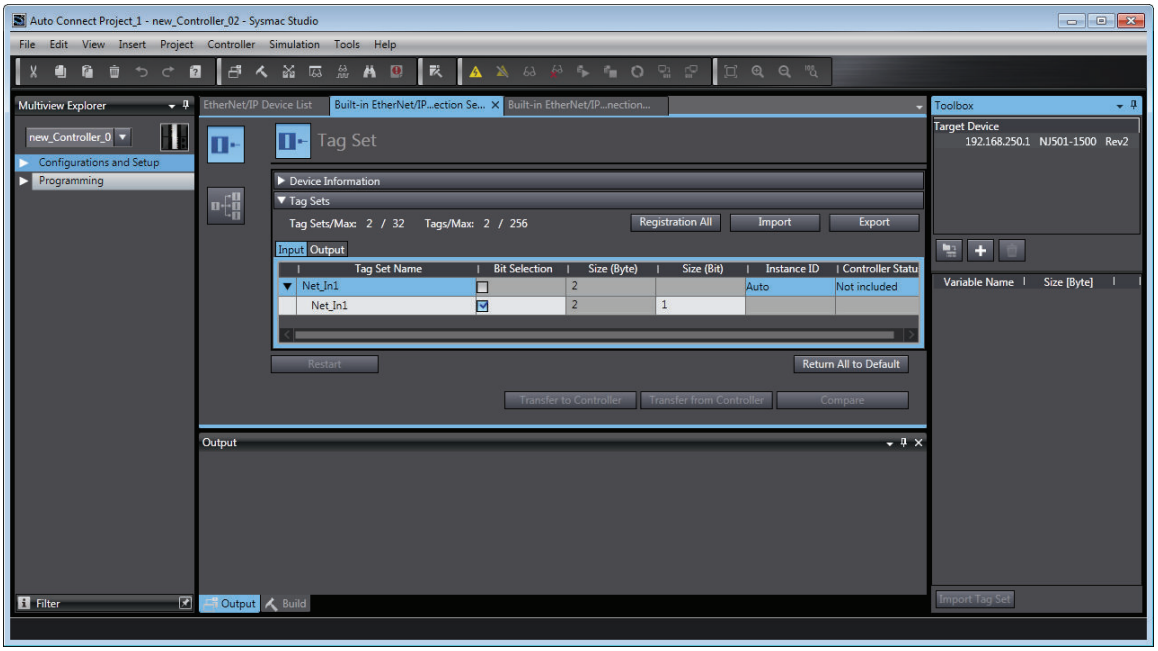
EtherNet/IP Connection Setting Tab Page

Precautions for Correct Use

To transfer only the connection settings, execute Transfer from the EtherNet/IP Connection Setting Tab Page.
Even if you clear the **Do not transfer the connection setting** Check Box, the connection settings are not transferred from the Synchronization Window, the **Transfer to the Controller** Dialog Box, or the **Transfer from the Controller** Dialog Box as long as the data in the computer is synchronized with the data in the Controller.

EtherNet/IP Device List Tab Page

The list indicates the devices to which EtherNet/IP connections can be set.
For information on how to access this tab page, refer to the *Registering the Tag and Tag Set* on page A-12.



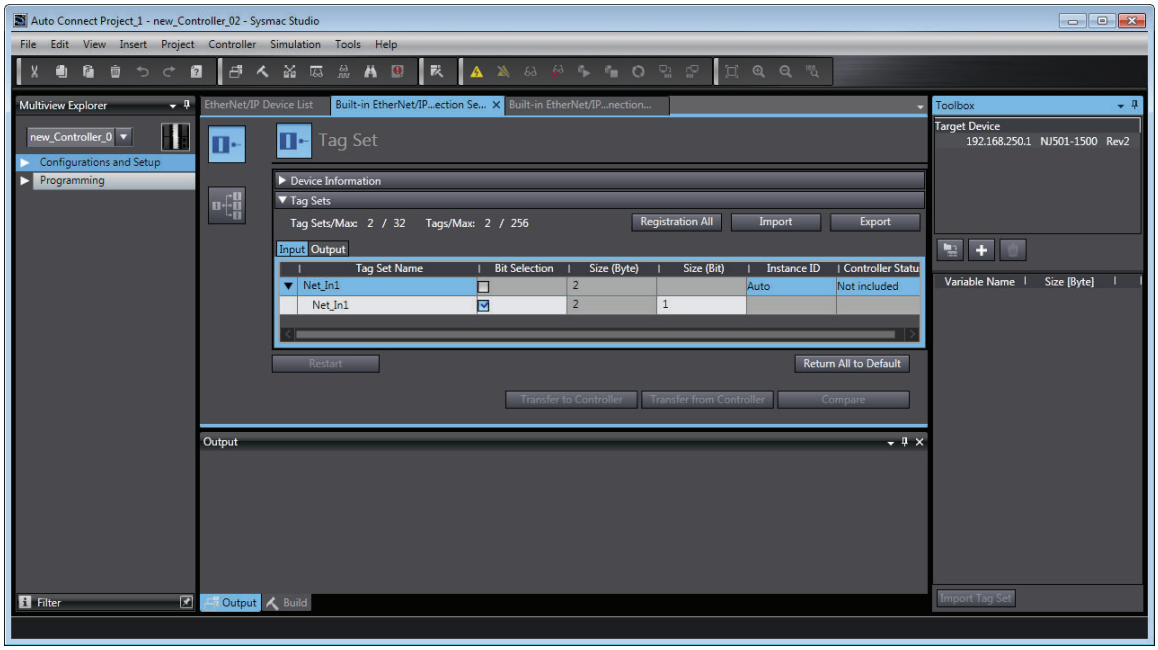
EtherNet/IP Connection Settings (Tag Set Display)

Register tag sets required to create connections.
Each tag set represents the data that is sent and received through a connection. You can register up to eight tags in one tag set.

The name and size of the tag must be the same as those of the network variable *1. Set whether to include the Controller status information in tags for the tag sets. You can also set the data output operation at a fatal error occurrence for output tags.

Refer to *Registering the Tag and Tag Set* on page A-12 for information on how to register tags and tag sets.

*1. A variable with its **Network Publish attribute** set to **Output** or **Input** in the Global Variable Table is called a network variable.



EtherNet/IP Connection Settings (Connection Display)

Specify the target devices and set their connections.

For each connection, set the following information: Connection Name, Connection I/O Type, I/O, target device tag set (target variable), originator device tag set (originator variable), Packet Interval (RPI), and Timeout Value.

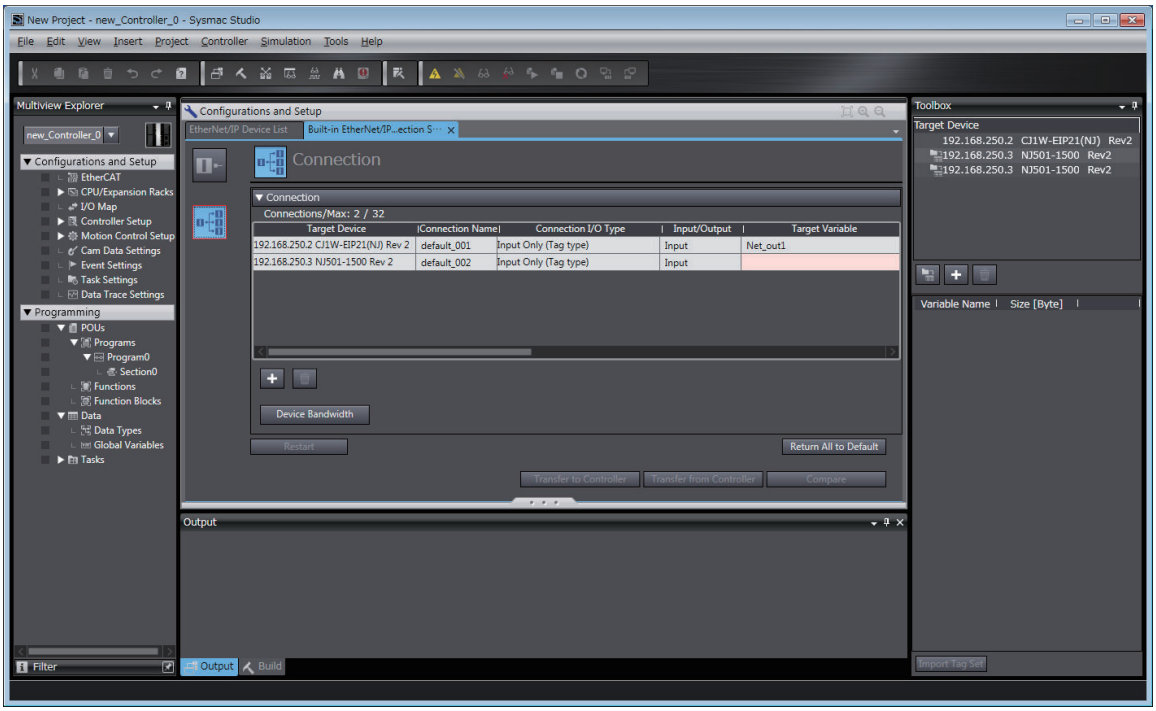
Refer to *Setting Connections for the Originator Device* on page A-15 for information on how to make connection settings.

Precautions for Correct Use

If you changed the IP address, model, or revision of the target device after making the connection settings, perform the following.

With the Sysmac Studio version 1.11 or higher, change the connection settings entirely.

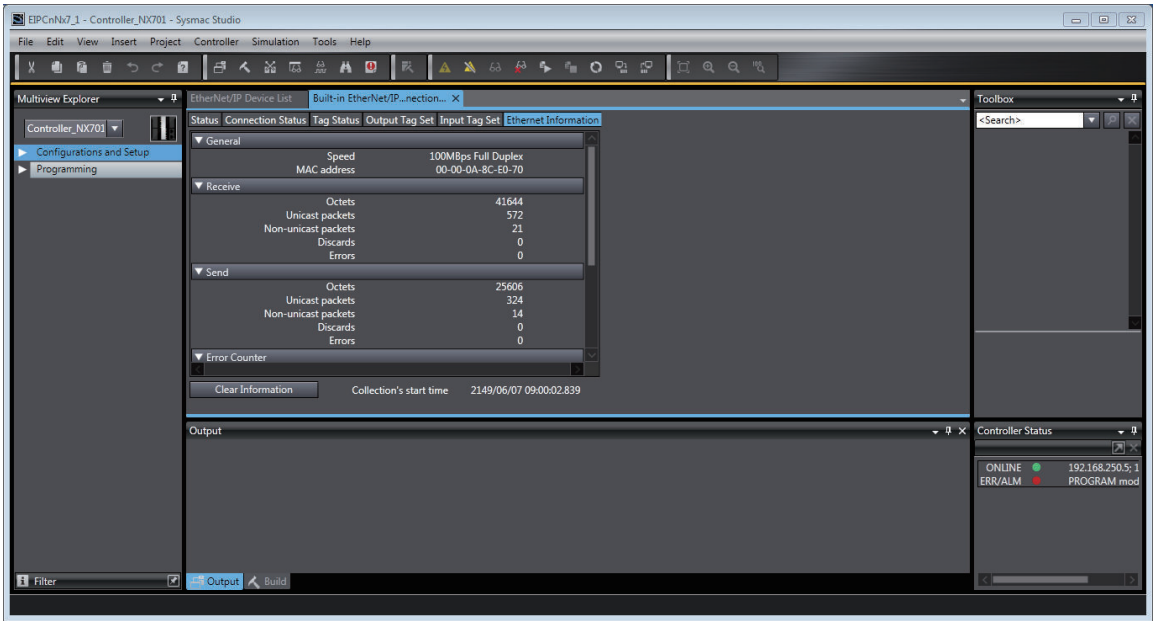
With the Sysmac Studio version 1.10 or lower, create the connections again.



EtherNet/IP Connection Monitor Tab Page

You can check the EtherNet/IP connection setting status offline and communications status online. When online, you can start and stop connections.

Refer to *A-2-5 Checking Communications Status with the Sysmac Studio and Troubleshooting* on page A-31 for information on how to check the EtherNet/IP connection setting status and communications status.



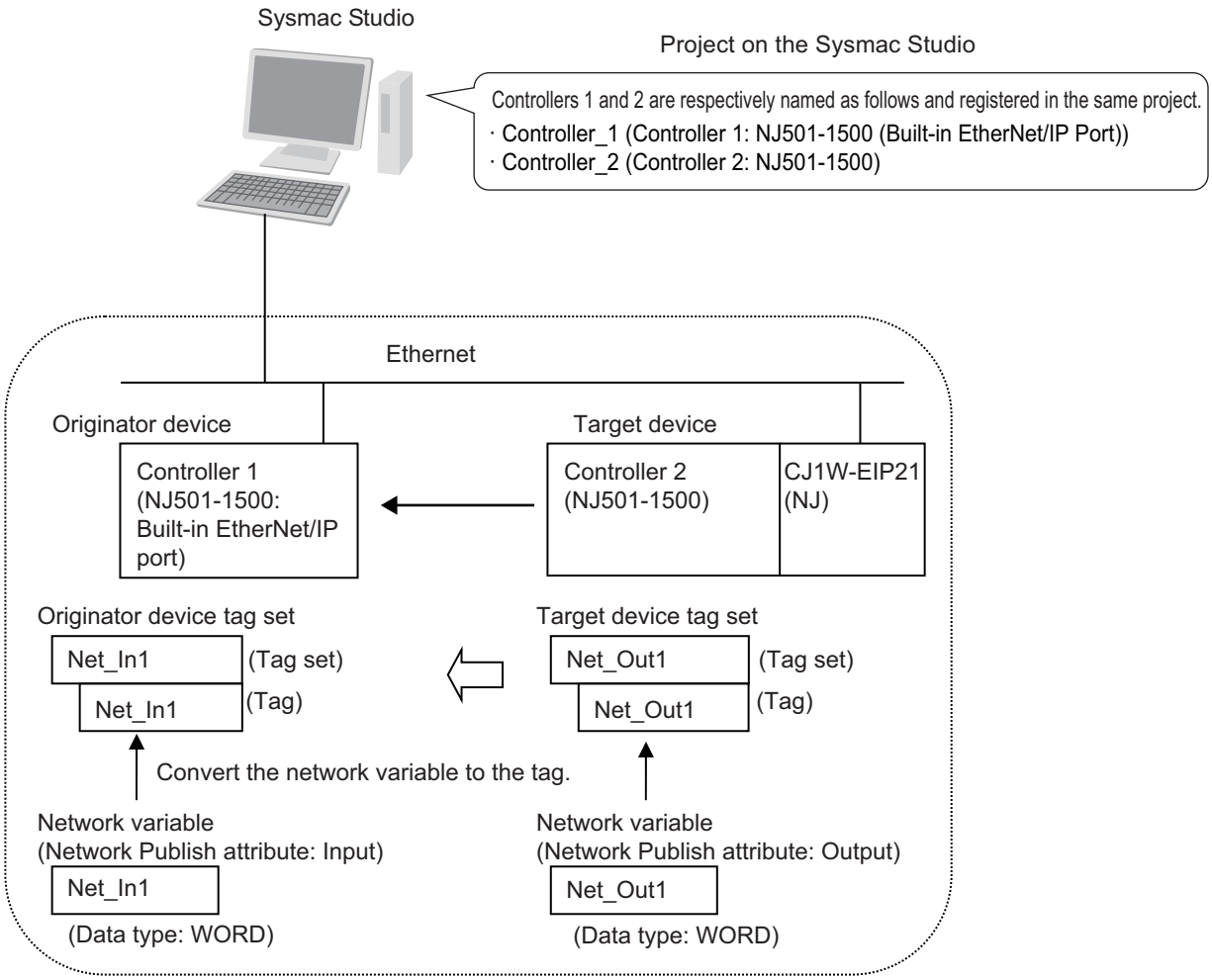
A-2-4 Making the EtherNet/IP Connection Settings with the Sysmac Studio

This section describes the procedure to make the EtherNet/IP connection settings with the Sysmac Studio.

Here, we take the following system configuration as an example to describe how to set the EtherNet/IP connection settings.

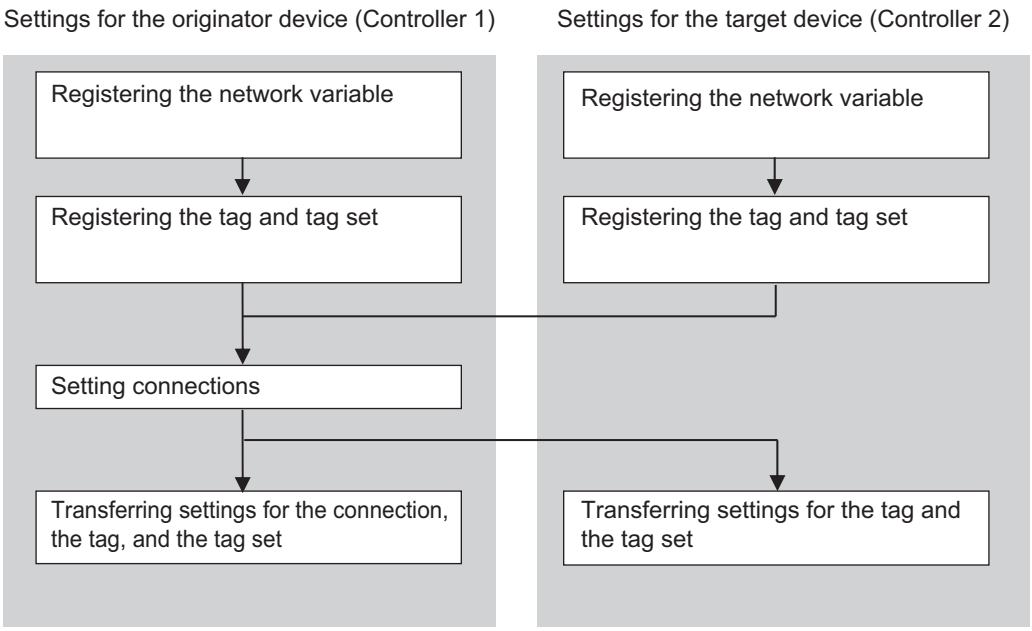
Example: System that connects the built-in EtherNet/IP port on Controller 1 and the built-in EtherNet/IP port on Controller 2 via Ethernet

- Set the settings so that values in the network variable Net_Out1 allocated for Controller 2 are sent to the network variable Net_In1 allocated for Controller 1 at the set RPI of 50 ms cycle.
- This example assumes the programs for both Controllers 1 and 2 are registered in the same project.



Follow the flow below to set the settings to Controllers 1 and 2 for which to establish EtherNet/IP connections.

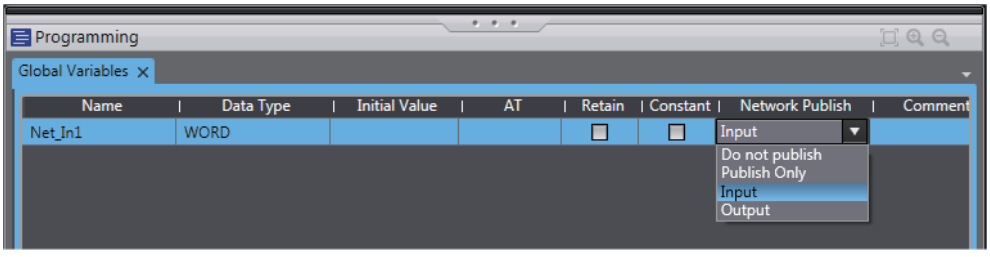
The required settings for the originator device and the target device are shown below.



Registering the Network Variable for the Originator Device

Register the network variable that is sent and received using the EtherNet/IP connections.
Refer to the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for the operations for registering variables.

1 Assign the network variable to the tag used for the EtherNet/IP connection for Controller 1 (originator device).
This network variable receives data from Controller 2 (target device).
Select **Input** or **Output** for **Network Publish** of a variable in the Global Variable Table so that the variable can serve as a network variable, i.e. the variable can be used for the EtherNet/IP connections.
In this example, set the network variable for Controller 1 as shown below.



- Variable name: Net_In1
- Data type: WORD
- Network Publish attribute: Input

● Network Variables Used for EtherNet/IP Connections

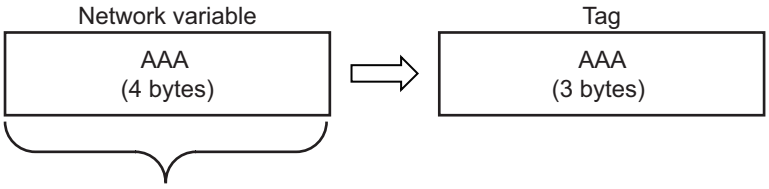
- Network variable name

You cannot specify an I/O memory address for a tag name in the EtherNet/IP connection settings. Thus, do not specify an I/O memory address for the network variable name that is to be assigned to a tag.
The following text strings are recognized as the I/O memory address names.

1. Variable names that contain only single-byte numerals from 0000 to 6143
2. Variable names with the following single-byte letters (uppercase or lowercase) followed by single-byte numerals
   - H (H000 to H511)
   - W (W000 to W511)
   - D (D00000 to D32767)
   - E0_ to E18_(E0_00000 ... E0_32767 to E18_00000 ... E18_32767)

To specify an I/O memory address for a tag on an NJ-series CPU Unit, NX102 CPU Unit, or NX1P2 CPU Unit, do not specify the I/O memory address for the tag directly. Instead, create a variable, set an AT specification of the I/O memory address on the Sysmac Studio, and then specify the variable with the AT specification for the tag.

- Size of variables
  To use an EtherNet/IP Unit as an EtherNet/IP device, set an even number of bytes for the size of the network variable used for the EtherNet/IP connections regardless of an odd number of bytes for the tag size.



The CPU Unit memory is consumed in units of two bytes. To assign tags of odd numbers of bytes to network variables, specify even byte numbers (i.e., sizes of the tags + 1) to the network variables.

- Data concurrency
  To maintain concurrency in the values of network variables that are assigned to tags, you must set refreshing tasks.
  Refer to *6-1-7 Concurrency of Tag Data Link Data* on page 6-12 for details.

Registering the Tag and Tag Set

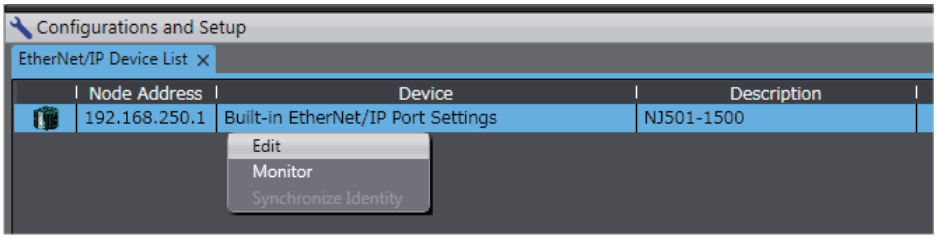
Register the required tag and tag set for the EtherNet/IP connections.
You can register tags and tag sets in the EtherNet/IP Connection Setting Tab Page.

Precautions for Correct Use

Make the following settings to refresh all of the tag data in the same tag set at the same time.
- Use the Sysmac Studio, in advance, to specify the same refreshing task for all of the variables that are assigned to tags in the tag set.
- If you use the NJ-series CPU Unit, do not place tag variables that have AT specifications in I/O memory and tag variables that do not have AT specifications in the same tag set.

1 Select **EtherNet/IP Connection Settings** from the **Tools** Menu.
The EtherNet/IP Device List Tab Page is displayed.

2 In this example, right click **Built-in EtherNet/IP Port Settings** for the originator device and select **Edit** from the menu to open the EtherNet/IP Connection Setting Tab Page.



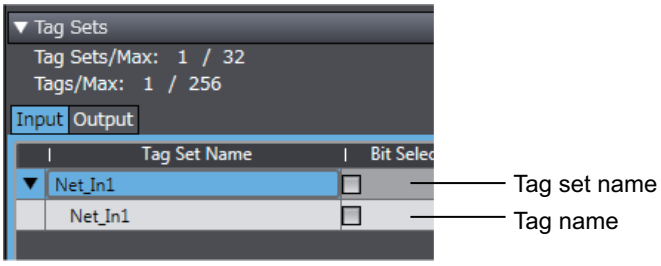
3 Click the  (Show Tag Set Display) icon in the EtherNet/IP Connection Setting Tab Page.

4 Click the **Input** tab to switch to the **Input** Tab Page. Register the tag set and the tag.
Use one of the following methods to register the tag set and the tag.

- Independent registration : Manually registers network variables in the Controller as tags.
- Batch registration : Registers all network variables in the Controller as tags at the same time.

5 Register tags and tag sets independently.

1) Right-click anywhere in the Input Tab Page of the EtherNet/IP Connection Setting Tab Page and select **Create New Tag Set** from the menu.
2) Enter the tag set name, *Net_In1*, directly into the list in the **Input** Tab Page.
3) Right-click anywhere in the Input Tab Page and select **Create New Tag** from the menu.
4) Enter tag name *Net_In1*.



Precautions for Correct Use

Any name can be specified for the tag set if the name matches one of the registered network variable names in the Controller.
As you enter characters (or immediately after you press the Ctrl + Space Keys), the Sysmac Studio Entry Assistance provides a list of variable names registered in the Controller. Select the variable name from the list.

Additional Information

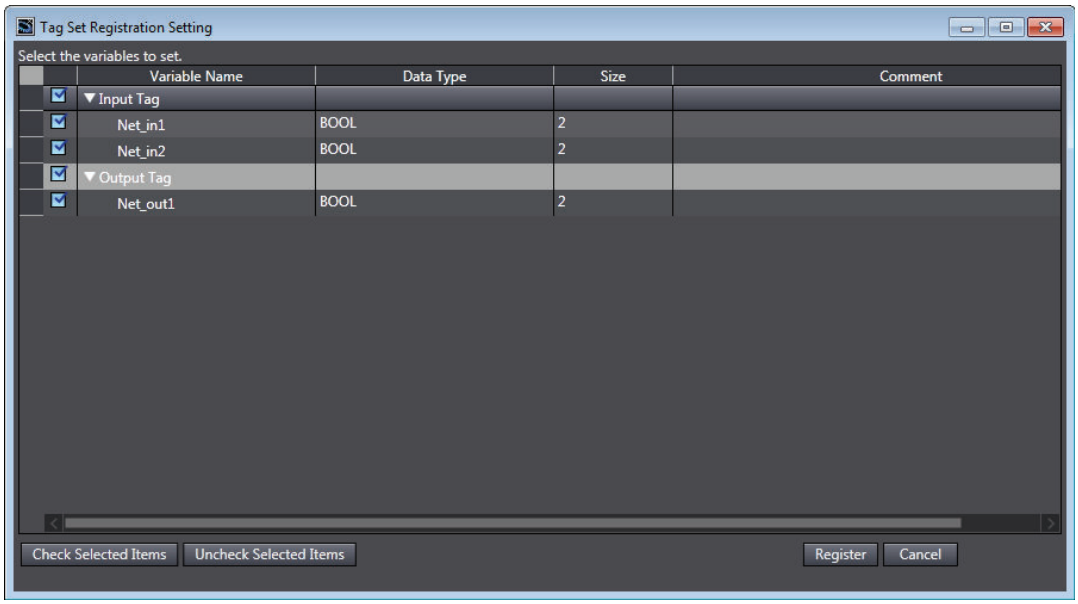
You can register up to 8 tags in a tag set.
Set as shown below to register multiple tags.
Examples:

	Tag set name	
▼	Network_Input_Value	.... (Tag set name)
	Net_In1	.... (Tag name)
	Net_In2	.... (Tag name)

6 Register all tags and tag sets at the same time.

1) Right-click anywhere on the Input Tab Page of the EtherNet/IP Connection Settings Tab Page and select **Register All Tag Sets** or click the **Registration All** Button to display the **Tag Set Registration Setting** Dialog Box.

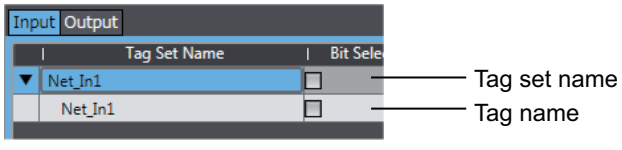
   This dialog box lists the variables that are registered in the Global Variable Table and also have the **Network Publish** attribute set to **Input** or **Output**.



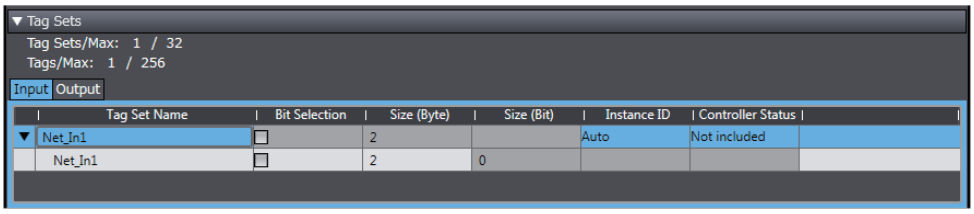
2) Select the variable to register as a tag, and then click the **Register** Button.

3) The automatically registered tag is added to the list in the EtherNet/IP Connection Setting Tab Page.

   With automatic registration, the tag is registered under a tag set having the same name as the tag, i.e., a single tag is registered in a single tag set.



7 Set the following settings for the registered tag and tag set.



- Setting for Tag Sets

Name	Item
Tag Set Name	Enter the tag set name. You can change the names as required.
Size (Byte)	Gives the total size of the tag in bytes.
Instance ID	Gives the instance ID. • Auto • IN_{min}...IN_{max} {min} represents the minimum number of Produced Assembly identification numbers recorded in the EDS files for the relevant devices. {max} represents the maximum number of Produced Assembly identification numbers recorded in the EDS files for the relevant devices.
Controller Status	Specify whether to include the Controller status in the tag set.

- Setting for Tags

Name	Item
Tag Name	Enter the tag name. Specify the tag name that matches one of the registered network variable names in the Controller.
Bit Selection	Specify whether to set the tag data size in bits. Selected: Set the size in bits. Not selected: Set the size in bytes.
Size (Byte)	Gives the size of the tag in bytes.
Size (Bit)	Gives the size of the tag in bits.
Output at Fatal Error	Specify whether to clear the output data or continue to send it when a major fault level Controller error occurs in the Controller. • Retained • Cleared

Setting Connections for the Originator Device

After the tag set registration, set the connection settings for transferring data using the EtherNet/IP connections.
Make the connection settings in the originator device (i.e., Controller 1 in this example) only.

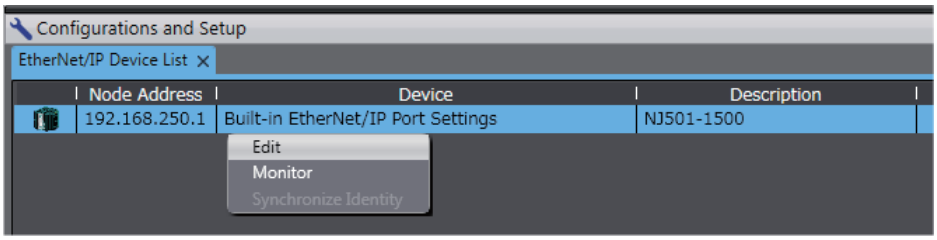
Register the tag and tag set for Controller 2 (Target device) before setting the connection settings as described in this example.
Refer to *Registering the Tag and Tag Set for the Target Device* on page A-23 for the operations for registering tags and tag sets.

Precautions for Correct Use

If you change the IP address, model, or revision of the target device after making the connection settings, you must also change the target device settings that are included in the connection settings.
For information on how to change the target device settings in the connection settings, refer to *Changing the Target Device Settings after Making Connection Settings* on page A-21.

1 Select **EtherNet/IP Connection Settings** from the **Tools** Menu to display the EtherNet/IP Device List Tab Page.

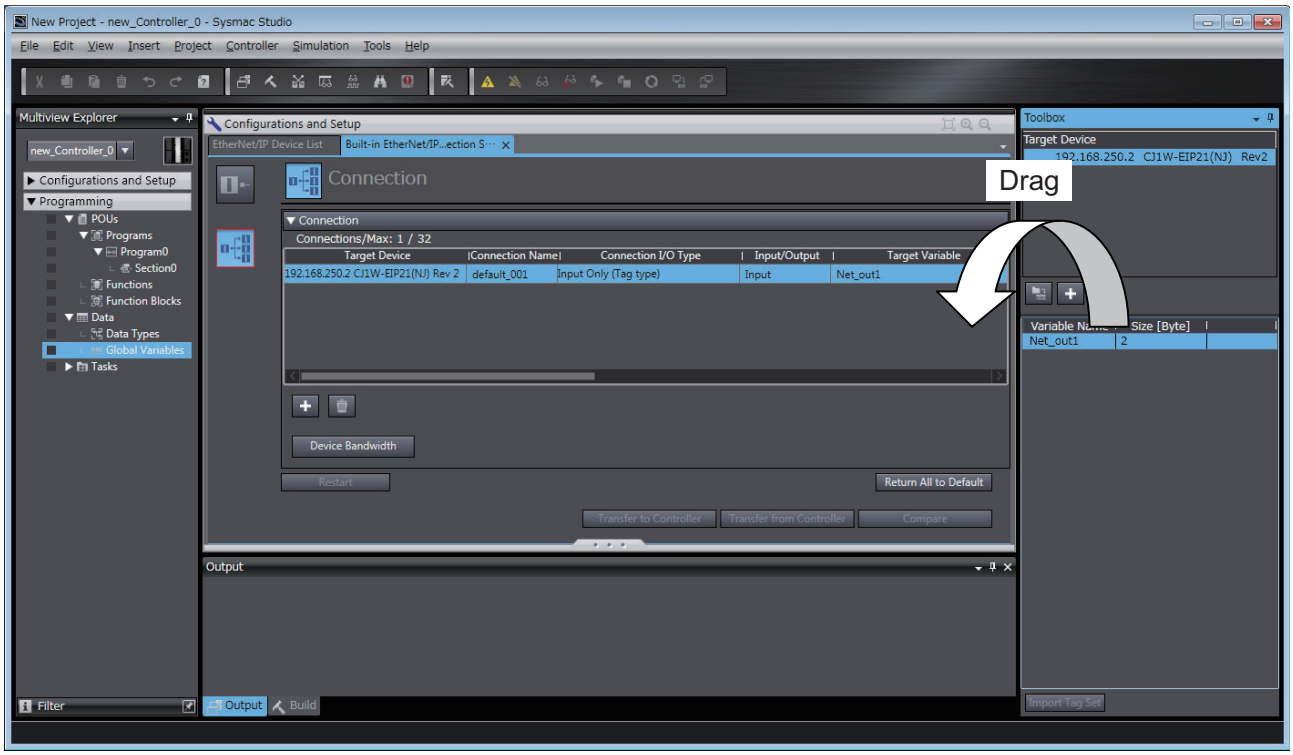
2 Right-click **Built-in EtherNet/IP Port Settings** for Controller 1 (originator device in this example), and select **Edit** from the menu.
The EtherNet/IP Connection Setting Tab Page is displayed.



3 Click the  (Show Connection Display) icon in the EtherNet/IP Connection Setting Tab Page.

4 Select **CJ1W-EIP21(NJ)** from **Target Device** in the Toolbox on the right of the tab page.
When you select **CJ1W-EIP21(NJ)**, the target device tag set (Net_Out1) for Controller 2 is displayed in the **Variable Name** column in the Toolbox.

5 Drag the target device tag set Net_Out1 from the **Variable Name** column in the Toolbox to the Connection List.
As you enter characters (or immediately after you press the Ctrl + Space Keys), a list of target device variables that can be set for the connection is provided. Select the value from the list.



6 Specify **Originator Variable** and its **Size [Byte]** for the tag set Net_Out1 added in step 5. Here, specify *Net_In1* for **Originator Variable** and *2* for its **Size [Byte]**.
Change the other settings as required.
You can set the following items in the connection settings.

Name	Setting Methods
Target Device	Select the target device.
Connection Name	Any name can be given to the connection (32 single-byte characters max.).
Connection I/O Type	Input Only (Tag type) is selected if the EtherNet/IP connection is established on a CS1W-EIP21, CJ1W-EIP21, CJ2B-EIP21, CJ2M-EIP21, CJ1W-EIP21(CJ2), CJ1W-EIP21(NJ), NX701, NX102-□□□□, NX1P2, NJ501-□□□□, NJ301-□□□□, or NJ101. When you create EtherNet/IP connection for another target device, select the connection I/O type specified in the device's EDS file. Use the **Input Only (ID type)** setting when the originator is a node from another manufacturer and does not support connection settings with a Tag type setting.
Input/Output	The connection's input/output is automatically displayed based on the selected connection. Input Only: Just Input is displayed.
Target Variable	Select the target node's tag set to assign it. • Input is specified for **Input/Output**: Select the target's output (produce) tag set. • Output is specified for **Input/Output**: Select the target's input (consume) tag set.
Size [Byte]	The data sizes of the target variables are displayed.

Name	Setting Methods
Originator Variable	Select the originator node's tag set to assign it. • Input is specified for **Input/Output**: Select the originator's input (consume) tag set. • Output is specified for **Input/Output**: Select the originator's output (produce) tag set.
Size [Byte]	Enter the data sizes of the originator variables.
Connection Type	Select whether the data is to be sent in the multicast or unicast (point-to-point) form. The default setting is multicast. • Multi-cast connection: Select when the same data is to be shared by multiple nodes. This setting is usually used. • Point-to-point connection: Select when the same data is not to be shared by multiple nodes. Since the data is sent in unicast transmission, other nodes are not burdened with unnecessary load. **Note** Refer to *6-1-4 Overview of Operation* on page 6-7 for details on how to use multi-cast and unicast connections, and how to count the number of connections.
RPI [ms]	Set the data update cycle (i.e., the packet interval) of each connection between the originator and target. The default setting is 50 ms (i.e., data is updated once every 50 ms).
Timeout Value	Set the time until a connection timeout is detected. The timeout value is set as a multiple of the packet interval (RPI) and can be set to 4, 8, 16, 32, 64, 128, 256, or 512 times the packet interval. The default setting is RPI x 4. The timeout value must be at least 10 ms.

7 The Toolbox displays the target devices if the devices are registered in the same Sysmac Studio project as where the originator devices are registered.
You can use one of the following methods to add unregistered devices in the same Sysmac Studio project as where the originator devices are registered to the Target Device List.

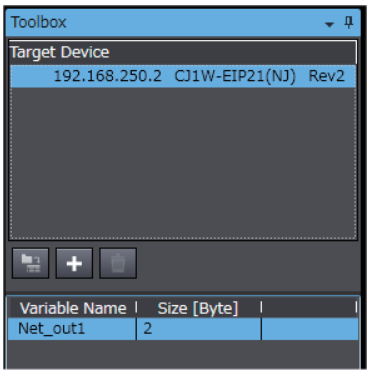
- Importing devices that are registered in another project
  You can import NJ/NX-series Controllers registered in another project data and add them to the Device List.
- Registering devices using user-specified settings
  You can manually add target devices to the device list.

Additional Information

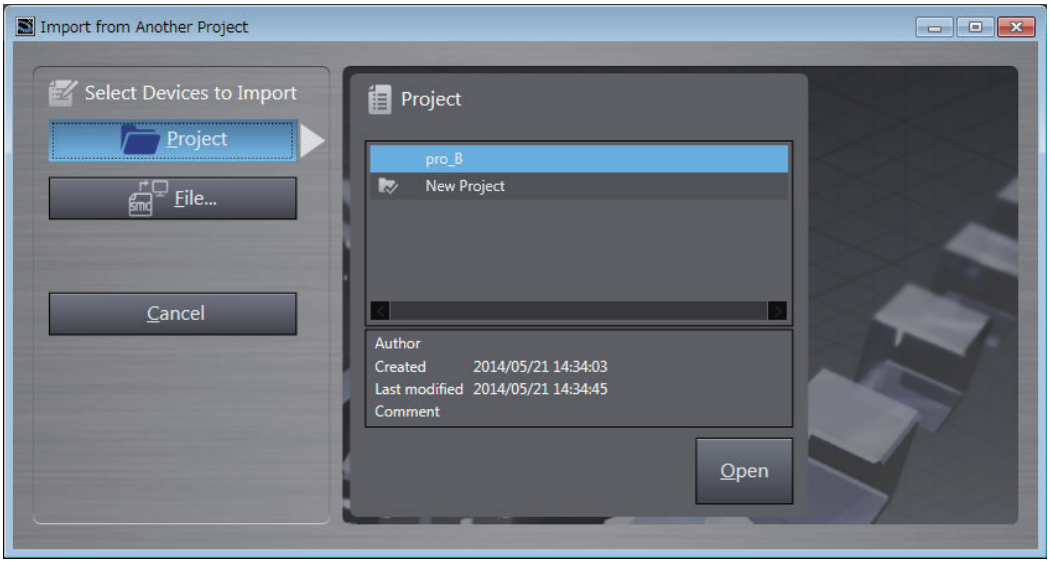
You can add target devices to the Device List by installing EDS files that include connection information for the devices in the Sysmac Studio and register the devices to the project.
Refer to *Adding EDS Files* on page A-20 for details.

8 Import devices that are registered in another project.

1) Click the  (Import a device from another project) Button in the Toolbox on the right of the EtherNet/IP Connection Setting Tab Page.



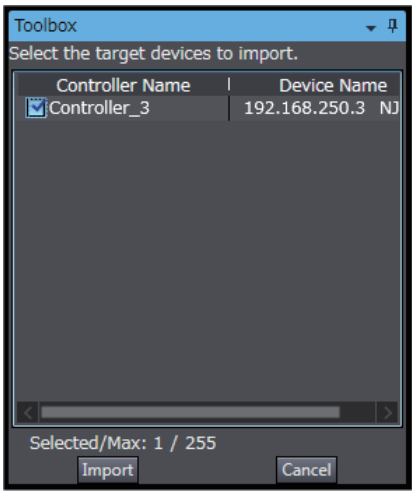
2) The Import from Another Project Dialog Box is displayed. Click the **Project** Button, select a project to import and click the **Open** Button.



3) The list of EtherNet/IP devices registered in the selected project will be displayed. Select the target devices to import, and click the **Import** Button.

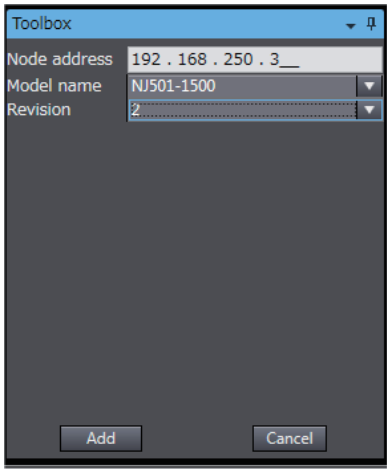
Note Only the project for which the EtherNet/IP connection settings are set will be displayed.

The imported EtherNet/IP devices are added to the Target Device List in the Toolbox.



9 Register devices as required.

1) Click the + Button under the Target Device List in the Toolbox.
   The Add Target Device Pane is displayed.
2) Enter relevant items for the target devices to add.



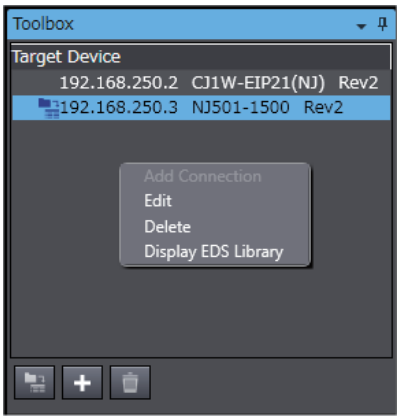
Menu	Description
Node address	Enter the target device IP address.
Model name	Select the target device model.
Revision	Select the revision of the target device.

3) Here, set the following items for Controller 3 and click the **Add** Button.
   The target device is added to the Target Device List in the Toolbox.
   Node address: 192.168.250.3
   Model name: NJ501-1500
   Revision: 2
4) You can click the **Import Tag Set** Button to import the tag sets that are set in the Network Configurator to the target devices.
   Select **To/From File - Export to File** in the **Tag Sets** Tab Page of the **Edit Device Parameters** Dialog Box, and generate CSV files to import.

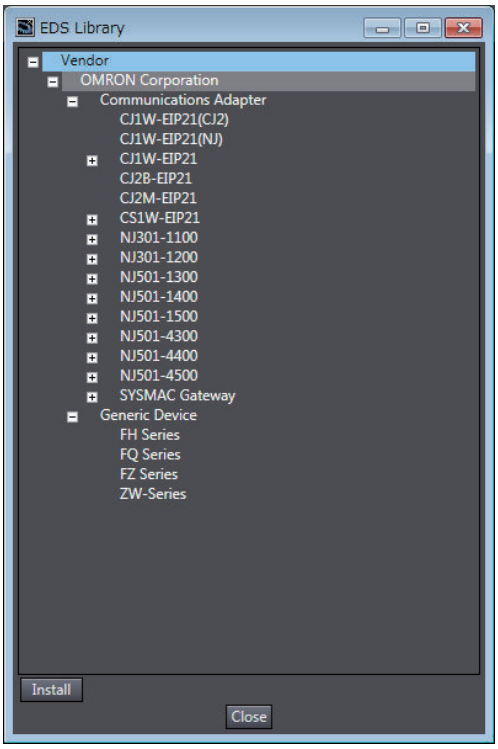
● Adding EDS Files

Note The Modular EDS device is supported by the Sysmac Studio version 1.11 or higher.

1 Right-click anywhere in the Target Device List in the Toolbox of the EtherNet/IP Connection Setting Tab Page and select **Display EDS Library** from the menu.



2 The EDS Library Dialog Box is displayed. Click the **Install** Button.



3 Select the EDS file to add, and then click the **Open** Button.
The EDS file is added.

4 The EtherNet/IP device with the EDS file installed is added to the EDS Library.
Devices listed in the EDS Library are used as a candidate device list when adding devices to the Target Device List in the Toolbox of the EtherNet/IP Connection Setting Tab Page.

Changing the Target Device Settings after Making Connection Settings

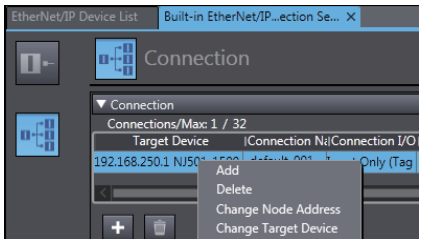
If you change the IP address, model, or revision of the target device after making the connection settings, you must also change the target device settings that are included in the connection settings. You can change the target device settings entirely.

Precautions for Correct Use

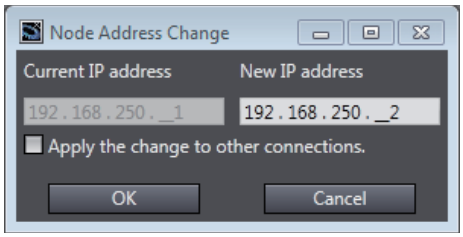
When you use the Sysmac Studio version 1.10 or lower, create the connections again if you changed the target device after configuring the connection settings.

● Changing the IP Addresses for All Target Devices

1 Right-click one of the connection lines and select **Change Node Address** from the menu.



2 The **Node Address Change** Dialog Box is displayed. Enter a new IP address in **New IP address**.
To apply the same change to other connections, select the **Apply the change to other connections** Check Box.



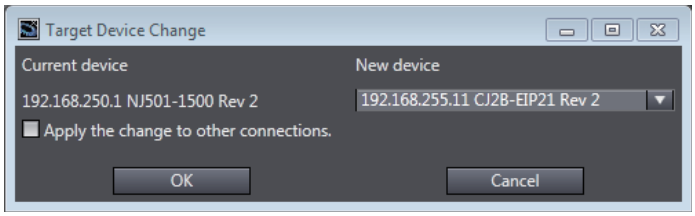
3 To apply the same change to other connections, select the **Apply the change to other connections** Check Box.

4 Click the **OK** Button.

● Changing All Target Device Information including Model Names and Revisions

1 Right-click one of the connection lines and select **Change Target Device** from the menu.

2 The **Target Device Change** Dialog Box is displayed. Select a target device from **New device**.



Precautions for Correct Use

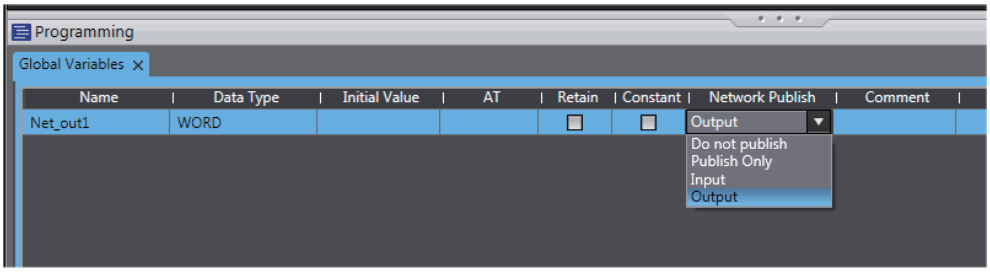
- Changeable target devices are limited to ones that have "OMRON" in the Vendor ID and is an EDS device of the Communications Adapter in the Device Type.
- To display a device in the list of selectable new target devices, the device must be registered as the target device in the Toolbox.

3 To apply the same change to other connections, select the **Apply the change to other connections** Check Box.

4 Click the **OK** Button.

Registering the Network Variable for the Target Device

1 Assign the network variable to the tag used for the EtherNet/IP connection for Controller 2 (target device).
This network variable stores data to send to Controller 1 (originator device).
Set the **Network Publish** attribute to **Input** or **Output** in the Global Variable Table for the variable so that the variable serves as a network variable, i.e., the variable can be used for the EtherNet/IP connections. In this example, set the network variable for Controller 1 as shown below.



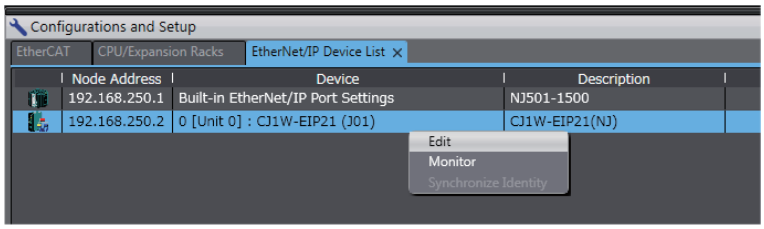
- Name: Net_Out1
- Data type: WORD
- Network Publish attribute: Output

Registering the Tag and Tag Set for the Target Device

Set the tag and tag set for the target device.

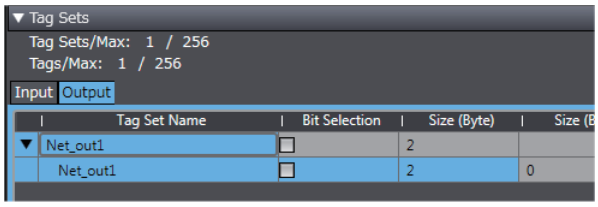
1 Select **EtherNet/IP Connection Settings** from the Tools Menu.
The EtherNet/IP Device List Tab Page is displayed.

2 Right-click **CJ1W-EIP21**, the EtherNet/IP Unit connected to the Controller 2 (originator device in this example), and select **Edit** from the menu.
The EtherNet/IP Connection Setting Tab Page is displayed.



3 Click the  (Show Tag Set Display) icon in the EtherNet/IP Connection Setting Tab Page.

4 Click the **Output** tab to switch to the **Output** Tab Page. Register the following tag and tag set. The tag and tag set can be registered in the same way as for the target device. (Refer to *Registering the Tag and Tag Set* on page A-12.)



Checking the Device Bandwidth Usage

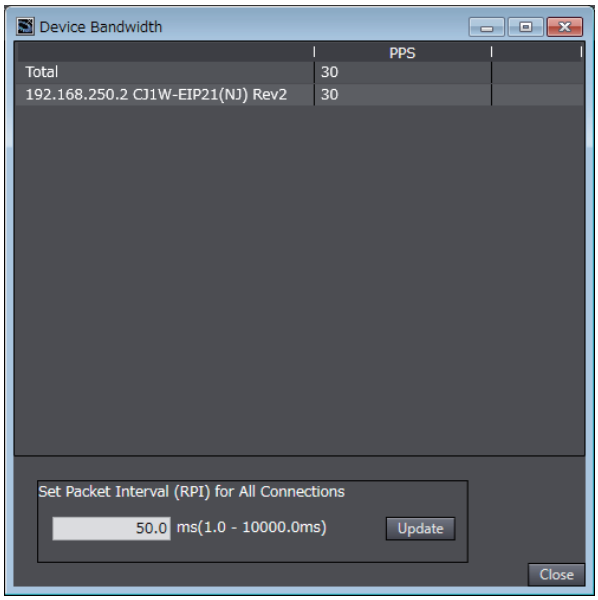
The bandwidth usage for the device can be displayed from the EtherNet/IP Connection Setting Tab Page.
This value is for when multicast filtering is used.

Precautions for Correct Use

In the Device Bandwidth Dialog Box, you can only check the bandwidth being used for the EtherNet/IP connections from one originator device to its target devices.
The actual bandwidth used for the EtherNet/IP network must be calculated by taking into account of all bandwidths used on the EtherNet/IP network (i.e., bandwidths used for connections for the other devices in the EtherNet/IP network than the one given on the dialog box must be included into the calculation).

● Procedure

Click the **Device Bandwidth** Button in the EtherNet/IP Connection Setting Tab Page for the target device.



Menu	Description
PPS	Gives the bandwidth used for each target device and total bandwidth used for all target devices.
Set Packet Interval (RPI) for All Connections	Changes all Packet Interval (RPI) values for all target devices.

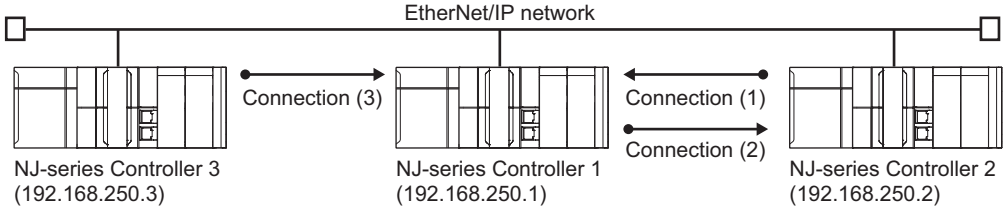
Additional Information

You can specify a value in **Set Packet Interval (RPI) for All Connections** and click the **Update** Button to change packet interval (RPI) values set in the connection settings for all target devices to the specified value.

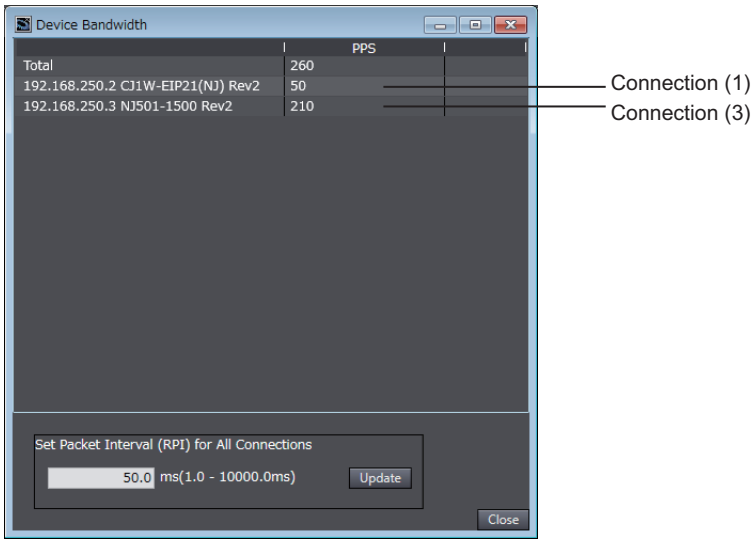
● Calculation Example for Bandwidth Used (PPS) for Each Device by the EtherNet/IP Connections

Establishing following three EtherNet/IP connections between Controllers (1) to (3) in the EtherNet/IP network

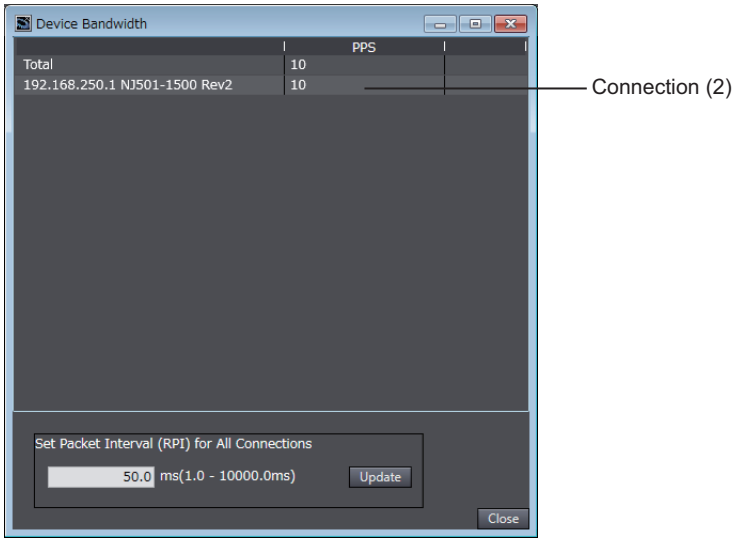
Connection type	Relevant devices in the EtherNet/IP connections	Device bandwidth usage (PPS)
Connection (1)	NJ-series Controller 2 (target device) to NJ-series Controller 1 (originator device)	50 pps
Connection (2)	NJ-series Controller 1 (target device) to NJ-series Controller 2 (originator device)	10 pps
Connection (3)	NJ-series Controller 3 (target device) to NJ-series Controller 1 (originator device)	210 pps



Bandwidth used (PPS) for each EtherNet/IP device is as given below.



EtherNet/IP connection settings for Controller 1



EtherNet/IP connection settings for Controller 2

In this example, the PPS for Connection (1) is 50 pps, the PPS for Connection (2) is 10 pps, and the PPS for Connection (3) is 210 pps. Therefore, bandwidth used (PPS) for each EtherNet/IP device is as given below.
192.168.250.1: 270 pps = 50 pps (for Connection (1)) + 10 pps (for Connection (2)) + 210 pps (for Connection (3))
192.168.250.2: 60 pps = 50 pps (for Connection (1)) + 10 pps (for Connection (2))
192.168.250.3: 210 pps = 210 pps (for Connection (3))

● Adjusting Method

If the calculation result value exceeds the values in the specifications of the devices used in the EtherNet/IP connections, re-evaluate the overall network configuration and correct it by taking steps such as selecting a different Ethernet switch or splitting the network.
If the RPI is made longer, the PPS for the EtherNet/IP connections will decrease.
You can change the RPI values in the connection settings for all the target devices by specifying a value in **Set Packet Interval (RPI) for All Connections** in this dialog box.

Refer to *14-2-2 Tag Data Link Bandwidth Usage and RPI* on page 14-9 for the relationship between the PPS for the device and the RPI.

Transferring the Connection Settings Data

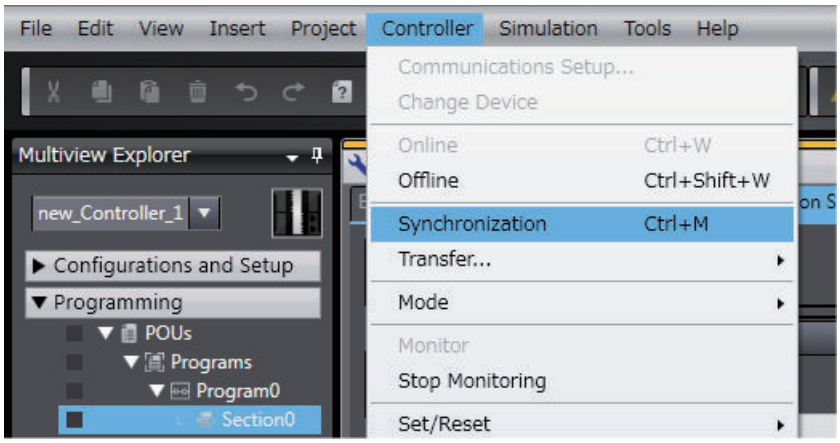
You can synchronize and transfer EtherNet/IP connection settings along with the program data. You can also transfer all the EtherNet/IP connection settings along with the program data.

Precautions for Correct Use

- If the node addresses (IP addresses) are not set correctly, you may connect to the wrong Controller and set incorrect device parameters. Download data only after you confirm that you are connected to the correct Controller.
- If incorrect connection settings are set, it may cause equipment to operate unpredictably. Even when the correct connection settings are set, make sure that there will be no effect on equipment before you transfer the data.
- A connection error will result if the network variables that are used in the tag settings are not set in the Controller. Before downloading the connection settings, check to confirm that the network variables used in the tag settings are set in the Controller.
- If a communications error occurs, the output status depends on the specifications of the device being used. When a communications error occurs for a device that is used along with output devices, check the operating specifications and implement safety countermeasures.
- The built-in EtherNet/IP port and the port on the EtherNet/IP Unit are automatically restarted after the parameters are downloaded. This restart is required to enable the tag set and connection information. Before you download the parameters, check to confirm that problems will not occur with the equipment when the port is restarted.
- Do not disconnect the Ethernet cable or reset or turn OFF the power to the EtherNet/IP Unit during the parameter download.
- The EtherNet/IP connections between relevant nodes is stopped during a download. Before you download data in RUN mode, make sure that it will not affect the controlled system. Also implement interlocks on data processing in ladder programming that uses EtherNet/IP connections when the connections are stopped or a connection error occurs.
- In the EtherNet/IP network, if the device bandwidth usage (PPS) exceeds the unit's allowable bandwidth (PPS), the EtherNet/IP connection operations may not agree with the settings. If you increase the RPI value in such a case, there are cases when the problem can be resolved (i.e., the operations agree the settings).

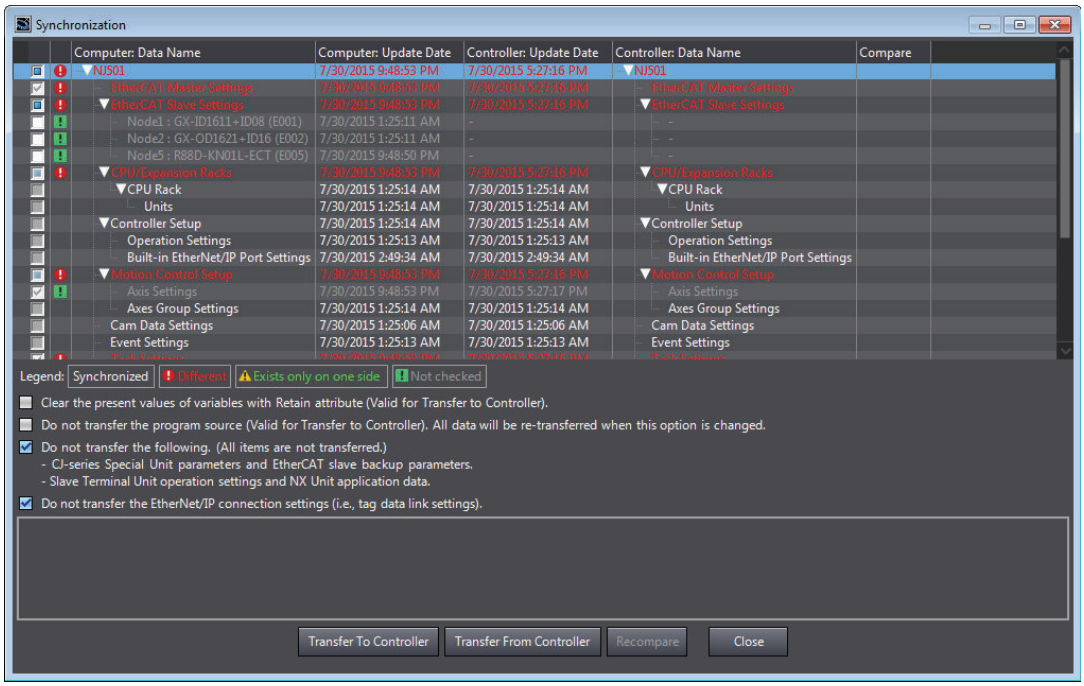
● Synchronizing/Transferring a Whole Project

1 Establish an online connection between the computer and the Controller, and then select **Synchronization** from the **Controller** Menu. (Or, click the  Button on the Toolbal.)



The Synchronization Window is displayed, and comparison of the user program and parameter settings between the Sysmac Studio and the Controller is started.

2 The following Uploading and Downloading Data Window is displayed after the automatic comparison.



3 Clear the **Do not transfer the EtherNet/IP connection settings (i.e., tag data link settings)** Check Box and then click the **Transfer To Controller** Button.
Then the EtherNet/IP connection settings are transferred along with the not-synchronized data.
If no EtherNet/IP connection settings are set in the Sysmac Studio, no data will be sent.

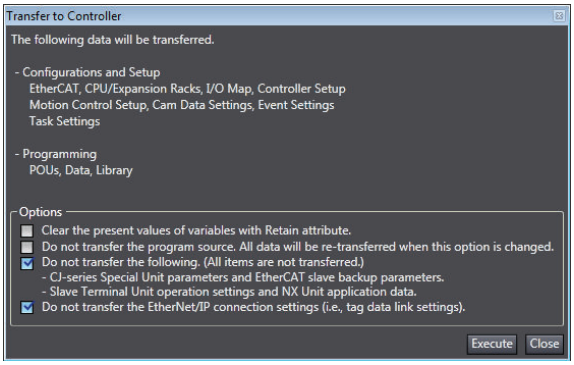
● Transferring all data

1 Establish an online connection between the computer and the Controller and then select **Transfer** - **To Controller** from the **Controller** Menu. (Or, click the  Button on the Toolbar.)

2 The **Transfer to Controller** Dialog Box is displayed.
Clear the selection of the **Do not transfer the EtherNet/IP connection settings (i.e., tag data link settings)** Check Box, and then click the **Execute** Button.

Precautions for Correct Use

To transfer only the connection settings, execute Transfer from the EtherNet/IP Connection Setting Tab Page.
Even if you clear the **Do not transfer the connection setting** Check Box, the connection settings are not transferred from the Synchronization Window, the **Transfer to Controller** Dialog Box, or the **Transfer from Controller** Dialog Box as long as the data in the computer is synchronized with the data in the Controller.



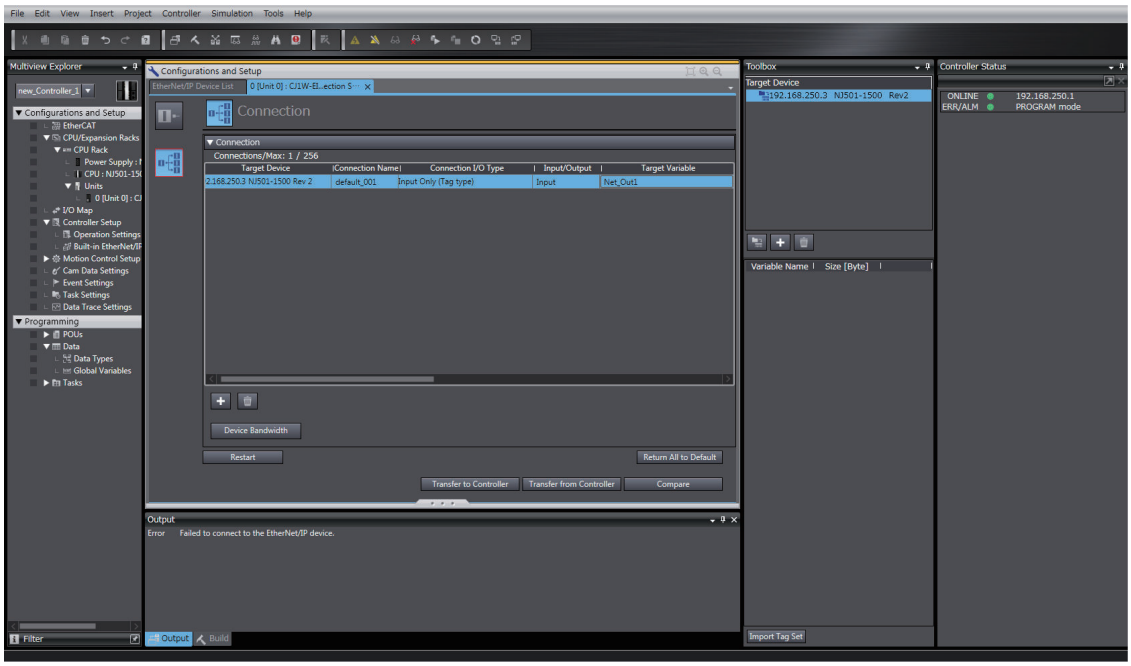
● Transferring Only the EtherNet/IP Connection Settings

You can transfer tag sets and connections to the EtherNet/IP devices.

1 Establish an online connection with the Controller.

2 Click the **Transfer to Controller** or **Transfer from Controller** Button in the EtherNet/IP Connection Setting Tab Page.
The tag settings and connection settings set at that time are transferred to the Controller connected online.

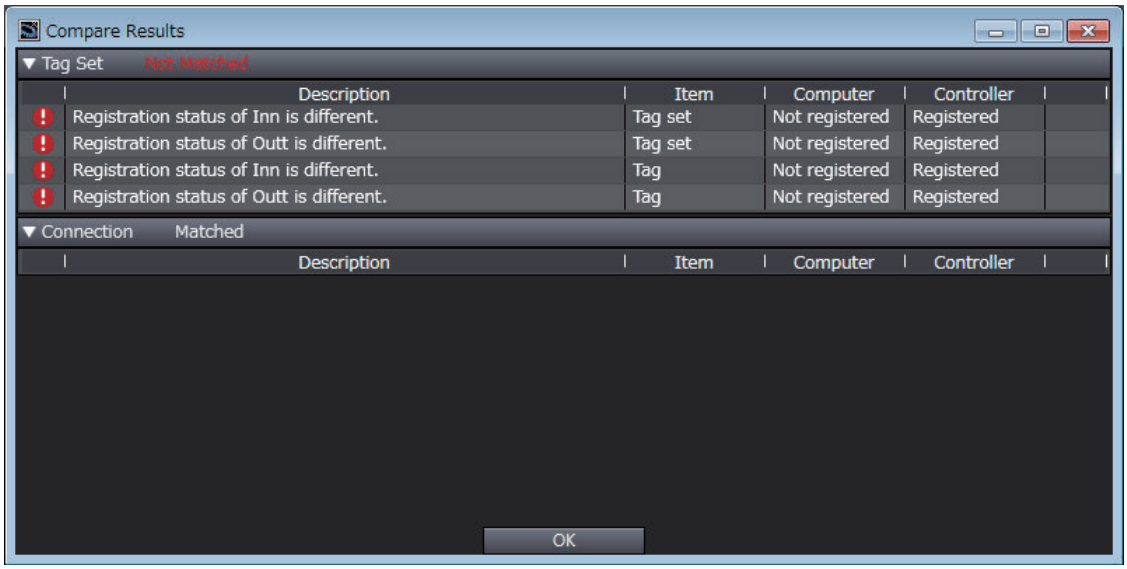
3 If the Controller connected online is in RUN mode, the dialog box to confirm whether to switch to PROGRAM mode before transferring the settings is displayed.



● Comparison

The differences in the tag set and connection settings between the project and the EtherNet/IP devices can be displayed.

1 Click the **Compare** Button in the EtherNet/IP Connection Setting Tab Page.



Starting and Stopping EtherNet/IP Connections

● Automatically Starting EtherNet/IP Connections

The EtherNet/IP device is automatically restarted and EtherNet/IP connections are automatically started immediately after the connection settings are downloaded from the Sysmac Studio.

Precautions for Correct Use

Connections are adversely cut off if any of the following errors occurs in the CPU Unit that is the originator while EtherNet/IP connections are active.

- Major fault level Controller error
- Partial fault level Controller error

● Starting and Stopping the EtherNet/IP Connections for the Entire Network

You can start and stop EtherNet/IP connections from the user program or from the Sysmac Studio.

Precautions for Correct Use

Use the same method (i.e., either the user program or the tool software) to both start and stop EtherNet/IP connections.
For example, if you use the *_EIP_TDLinkStopCmd* (Tag Data Link Communications Stop Switch) system-defined variable to stop EtherNet/IP connections, you cannot start them from the Sysmac Studio and the Network Configurator.

A-2-5 Checking Communications Status with the Sysmac Studio and Troubleshooting

You can monitor the communications status of the EtherNet/IP connections after their settings are set. You can also check errors.

Precautions for Correct Use

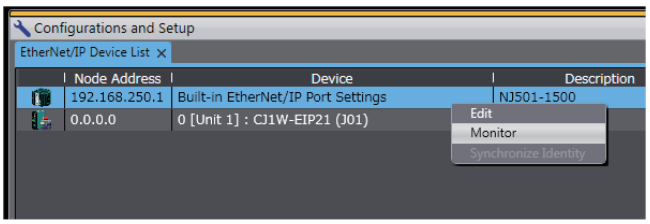
Make sure that the connection settings in both the Sysmac Studio and the Controller are consistent before using the monitor functions. You can use the *Comparison* on page A-30 to see if they are the same.

Checking Communications Status with the Sysmac Studio

You can check the communications status on the EtherNet/IP connections in the EtherNet/IP Connection Monitor Tab Page.

1. Select **EtherNet/IP Connection Settings** from the **Tools** Menu to display the EtherNet/IP Device List Tab Page.

2. Right-click the Controller for which you want to check the communications status, and select **Monitor** from the menu.
   The EtherNet/IP Connection Monitor Tab Page is displayed. In the EtherNet/IP Connection Monitor Tab Page, each communications status is displayed in six tabs.

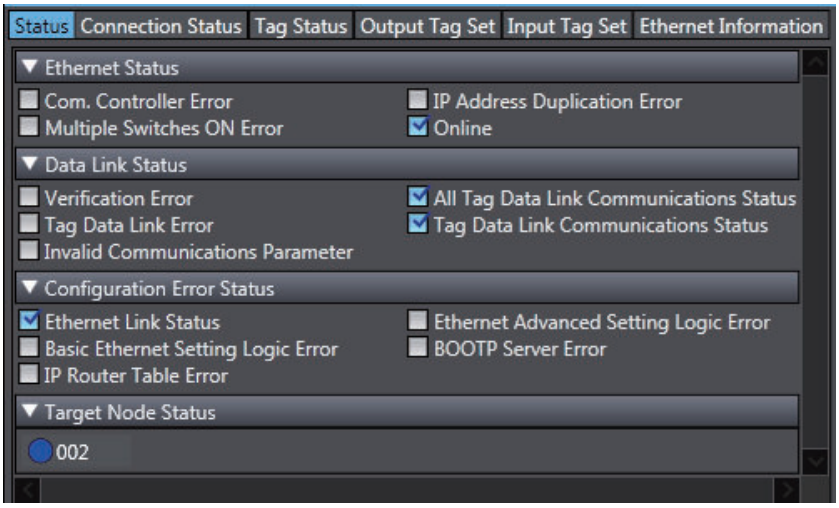


3 Select one of the six tabs for which you want to confirm the communications status.

- **Status** Tab Page

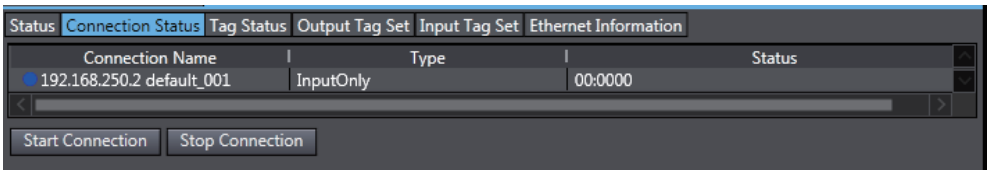
  This tab page gives the TRUE/FALSE status of the system-defined variables that monitors the tag data link status and communication status for errors. If any of the variables is TRUE, its checkbox is marked with ☑.

  Refer to *15-2-1 The Network Configurator's Device Monitor Function* on page 15-3 for details on each status item.



- **Connection Status** Tab Page

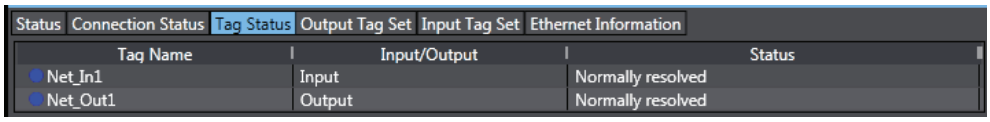
  Current status of each connection is given.



Name	Description
Connection Name	Gives the current status of each connection with the following text colors. Blue: Normal Red: There is at least one connection that has not been established. Gray: There are no connections or the connection operation is stopped.
Type	Gives the connection type.
Status	Gives the current status on each connection with codes. • Normal operation: 00:0000 • Abnormal operation: Gives an error code. This information can be used to identify the cause of EtherNet/IP connection errors. Refer to *15-2-2 Connection Status Codes and Troubleshooting* on page 15-11 for details on the connection status.

- **Tag Status** Tab Page

  This tab page gives if the tag settings for each tag for EtherNet/IP connections are set so that data can be exchanged with target devices.



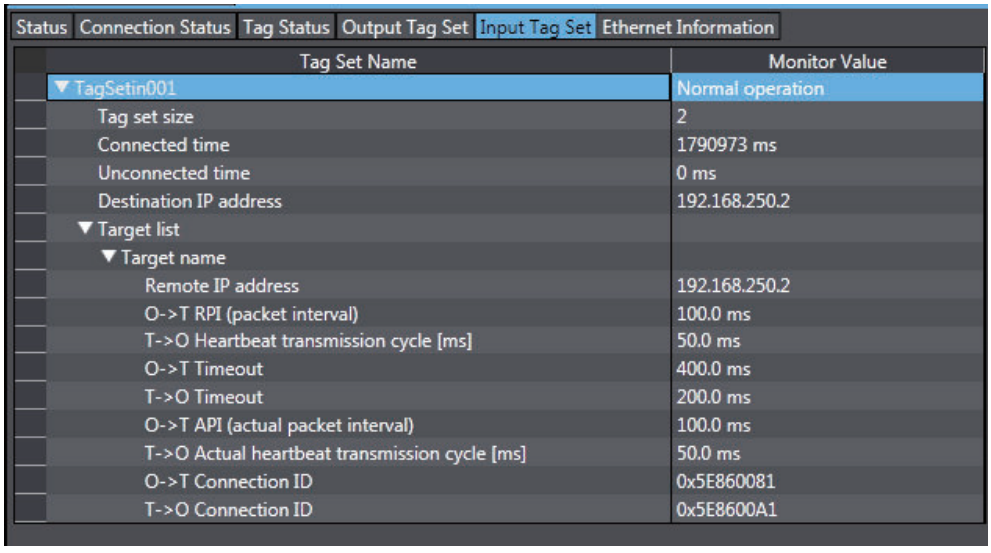
Name	Description
Tag Name	The current status of each tag is indicated by its color. Red: Tag name resolution error Blue: Tag name resolution normal Gray: Not yet transferred (no information in device).
Input/Output	Gives the type of the tag.
Status	The following status is displayed depending on the status that is set. • Normally resolved: Normal data exchange is possible. • Different sizes: Different sizes are set for the network variables and the tag settings. A connection will not be established for a tag for which this error occurs. • No tag: A network variable is not set in the variable table in the CPU Unit for the specified tag setting. Or, instead of a member of union variable, unions are specified. A connection will not be established for a tag for which this error occurs. • Attribute error: The following two factors cause this error. 1. Writing is not possible for constant attributes. 2. The I/O direction that is set in the tag data link settings does not agree with the I/O direction of the variable in the CPU Unit. There is an error in the setting of a Network Publish attribute for a CPU Unit variable. A connection will not be established for a tag for which this error occurs.

- **Output Tag Set** and **Input Tag Set** Tab Pages

  You can monitor the status of each input/output tag set that is used for the EtherNet/IP connections.

  Note The tag set status monitor is not available for a built-in EtherNet/IP port on NJ-series Controller version 1.08 or earlier.

  Click ▼ of each tag to display its detailed information.



Name	Description
Tag Set Name	Gives the connection status. If there is a connection error, "Not connected or error" is given.

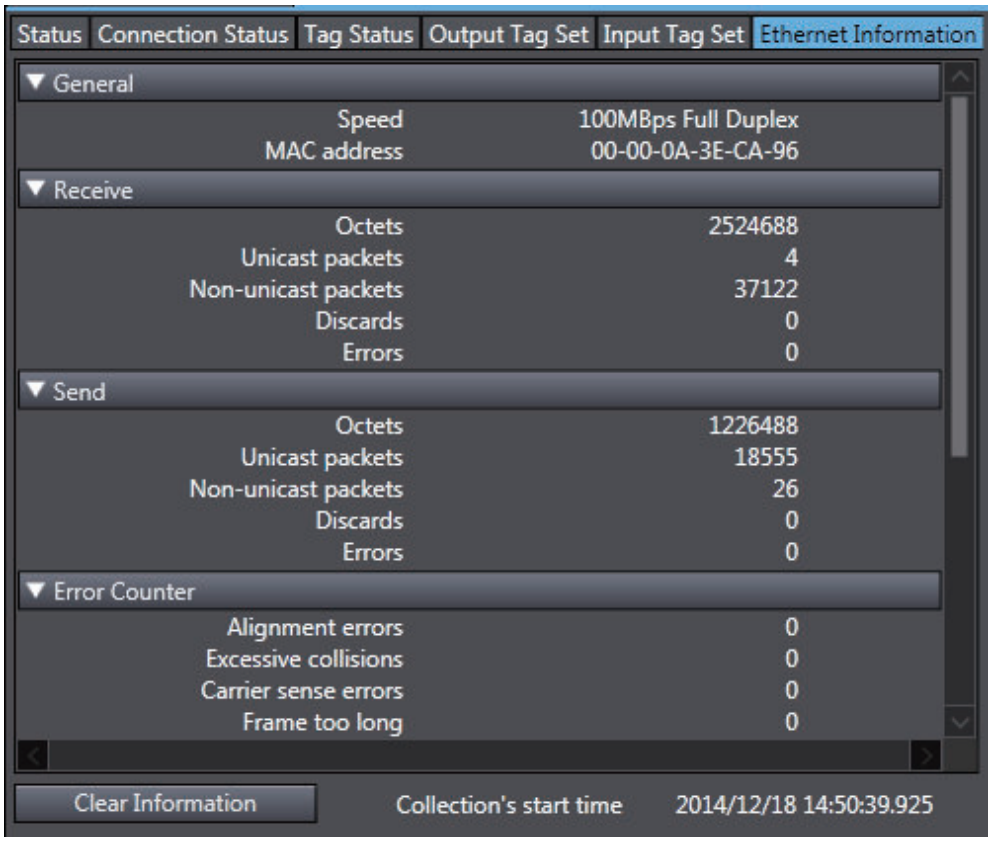
Name	Description
Tag set size	Gives the size of the tag set in bytes.
Connected time	Gives the total connection duration in milliseconds.
Unconnected time	Gives the total disconnection duration in milliseconds.
Number of connections (in the **Output Tag Set** Tab Page)	Gives the number of connections.
Number of connected originators (in the **Output Tag Set** Tab Page)	Gives the number of the connected originator devices.
Originator list (in the **Output Tag Set** Tab Page), Target list (in the **Input Tag Set** Tab Page)	Gives the detailed information of the connected originators.
Originator name (in the **Output Tag Set** Tab Page), or Produced tag name (in the **Input Tag Set** Tab Page)	Gives no information.
IP address (in the **Output Tag Set** Tab Page), or Remote IP address (in the **Input Tag Set** Tab Page)	Gives the IP addresses allocated for the originators.
Connected time (in the **Output Tag Set** Tab Page)	Gives the total duration of connection with the originator in milliseconds.
Unconnected time (in the **Output Tag Set** Tab Page)	Gives the total duration of disconnection with the originator in milliseconds.
Destination IP address (in the **Output Tag Set** Tab Page)	Gives the destination IP addresses. If the multi-cast connections are used, its own multi-cast address is displayed.
O->T RPI (packet interval)	Gives the RPI of connection from the originator to the target in milliseconds.
T->O Heartbeat transmission cycle (ms)	Gives the heartbeat transmission period of the connections from the target to the originator in milliseconds.
O->T Timeout	Gives the timeout time for the connections from the originator to the target in milliseconds.
T->O Timeout	Gives the timeout time for the connections from the target to the originator in milliseconds.
O -> T API (actual packet interval)	Gives the RPI of connection from the originator to the target in milliseconds.
T->O Actual heartbeat transmission cycle (ms)	Gives the actual heartbeat transmission period of the connections from the target to the originator in milliseconds.
O->T Connection ID	Gives the connection identification for the connections from the originator to the target in hexadecimal.
T->O Connection ID	Gives the connection identification for the connections from the target to the originator in hexadecimal.

- **Ethernet Information** Tab Page

  This tab page displays the communications status at the communications driver level of the built-in Ethernet/IP port.

  The error counter information can be used to confirm whether communications problems have occurred.

  Under the Tag Data Link, you can confirm characteristics such as the bandwidth usage (PPS).



Display example for an NJ-series CPU Unit

Display example for an NJ-series CPU Unit
With an NX701 CPU Unit, the status for each port is displayed.

A-2-6 Troubleshooting

In the case that there is a setting error or a communications error in the EtherNet/IP networks, the Sysmac Studio displays the error in the Troubleshooting Dialog Box.
Refer to the *NJ/NX-series Troubleshooting Manual (Cat. No. W503)* for how to identify errors and details on errors.

Troubleshooting When Transferring and Monitoring the EtherNet/IP Connection Settings Fail with Sysmac Studio Version 1.10 or Higher

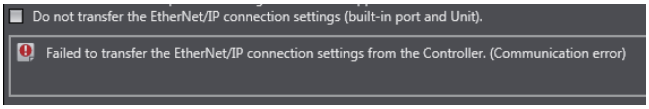
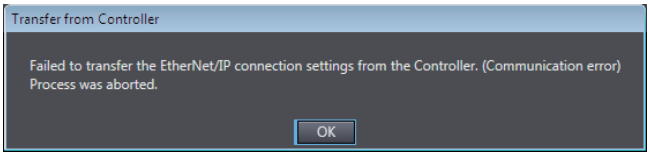
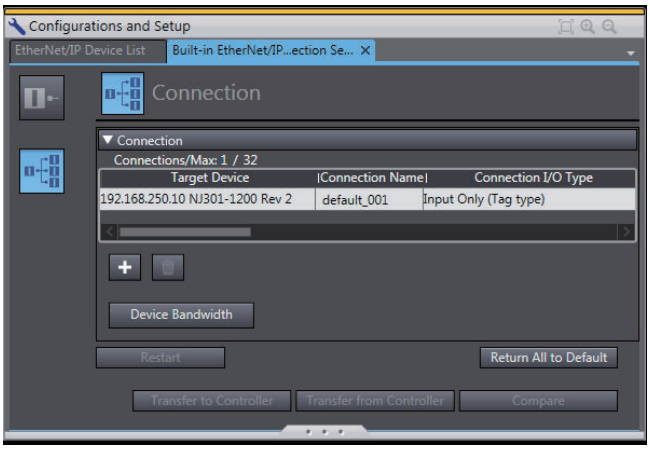
The first time you establish an online connection between the Controller and the computer that runs the Sysmac Studio version 1.10 or higher with Windows Firewall on the computer enabled, a dialog box may be displayed to confirm the connection. If that occurs, make the following selection in the dialog box.

- **Unblock** (on Windows XP/Vista)
- **Allow access** (on Windows 7 or higher)

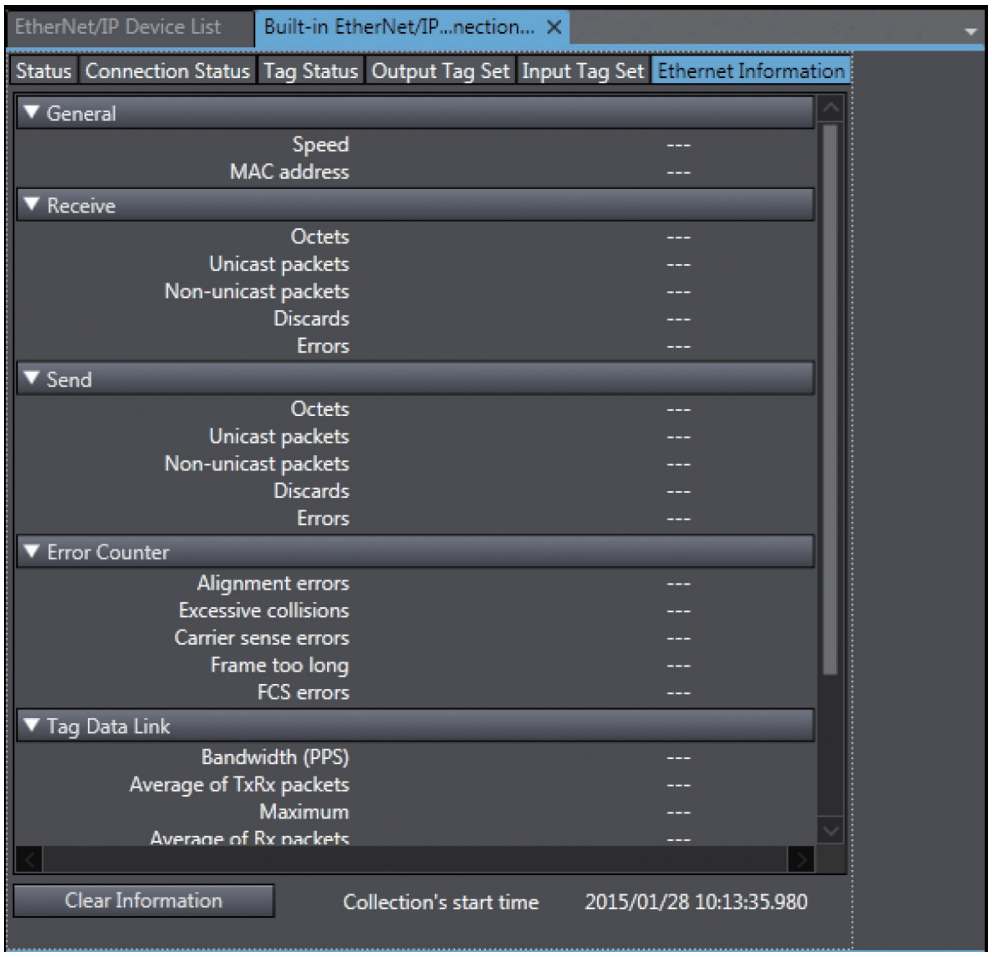
If you make other settings than above, transferring and monitoring the EtherNet/IP connection settings may not be properly performed even if the online connection is successfully established between the Sysmac Studio version 1.10 or higher and the Controller.
If the above problem occurs, take the following corrective method 1 or 2.

● Problems

- The connection setting data cannot be transferred.

Data Transmission Screen	Problem
Synchronization Window	The Sysmac Studio displays the following error message and the data will not be transferred. 
Transfer to Controller Dialog Box	The Sysmac Studio displays the following error dialog box and the data will not be transferred. 
EtherNet/IP Connection Setting Tab Page	The **Transfer to Controller** and **Transfer from Controller** Buttons are grayed out and the data cannot be transferred/compared. 

- Monitoring cannot be performed
  Monitor results in the EtherNet/IP Connection Monitor Tab Page remain as "---".



Method 1: Disabling Windows Firewall Settings

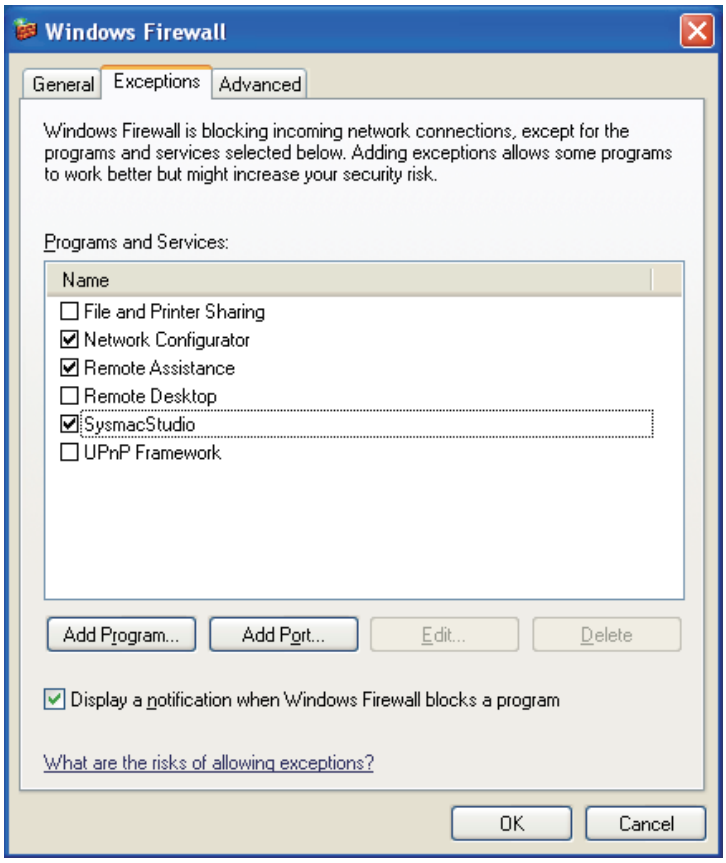
Precautions for Correct Use

The main function of the firewall is to prevent unwanted access from external sources (e.g., the Internet).
The changes that are made with the following procedures are to allow the Sysmac Studio and the NJ/NX-series Controller to connect. If your computer is on an inhouse network, make sure that security will not be jeopardized before you change the settings.

● Windows XP

1 Open the **Control Panel** from the **Windows Start Menu** and then select **Windows Firewall** icon.
The **Windows Firewall** Dialog Box is displayed.

2 Click on the **Exceptions** tab and select **Sysmac Studio** in the **Programs and Services** list.

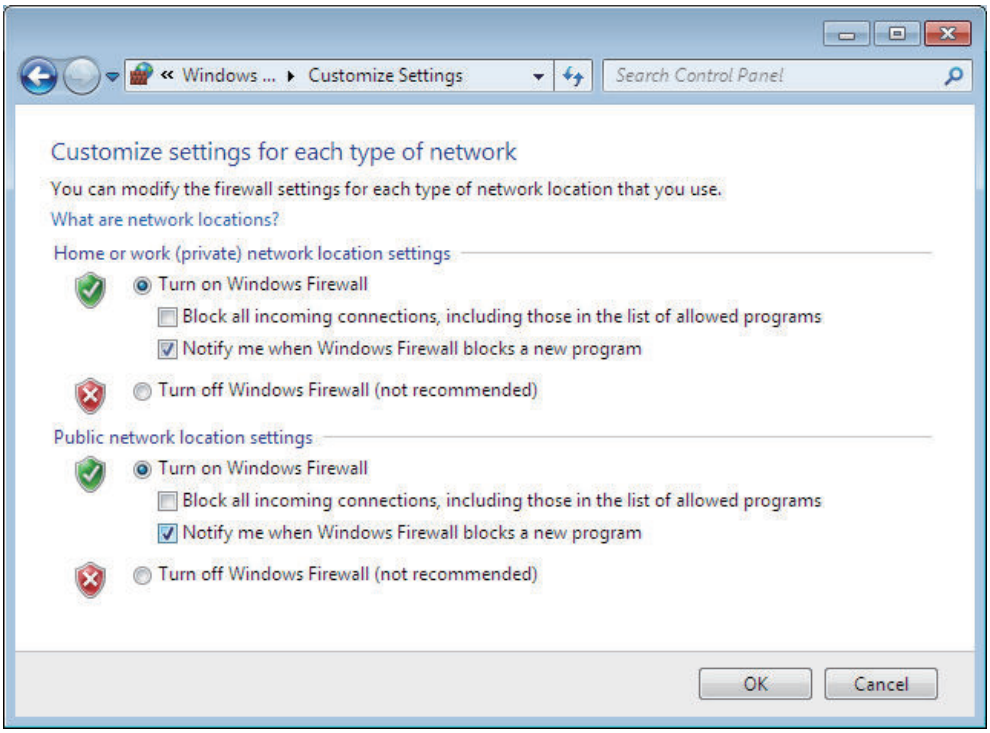


● Windows Vista, Windows 7, or later version

1 Open the **Control Panel** from the **Windows Start Menu** and then select **Windows Firewall** icon.
The **Windows Firewall** Dialog Box is displayed.

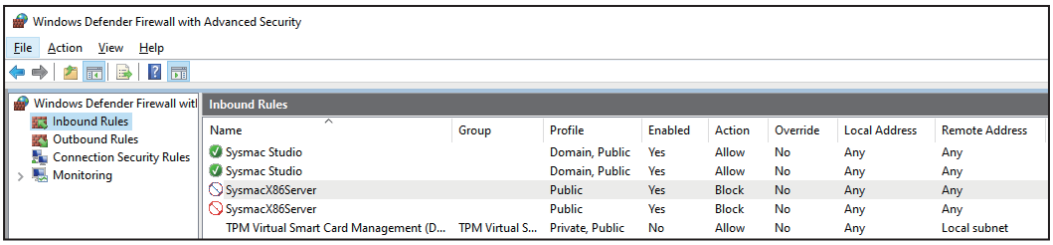
2 Select **Turn Windows Firewall On or Off**.
The **Customize Settings** Dialog box is displayed.

3 Clear the **Block all incoming connections, including those in the list of allowed programs** Check Box and click the **OK** Button.



4 Select the **Advanced** Tab in the Windows Firewall Dialog Box.
The **Windows Firewall with Advanced Security** Dialog Box is displayed.

5 Click **Inbound Rules** in the left pane and then double-click **SysmacX86Server** in the **Inbound Rules** list for Sysmac Studio Ver.1.31 or later. For Sysmac Studio earlier than Ver.1.31, double-click **Sysmac Studio**.
If you double-click **SysmacX86Server**, **SysmacX86Server Properties** window appears. If you double-click **Sysmac Studio**, **Sysmac Studio Properties** window is displayed.



6 For Sysmac Studio Ver.1.31 or later, make the following settings in the **SyamacX86Server Properties** window. If Sysmac Studio version is earlier than Ver. 1.31, make the following settings in the **Syamac Studio Properties** window.

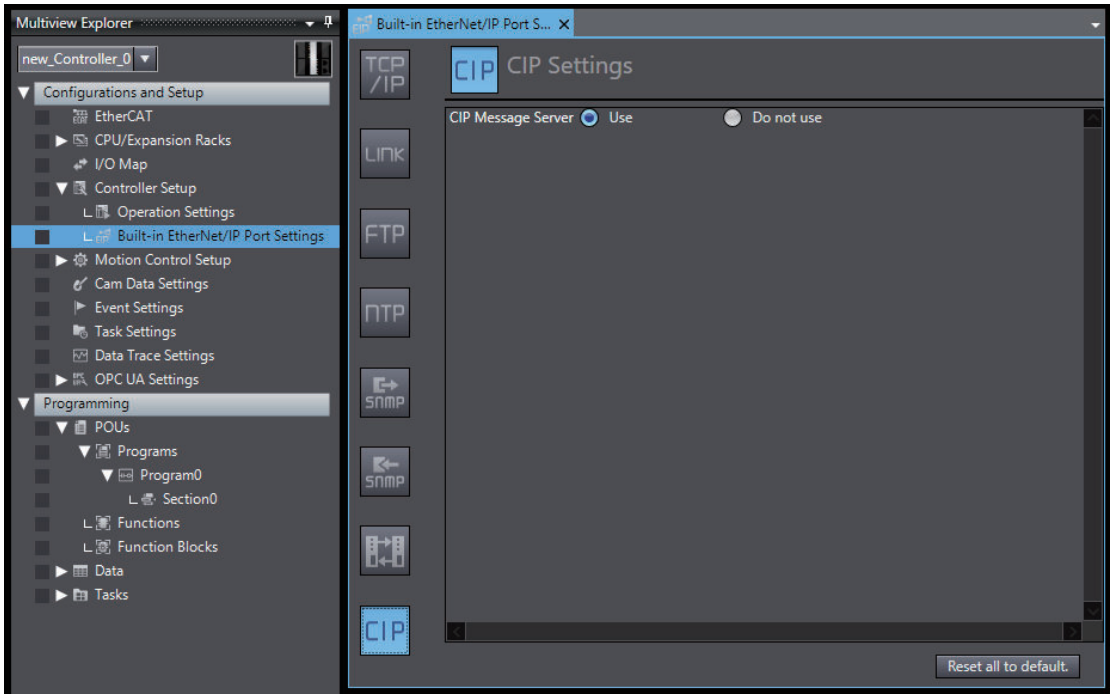
- If the **Public** Check Box under **Profiles** is not selected in the **Advanced** Tab Page, select it.
- If the **Enabled** under **General** is not selected in the **General** Tab Page, select it.
- Select **Allow the connection** under **Action** in the **General** Tab Page.

Method 2: Selecting the Use Option for the CIP Message Server

1 Connect the Sysmac Studio to the Controller.

2 Select **Configurations and Setup** - **Controller Setup** - **Built-in EtherNet/IP Port Settings** - **CIP Settings**.

3 Change the setting to select the **Use** Option for **CIP Message Server**.



Method 3: Configuring Packet Filter Settings to Allow Packets Used by Sysmac Studio's EtherNet/IP Connection Settings

1 Connect the Sysmac Studio to the Controllers.

2 Select **Configurations and Setup** - **Controller Setup** - **Built-in EtherNet/IP Port Settings** - **TCP/IP Settings**.

3 Enter the settings for **Packet Filter** to allow packets used by Sysmac Studio's EtherNet/IP connection settings. Refer to *Packet Filter* on page 4-7 for detailed settings.

Method 4: Cycling the Power Supply to the Controller

Cycle the power supply to the NJ/NX-series Controller and transfer/monitor the EtherNet/IP connections settings again.

Note You may need to cycle the power supply when reflecting the changes in the IP address of the built-in EtherNet/IP port or executing Transfer to the Controller.

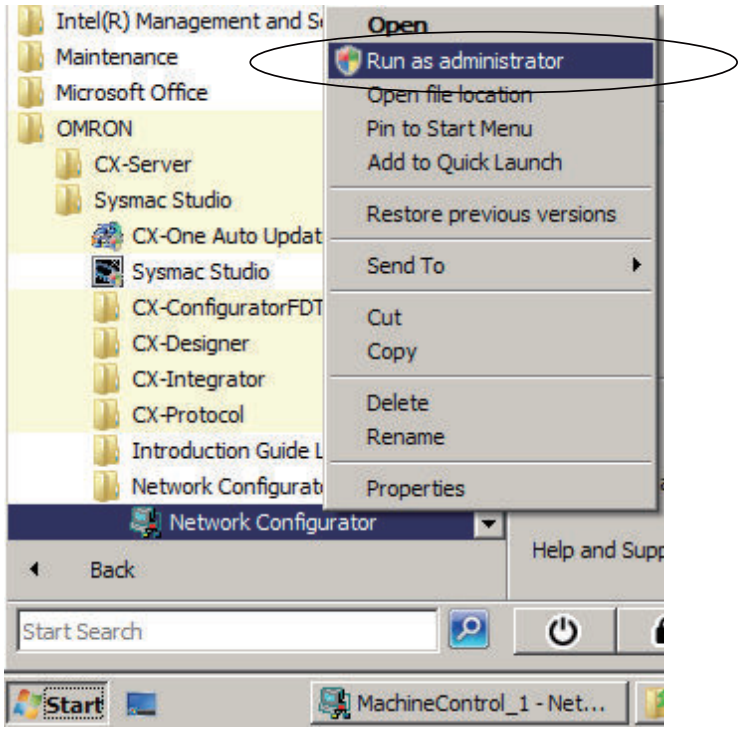
A-3 EDS File Management

This section describes the EDS file management on the Network Configurator.

Precautions for Correct Use

On Windows Vista or Windows 7:
We recommend that you select **Run as administrator** to start the Network Configurator for operations with EDS files.
If you do not select **Run as administrator**, the following condition will result according to Windows user management for security purposes.
The following operations are not valid if you log in with another user account, and you need to restart the Network Configurator again: **Install**, **Create**, **Delete**, and **Create EDS Index File** under **EDS File**.
When you start the Network Configrator, select **Run as administrator** as below.

1. Select the Network Configurator from the Start Menu, and then right-click.
2. Select **Run as administrator** from the displayed pop-up menu.



A-3-1 Installing EDS Files

EDS File - Install

The Network Configurator can support new devices if the proper EDS files are installed.
To install the EDS file, use the following procedure.

1 Select **EDS File** - **Install**.
The Install EDS File Dialog Box is displayed.

2 Select the EDS file to install, and click the **Open** Button.
Next, select the icon file (*.ico). The EDS file is added to the Hardware List as a new device.
If the hardware already exists, the new Hardware List will overwrite the previous one.
If the hardware has different versions, each hardware version is added to the Hardware List.

A-3-2 Creating EDS Files

EDS File - Create

The EDS files are required for the Network Configurator to create a network configuration. To create an EDS file, use the following procedure.

1 Select **EDS File** - **Create**.

2 Set the device information. You can obtain the device information from the device on the network if it is online.

3 The device is added to the Hardware List as a new device, just like when you install an EDS file.

Additional Information

You cannot set device parameters when you create an EDS file with the Network Configurator. Obtain a proper EDS file from the manufacturer of the device to make device parameter settings for the device.

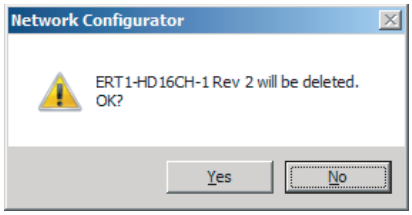
A-3-3 Deleting EDS Files

EDS File - Delete

To delete an EDS file, use the following procedure.

1 Select the device from the Hardware List.

2 Select **EDS File** - **Delete**.
The following confirmation dialog box is displayed.



3 Click the **Yes** Button.
The selected device is deleted from the Hardware List together with the EDS file.

A-3-4 Saving EDS Files

EDS File - Save

To save the EDS file, use the following procedure.

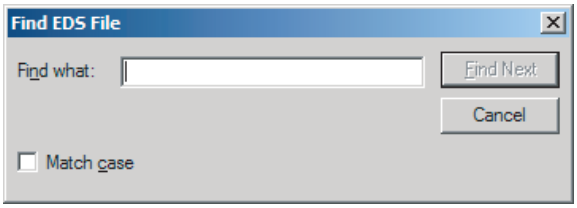
1. Select the target hardware device in the Hardware List, and then select **EDS File** - **Save**.
2. A Save EDS File Dialog Box is displayed.
3. Input the folder and file names and click the **Save** Button.
   The EDS file is saved.

A-3-5 Searching EDS Files

EDS File - Find

To search the devices in the Hardware List for EDS files, use the following procedure.

1. Select **EDS File** - **Find**.
   The following dialog box is displayed.



2. Input the character string to search for, and click the **Find Next** Button.
3. If a matching device is found, the cursor moves to the position of the device.
4. To quit the search operation, click the **Cancel** Button.

Additional Information

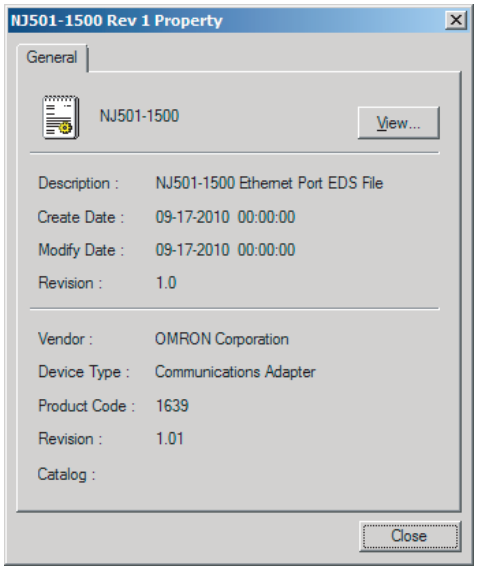
- The search is performed for the device on which the cursor stays and subsequent ones in the Hardware List.
- To search all the devices, select *Hardware* in the Hardware List before you perform the search.

A-3-6 Displaying EDS File Properties

EDS File - Property

To display the properties of the EDS file, use the following procedure.

1. Select the desired hardware (device) from the Hardware List.
2. Select **EDS File** - **Property**.
   The following dialog box is displayed.



   The time and date when the EDS file was created is displayed, along with the device information.

A-3-7 Creating EDS Index Files

EDS File - Create EDS Index File

When an EDS file is manually added or when a device is not correctly indicated in the Hardware List, use the following procedure to recreate the EDS index file.
(This applies to Network Configurator version 3.30 or higher.)

1. Select **EDS File** - **Create EDS Index File**.
2. Restart the Network Configurator.

A-4 Precautions for Using the Network Configurator on Windows XP, Windows Vista, or Windows 7 or Higher

Better firewall security for Windows XP (SP2 or higher), Windows Vista, and Windows 7 or higher has increased the restrictions for data communications. Before connecting the Network Configurator and an NJ/NX-series CPU Unit and starting communications through the following procedures, you may need to change the settings of the Windows firewall as described in this section.

- If you select **Option** - **Select Interface** - **Ethernet I/F**.
- If you select **Option** - **Select Interface** - **NJ/NX Series Ethernet Direct I/F**.
- If you select **Option** - **Select Interface** - **NJ/NX Series USB Port**.

Precautions for Correct Use

The main function of the firewall is to prevent illegal access from external sources (e.g., the Internet). The purpose of changing the firewall settings through this procedure is to connect the Network Configurator to an NJ/NX-series CPU Unit. If your computer is connected to an in-house network, make such changes only after confirming that they have no security impact on the network.

A-4-1 Changing Windows Firewall Settings

Windows XP

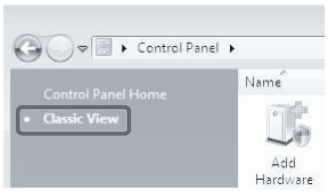
1 When you attempt to connect to the NJ/NX-series CPU Unit from the Network Configurator, the **Windows Security Alert** Dialog Box is displayed.

2 Click the **Unblock** Button.
This allows USB connection and EtherNet/IP connection to the Network Configurator, and you will be able to connect to the NJ/NX-series CPU Unit via the Network Configurator.

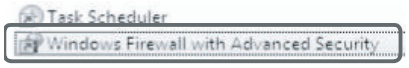
Windows Vista or Windows 7 or Higher

Use the following procedure to change the settings.
Always perform steps 1 to 6 if you cannot go online. The **User Account Control** Dialog Box may be displayed during this procedure. If it appears, click the **Continue** Button and continue with the procedure.

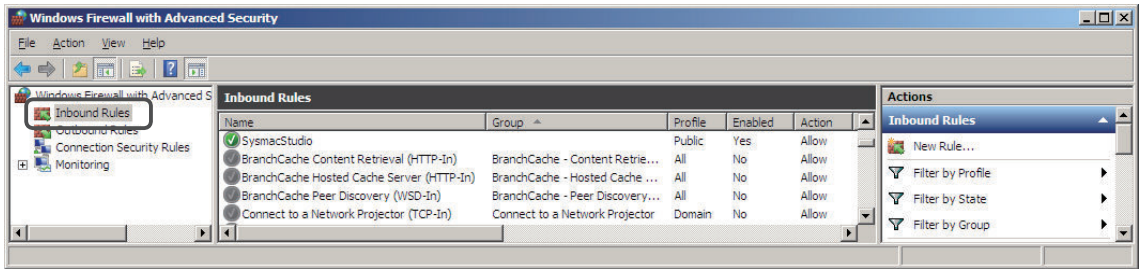
1 Select **Control Panel** from the Windows Start Menu, and select **Classic View** to change the view.



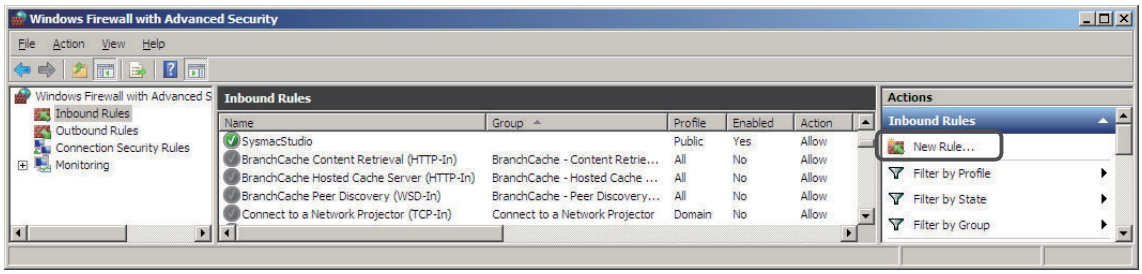
2 Open **Administrative Tools**, and select **Windows Firewall with Advanced Security** in the displayed dialog box.



3 Select **Inbound Rules** under **Windows Firewall with Advanced Security on Local Computer** on the left side of the **Windows Firewall with Advanced Security** Dialog Box.



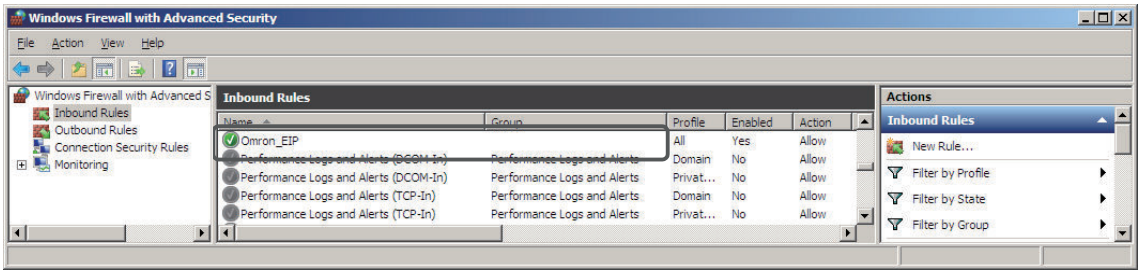
4 Select **New Rule** under **Inbound Rules** in the **Actions** Area on the right side of the dialog box.



5 Follow the steps below to make the settings in the **New Inbound Rule Wizard** Dialog Box. Select the specified option at each step, and click the **Next** Button to move to the next step.

Rule Type	Select **Custom**.
Program	Select **All Programs**.
Protocol and support	Select **ICMPv4** as the protocol type. 
Scope	Select **Any IP address** for all.
Action	Select **Allow the connection**.
Profile	Select **Domain**, **Private**, and **Public**.
Name	Enter an arbitrary name (e.g., Omron_EIP).

6 Click the **Finish** Button. The rule that you defined (i.e., Omron_EIP) is registered in the list of **Inbound Rules**.
Close the **Windows Firewall with Advanced Security** Dialog Box.



7 When you attempt to connect to the NJ/NX-series CPU Unit from the Network Configurator, the **Windows Security Alert** Dialog Box is displayed.

8 Click the **Allow access** Button.

(On Windows 7) This allows USB connection and EtherNet/IP connection to the Network Configurator, and you will be able to connect to the NJ/NX-series CPU Unit via the Network Configurator.

A-5 Variable Memory Allocation Methods

You must be aware of the way in which memory is allocated to variables to align the memory locations of the members of structure or union variables with variables in other devices. Adjustments are necessary mainly when structure or union variables are used in the following type of communications with other devices.

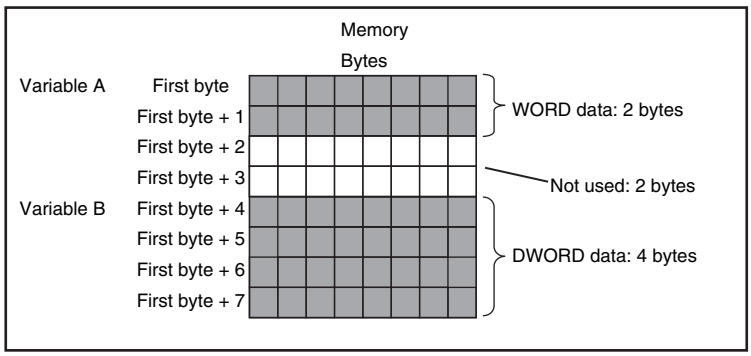
- When using EtherNet/IP tag data links or CIP messages to access variables between NJ/NX-series CPU Units and other CPU Units
- When using structure or union variables to exchange data with devices other than CPU Units, such as ID Tags

A-5-1 Variable Memory Allocation Rules

The amount of memory and the memory locations that are allocated for a variable depend on the data type of the variable. The amount of memory and the memory locations that are allocated for array elements, structure members, and union members depend on the data types, but also on the declarations that are made for the arrays, structures, and unions.

Data Type Alignment and Memory Allocation Amounts

The data size is determined for each data type. The data size is the minimum amount of memory that is required to store the value or values of that data type.
On the other hand, memory for variables is automatically structured by the Controller for the most efficient access. Therefore, the total amount of memory that is required for variables is not necessarily the total of the data sizes of the variables. For example, if WORD and DWORD variables are declared, the total of the data sizes is six bytes, but eight bytes are allocated in memory, as shown in the following figure.



Variable Table

Name	Data type
A	WORD
B	DWORD

This information for determining the location of a variable in memory is called the alignment. The alignment is determined for each data type. The amount of memory and the memory locations for the variables are given below.

Item	Specification
Amount of memory that is allocated	An integral multiple of the alignment. However, the minimum amount of memory is the data size.

Item	Specification
Locations in memory	At an integral multiple of the alignment starting from the start of the variable in memory.

The alignments and the amounts of memory that are allocated for the basic data types and enumerations are given below.

Data type	Alignment [bytes]	Amount of memory that is allocated [bytes]
BOOL	2	2
BYTE, USINT, or SINT	1	1
WORD, UINT, or INT	2	2
DWORD, UDINT, or DINT	4	4
LWORD, ULINT, or LINT	8	8
REAL	4	4
LREAL	8	8
TIME, DATE, TIME_OF_DAY, or DATE_AND_TIME	8	8
STRING[N+1]*1	1	N+1
Enumerations	4	4

*1. N is the maximum number of characters handled. For example, if a maximum of 10 single-byte characters are handled, the NULL character is added, so memory for 11 characters must be reserved.

The elements of arrays and the members of structures and unions are located in memory for the most efficient access. The alignments and the amounts of memory that are allocated for arrays, structures, and unions are determined by the variable declarations, as described below.

Data type	Alignment	Amount of memory that is allocated
Array	Same as alignment of the data type of the elements	(Amount of memory that is allocated for the data type of the elements) × Number of elements*1
Structure	The largest alignment of all of the members	The integral multiple of the alignment that is larger than the total amount of memory that is allocated when the members are arranged in order at integral multiples of the alignment of the data types of the members
Union	The largest alignment of all of the members	The largest amount of memory that is allocated for any of the members

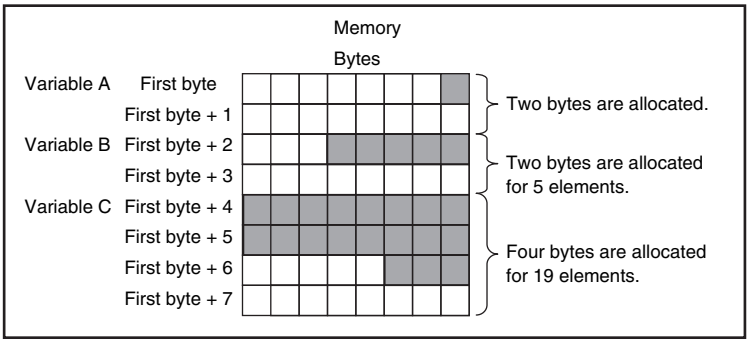
*1. BOOL arrays are an exception. Refer to *Precautions for Correct Use*, below, for the amount of memory that is allocated for BOOL arrays.

Precautions for Correct Use

Amount of Memory That Is Allocated for BOOL Arrays

Two bytes are allocated in memory for individual BOOL variables, BOOL structure members, and BOOL union variables.

However, for a BOOL array, two bytes of memory are not allocated for each element. One bit is allocated in order for each element. For the entire array, a multiple of two bytes of memory is allocated (including unused bits).



Variable Table

Name	Data type
A	BOOL
B	ARRAY[1..5]OF BOOL
C	ARRAY[0..18]OF BOOL

Therefore, the following formula gives the amount of memory that is allocated for a BOOL array. For 1 to 16 elements, 2 bytes are allocated. For 17 to 32 elements, 4 bytes are allocated.

$$\text{Amount of memory} = 2 \left[\frac{\text{Number of elements} - 1}{16} \right] + 2$$

Truncate the decimal portion of the result of the calculation in brackets.

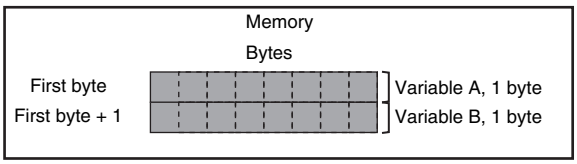
Specific examples of the rules for memory allocation for variables of each data type are given below.

Basic Data Types

● Variables with One-Byte Alignments (e.g., BYTE)

One byte of memory is allocated for the one-byte alignment.

Example: Two consecutive BYTE variables



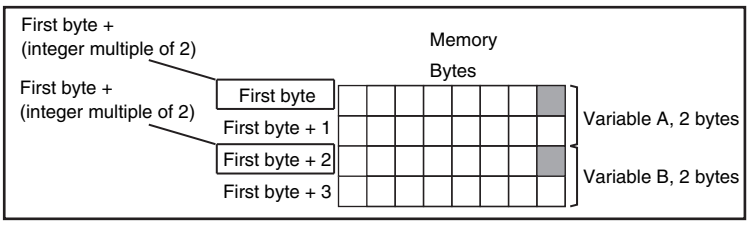
Variable Table

Name	Data type
A	BYTE
B	BYTE

● Variables with Two-byte Alignments (e.g., BOOL and WORD)

Two bytes of memory are allocated for the two-byte alignment.

Example: Two consecutive BOOL variables



Variable Table

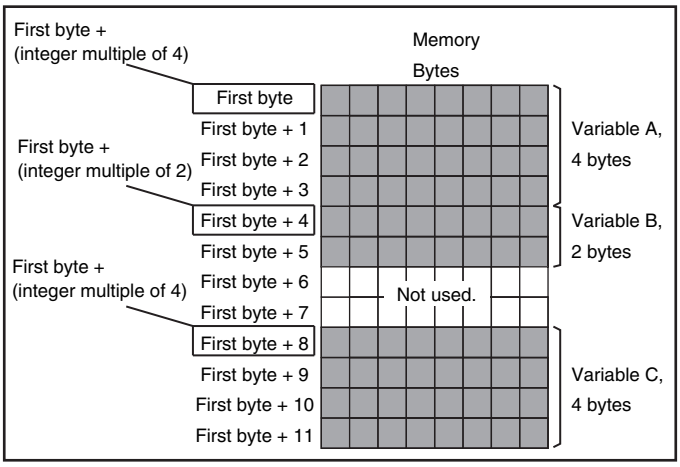
Name	Data type
A	BOOL
B	BOOL

● Variables with Four-byte Alignments (e.g., DWORD)

Four bytes of memory are allocated for the four-byte alignment.

The location of the first byte of data in memory is an integer multiple of four bytes. Therefore, if a variable with a two-byte alignment, such as WORD data, is inserted, two bytes of unused memory will remain.

Example: Consecutive variables in the following order: DWORD, WORD, and DWORD



Variable Table

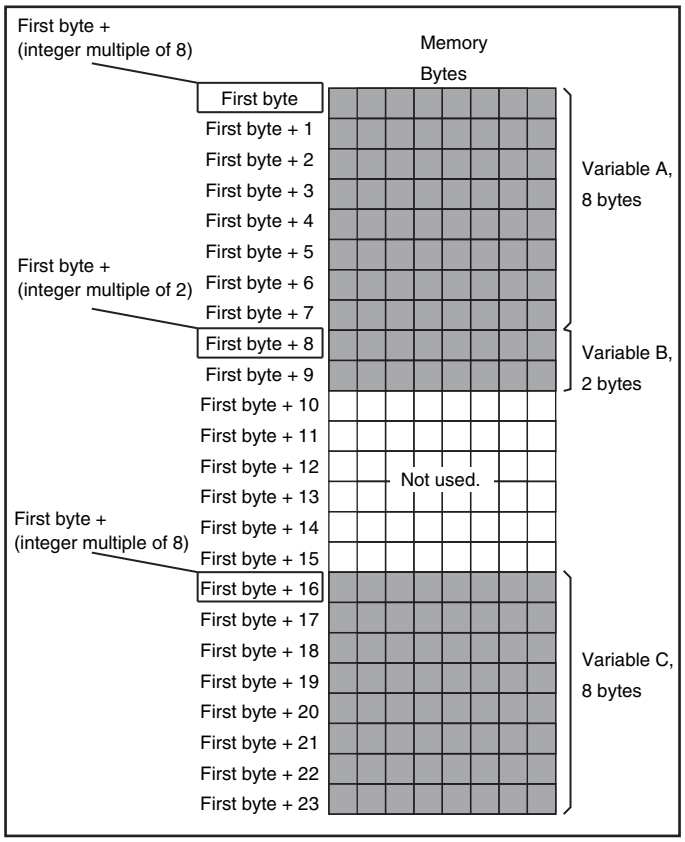
Name	Data type
A	DWORD
B	WORD
C	DWORD

● Variables with Eight-byte Alignments (e.g., LWORD)

Eight bytes of memory are allocated for the eight-byte alignment.

The location of the first byte of data in memory is an integer multiple of eight bytes. Therefore, if a variable with a two-byte alignment, such as WORD data, is inserted, six bytes of unused memory will remain. If a variable with a four-byte alignment, such as DWORD data, is inserted, four bytes of unused memory will remain.

Example: Consecutive variables in the following order: LWORD, WORD, and LWORD



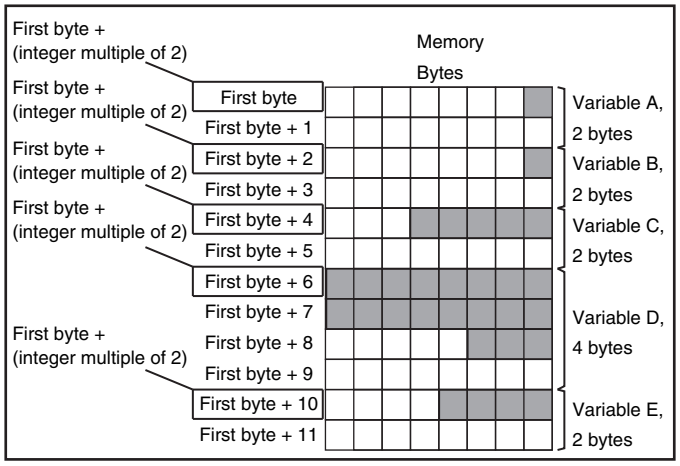
Variable Table

Name	Data type
A	LWORD
B	WORD
C	LWORD

Arrays

A continuous section of memory is allocated for the elements of the array based on the data size of the data type of the array variable. The alignment of an array is the same as alignment of the data type of the elements.

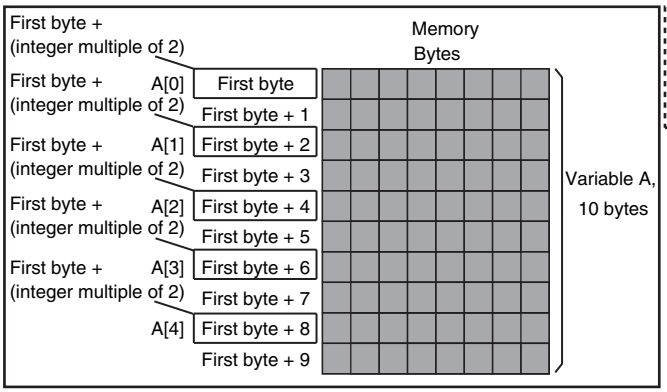
Example: Continuous variables in the following order: two BOOL variable, one BOOL array with five elements, one BOOL array with 19 elements, and one BOOL array with four elements



Variable Table

Name	Data type
A	BOOL
B	BOOL
C	ARRAY[1..5]OF BOOL
D	ARRAY[0..18]OF BOOL
E	ARRAY[5..8]OF BOOL

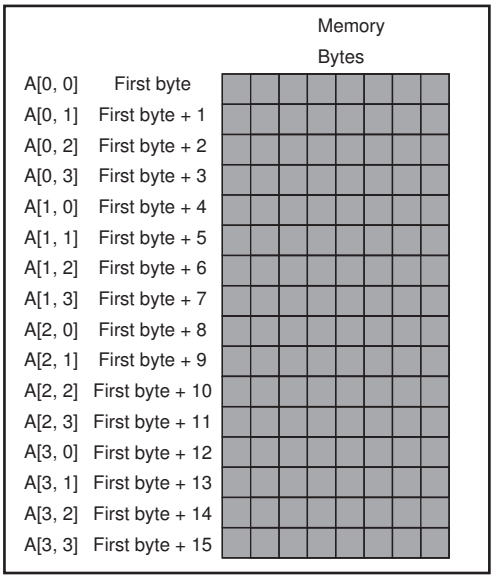
Example: INT array with five elements



Variable Table

Name	Data type
Variable A	ARRAY[0..4] OF INT

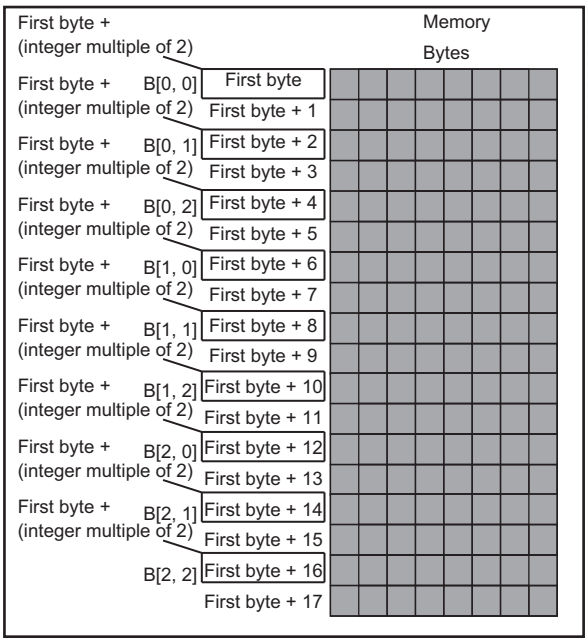
Example: BYTE array with four elements for each dimension with two-dimensional array



Variable Table

Name	Data type
Variable A	ARRAY[0..3, 0..3] OF BYTE

Example: WORD array with three elements for each dimension with two-dimensional array



Variable Table

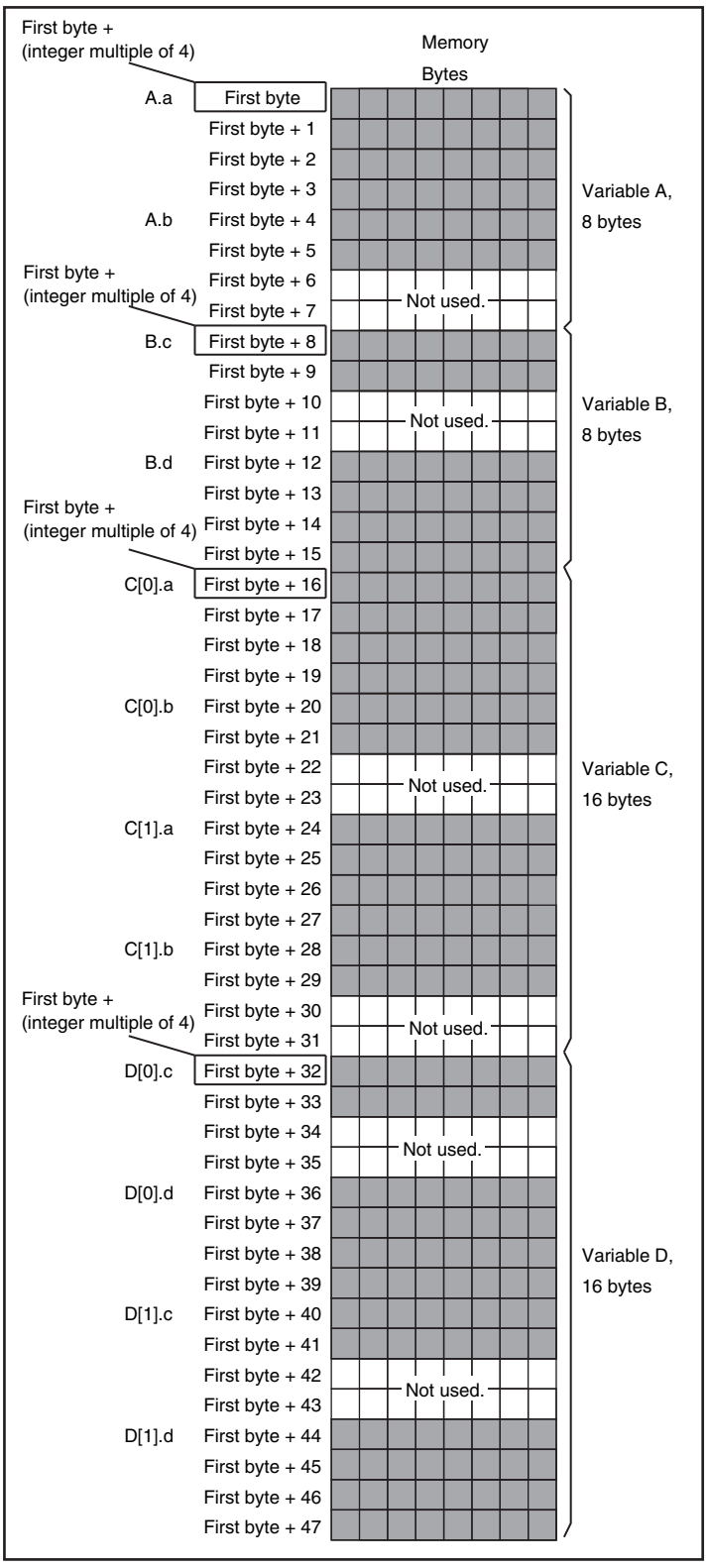
Name	Data type
Variable B	ARRAY[0..2, 0..2] OF WORD

Structures

For a structure variable, the members are located in memory in the order that they are declared. Each member is located at an integer multiple of the alignment of the data type of the member. Therefore, there can be unused memory between members or at the end of members. The alignment of a structure is the largest alignment of all of the members. The amount of memory that is allocated is the integral multiple of the alignment that is larger than the total amount of memory that is allocated when the members are arranged in order at integral multiples of the alignment of the data types of the members.

Example: The alignments and the amounts of memory that are allocated for the four variable declarations given in the following figure are given in the following table.

Variable	Alignment [bytes]	Amount of memory that is allocated [bytes]
A	4	8
B	4	8
C	4	16
D	4	16



Data Type Definitions

Name	Data type
Structure *STR_A*	STRUCT
a	DINT
b	INT

Name	Data type
Structure *STR_B*	STRUCT
c	INT
d	DINT

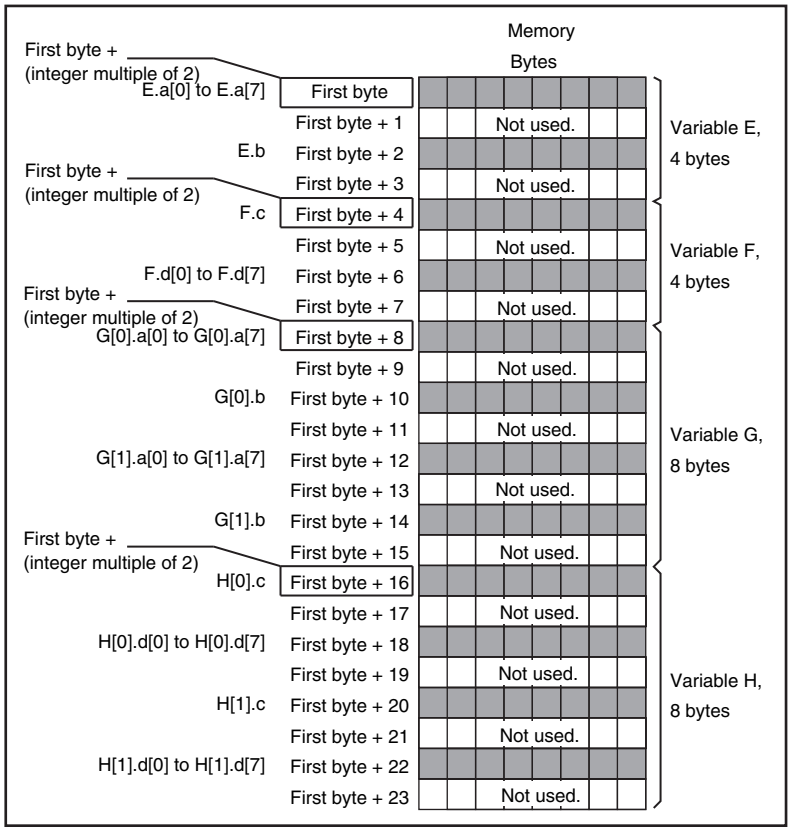
Variable Table

Name	Data type
Variable A	Structure *STR_A*
Variable B	Structure *STR_B*
Variable C	ARRAY[0..1] OF STR_A
Variable D	ARRAY[0..1] OF STR_B

Example: The alignments and the amounts of memory that are allocated for the four variable declarations given in the following figure are given in the following table.

Variable	Alignment [bytes]	Amount of memory that is allocated [bytes]
E	2	4
F	2	4

Variable	Alignment [bytes]	Amount of memory that is allocated [bytes]
G	2	8
H	2	8



Data Type Definitions

Name	Data type
Structure *STR_C*	STRUCT
a	ARRAY[0..7] OF BOOL
b	BYTE

Name	Data type
Structure *STR_D*	STRUCT
c	BYTE
d	ARRAY[0..7] OF BOOL

Variable Table

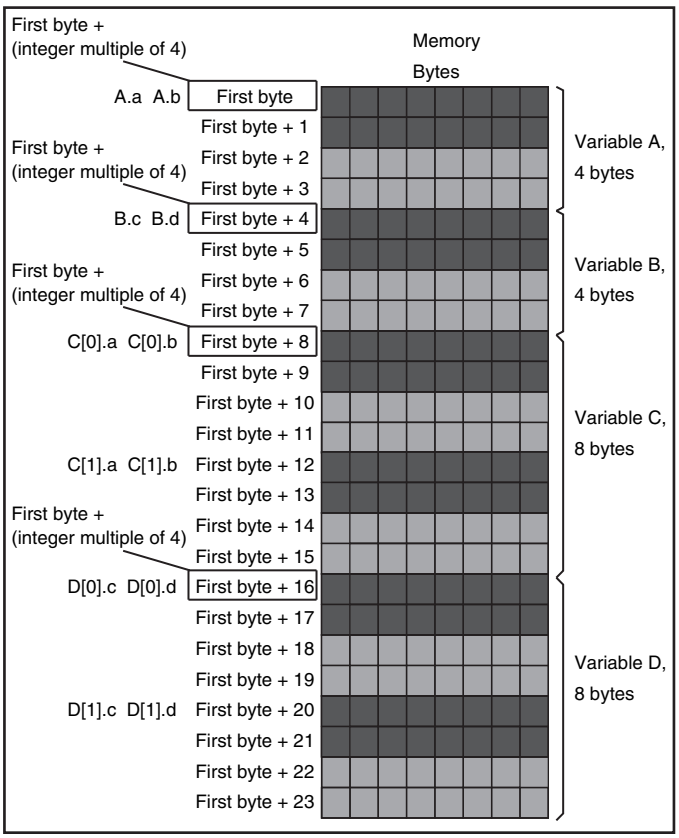
Name	Data type
Variable E	Structure *STR_C*
Variable F	Structure *STR_D*
Variable G	ARRAY[0..1] OF STR_C
Variable H	ARRAY[0..1] OF STR_D

Unions

For a union variable, the members overlap in the same memory locations.
The alignment of a union is largest alignment of all of the members. The amount of memory that is allocated is the largest amount of memory that is allocated for any of the members.

Example: The alignments and the amounts of memory that are allocated for the four variable declarations given in the following figure are given in the following table.

Variable	Alignment [bytes]	Amount of memory that is allocated [bytes]
A	4	4
B	4	4
C	4	8
D	4	8



Data Type Definitions

Name	Data type
Union *UNI_A*	UNION
a	DWORD
b	WORD

Name	Data type
Union *UNI_B*	UNION
c	WORD
d	DWORD

Variable Table

Name	Data type
Variable A	Union *UNI_A*
Variable B	Union *UNI_B*
Variable C	ARRAY[0..1] OF UNI_A
Variable D	ARRAY[0..1] OF UNI_B

A-5-2 Important Case Examples

When you exchange structure variable data between an NJ/NX-series CPU Unit and a remote device, you must align the memory configuration of the structure variable members with those of the remote device.

This section describes what to do in either the NJ/NX-series CPU Unit or in the remote device.

Additional Information

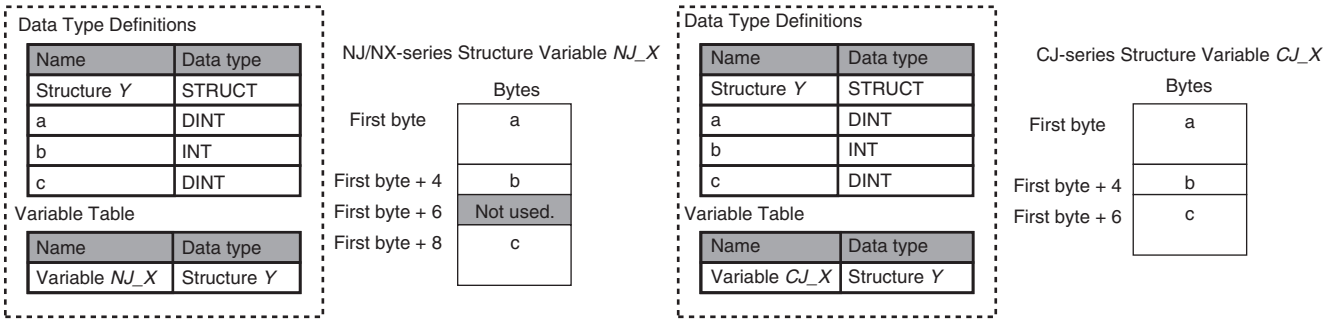
This is not necessary when you exchange data between NJ/NX-series CPU Units.

Aligning the Memory Configuration with a Remote Device

There are two methods that you can use to align the memory configuration with a remote device.

For example, the differences in the memory configuration for structure variables between an NJ/NX-series CPU Unit and a CJ-series CPU Unit are shown below.

This section describes how to align the memory configuration for these Units.



● Method 1: Changing the Memory Configuration of the Structure Variable in the NJ/NX-series CPU Unit

With an NJ/NX-series CPU Unit, you can specify member offsets to change the memory configuration of the members of a structure variable. You can change the memory configuration of the members of a structure variable in the NJ/NX-series CPU Unit so that it is the same as the memory configuration in a remote device that the CPU Unit will communicate with.

Specify the member offsets for a structure variable when you register the structure data type.

To communicate with a CJ-series CPU Unit, you can set the offset type to *CJ* to automatically use the CJ-series memory configuration.

You can set the offset type to *User* to freely set your own offsets.

Version Information

The following table gives the unit version of the CPU Units and the Sysmac Studio version that are required to specify member offsets.

Unit version of CPU Unit	Sysmac Studio version		
	Ver.1.01 or lower	Ver.1.02	Ver.1.03 or higher
Ver.1.01 or later	Not possible.	Possible.*1	Possible.
Ver.1.00	Not possible.	Not possible.	Not possible.

*1. You cannot select the memory offset type. You can set member offsets.

If you change the memory configuration of a structure variable by setting offsets, you must make the same changes for the same structure variable in other NJ/NX-series CPU Units on the network.

Refer to the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for the procedure to change the memory configuration of a structure variable.

Example: The following example shows how the memory configuration of the structure variable in the NJ/NX-series CPU Unit is changed to match the memory configuration of the structure variable in the CJ-series CPU Unit.

Data Type Definitions

Name	Data type
Structure *Y*	STRUCT
a	DINT
b	INT
c	DINT

Variable Table

Name	Data type
Variable *NJ_X*	Structure *Y*

NJ/NX-series Structure Variable *NJ_X*

Bytes

First byte	a
First byte + 4	b
First byte + 6	Not used.
First byte + 8	c

Communications is not possible because the memory configuration is not the same.

Data Type Definitions

Name	Data type
Structure *Y*	STRUCT
a	DINT
b	INT
c	DINT

Variable Table

Name	Data type
Variable *CJ_X*	Structure *Y*

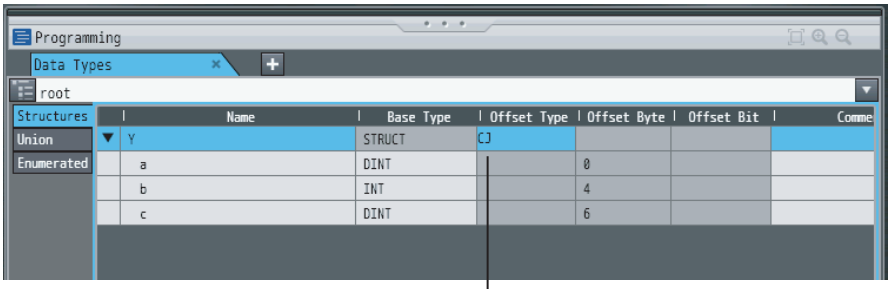
CJ-series Structure Variable *CJ_X*

Bytes

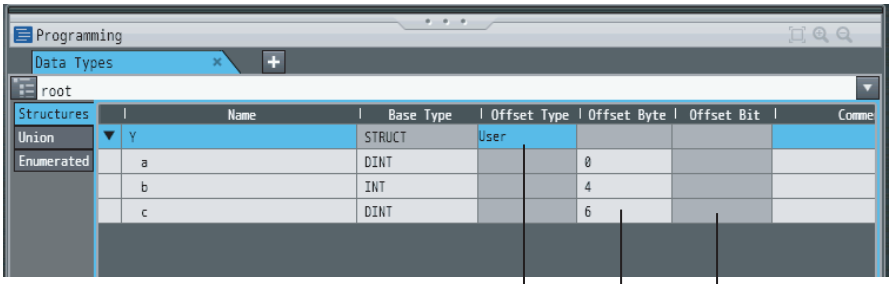
First byte	a
First byte + 4	b
First byte + 6	c

To align the memory configurations in the NJ-series and CJ-series CPU Units, offsets are set in the Sysmac Studio.

Here, the following offsets are set for member c of data type Y of the structure variable NJ_X.



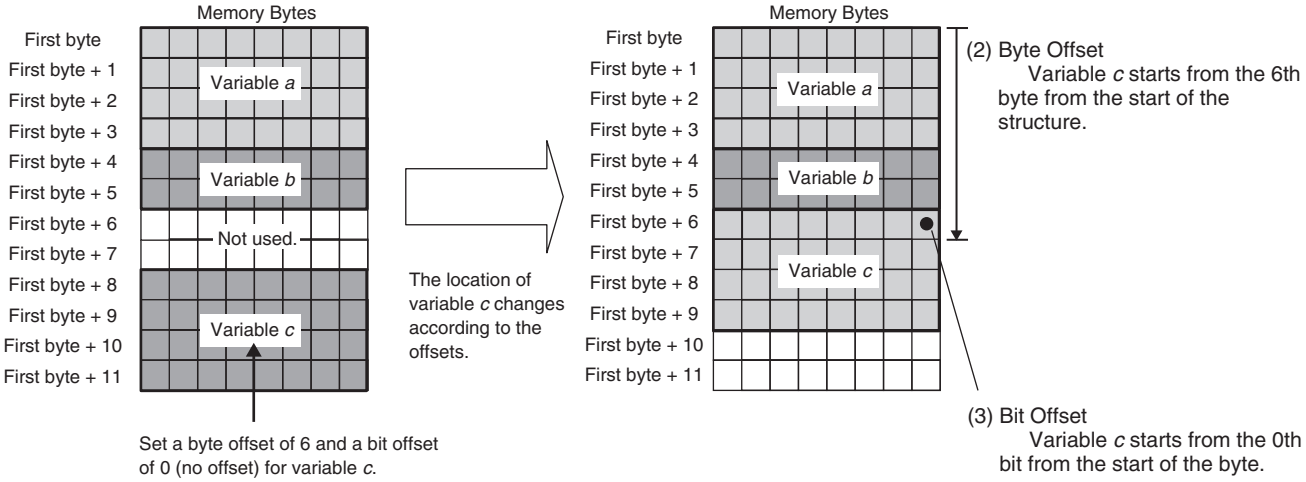
(1) Offset type is set to *CJ*.



(3) Bit Offset
Set the location of the first bit of the member variable.

(2) Byte Offset
Set the location of the first byte of the member from the beginning of the structure variable.

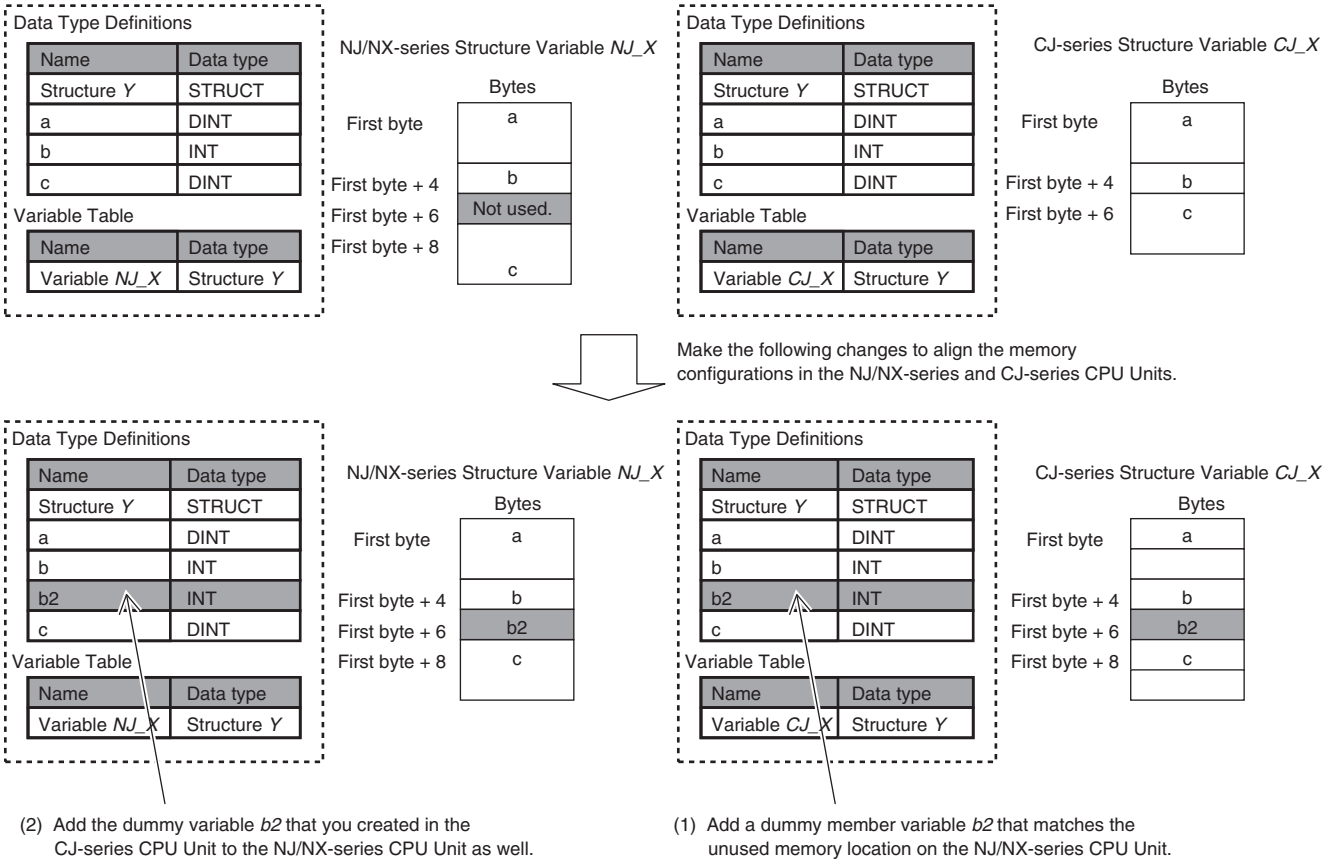
(1) Offset Type
Specify *User*.



● Method 2: Changing the Memory Configuration of the Structure Variable in the Remote Device

You can insert a member into the structure variable of the remote device to change it to match the memory configuration of the structure variable in the NJ/NX-series CPU Unit.
Both the memory configuration and the data types must be the same between the two structure variables. You therefore need to create the same members in both the remote device and the NJ/NX-series CPU Unit.

Example: The following example shows how the memory configuration of the structure variable in the CJ-series CPU Unit is changed to match the memory configuration of the structure variable in the NJ/NX-series CPU Unit.



A-6 Precautions When Accessing External Outputs in CPU Units

Observe the following precautions when you access variables or I/O memory addresses that are assigned to external outputs in an NJ/NX-series CPU Unit.

● Precaution on Writing from External Devices, Variables That Are Assigned to External Outputs

Any value that is written to a variable that is assigned to an external output in an NJ/NX-series CPU Unit through a tag data link or communications instruction will be overwritten by the execution results of the user program.

The value that is written from the tag data link or communications instruction will therefore not be output to the external device.

The following types of variable are assigned to the external outputs.

CPU Unit Common

- The device variables (or global variables) that are assigned to an I/O port of an EtherCAT output slave

NJ-series CPU Unit

- The devices variables (or global variables) that are assigned to an I/O port of a CJ-series Basic Output Unit
- The global variables with AT specifications to output bits that are assigned to CJ-series Basic Output Units

NX102 CPU Unit and NX1P2 CPU Unit

- The global variables with AT specifications to the memory used for CJ-series Units, of which Network Publish attributes are set to output

● Precaution When Directly Writing to I/O Memory Addresses Assigned to Output Bits for CJ-series Basic Output Units

Any value that is written to an I/O memory address that corresponds to an output bit that is assigned to a CJ-series Basic Output Unit through a tag data link will be overwritten by the execution results of the user program.

The value that is written directly to the I/O memory address from the tag data link will therefore not be output to the external device.

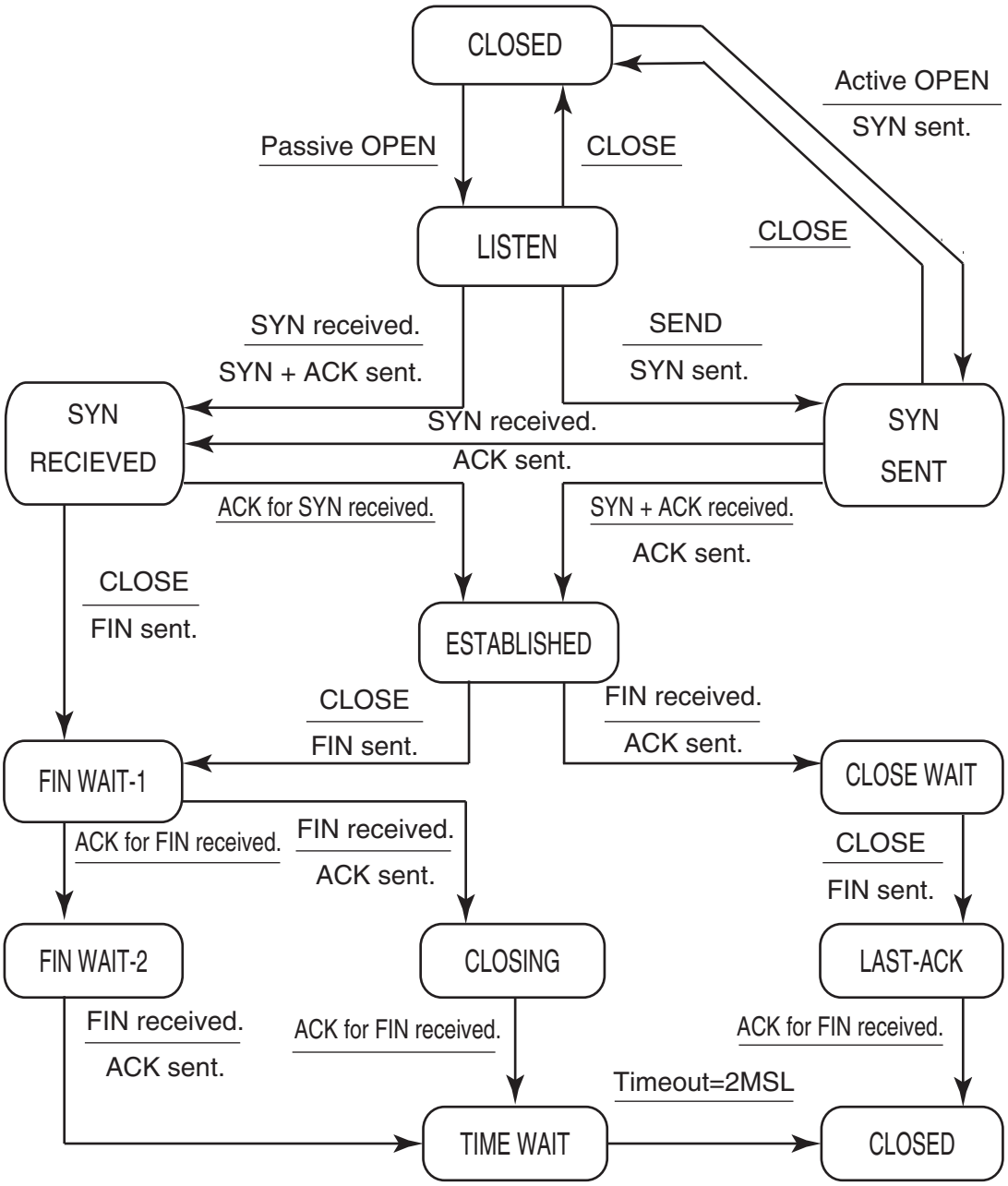
A-7 TCP State Transitions

There are 11 types of TCP connection state.
You can check the TCP state with the TCP connection status that is output by the SktGetTCPStatus (Read TCP Socket Status) instruction.
The table below shows the TCP states and what each state means.

TCP state	Definition
CLOSED	The connection is closed.
LISTEN	The server is waiting for a connection request (SYN) with a passive open.
SYN SENT	The client sent a connection request (SYN) for an active open and is waiting for acknowledgment (SYN + ACK).
SYN RECEIVED	The server sent an acknowledgment (SYN + ACK) to a connection request (SYN) and is waiting for acknowledgment (ACK).
ESTABLISHED	A connection is established.
CLOSE WAIT	The server sent acknowledgment (ACK) to a connection close request (FIN) and is waiting for the server application to be ready to close.
FIN WAIT-1	The client sent a connection close request (FIN) and is waiting for acknowledgment (ACK).
CLOSING	The client and server simultaneously received a connection close request (FIN) and are waiting for acknowledgment (ACK).
LAST-ACK	The server sent a connection close request (FIN) and is waiting for acknowledgment (ACK).
FIN WAIT-2	The client is waiting for a connection close request (FIN).
TIME WAIT	The client received acknowledgment (ACK) to a connection close request (FIN) and is waiting for it to be received and processed by the server.

The TCP state changes as requests and acknowledgments are received from the remote node, and as TCP socket connection and close instructions are executed in the user program.
When the state changes, a connection request (SYN), close request (FIN), and acknowledgment (ACK) to those requests are sent to and received from the remote node.

The following figure shows TCP state transitions.
The TCP states are given in □ in the figure. Between states, the text in the upper row indicates the condition for the state change, and the text in the lower row indicates the action that is performed at the state change. (If no action is performed, none is given.)
Example: When SYN and ACK are received in *SYN SENT* state, ACK is sent and the state changes to *ESTABLISHED*.


CLOSED
Active OPEN
SYN sent.
Passive OPEN
CLOSE
LISTEN
CLOSE
SYN received.
SYN + ACK sent.
SEND
SYN sent.
SYN
RECIEVED
SYN received.
ACK sent.
SYN
SENT
ACK for SYN received.
SYN + ACK received.
ACK sent.
CLOSE
FIN sent.
ESTABLISHED
CLOSE
FIN sent.
FIN received.
ACK sent.
FIN WAIT-1
CLOSE WAIT
ACK for FIN received.
FIN received.
ACK sent.
CLOSE
FIN sent.
FIN WAIT-2
CLOSING
LAST-ACK
FIN received.
ACK sent.
ACK for FIN received.
ACK for FIN received.
Timeout=2MSL
TIME WAIT
CLOSED

A-8 Example of NX Unit Setting Using NX Configuration Object Service

You can change the NX Unit settings by using the NX Configuration object service.
This section provides examples of the procedure for NX Unit setting using the NX Configuration object service.
Refer to *7-5-3 NX Configuration Object (Class ID: 74 hex)* on page 7-52 for details on the NX Configuration object.

The following three types of procedure are given as the examples.
- Changing the Unit operation settings for a singe NX Unit.
- Changing the Unit operation settings for multiple NX Units.
- Initializing the Unit operation settings for a singe NX Unit.

Precautions for Correct Use

Refer to *15-2 Checking Status with the Network Configurator* on page 15-3 for troubleshooting errors that may occur while setting NX Units using the NX Configuration object service.

Version Information

You can perform the NX Unit setting using the NX Configuration object service only with NX102 CPU Units.

A-8-1 Changing the Unit Operation Settings for Singe NX Unit

Change the Unit operation settings for a single NX Unit mounted to the Controller. In this example, the unit number of the NX Unit is 1.
The following table gives the setting procedure.

Step	Description	CIP Object to use			
		Class ID	Instance ID	Service code	Unit number
1	Change the parameter write mode of the NX Unit to Write mode.	0x74 NX Configuration object	0x01	0x37 Switch parameter write mode	0x01
2	Write values to the NX object of the NX Unit.	0x74 NX Configuration object	0x01	0x34 Write NX object	0x01
3	Save the values that are set in the NX Unit.	0x74 NX Configuration object	0x01	0x36 Save parameter	0x01
4	Restart the NX Unit.	0x74 NX Configuration object	0x01	0x35 Restart NX unit	0x01

A-8-2 Changing the Unit Operation Settings for Multiple NX Units

Change the Unit operation settings for multiple NX Units mounted to the Controller. In this example, the unit numbers of the NX Units are 1 and 2.
The following table gives the setting procedure.

Step	Description	CIP Object to use			
		Class ID	Instance ID	Service code	Unit number
1	Change the parameter write mode of the NX Unit with unit number 1 to Write mode.	0x74 NX Configuration object	0x01	0x37 Switch parameter write mode	0x01
2	Change the parameter write mode of the NX Unit with unit number 2 to Write mode.	0x74 NX Configuration object	0x01	0x37 Switch parameter write mode	0x02
3	Write values to the NX object of the NX Unit with unit number 1.	0x74 NX Configuration object	0x01	0x34 Write NX object	0x01
4	Write values to the NX object of the NX Unit with unit number 2.	0x74 NX Configuration object	0x01	0x34 Write NX object	0x02
5	Save the values that are set in the NX Unit with unit number 1.	0x74 NX Configuration object	0x01	0x36 Save parameter	0x01
6	Save the values that are set in the NX Unit with unit number 2.	0x74 NX Configuration object	0x01	0x36 Save parameter	0x02
7	Restart the NX Unit with unit number 1.	0x74 NX Configuration object	0x01	0x35 Restart NX unit	0x01
8	Restart the NX Unit with unit number 2.	0x74 NX Configuration object	0x01	0x35 Restart NX unit	0x02

A-8-3 Initializing the Unit Operation Settings for Singe NX Unit

Initialize the Unit operation settings for a single NX Units mounted to the Controller. In this example, the unit number of the NX Unit is 1.
The following table gives the setting procedure.

Step	Description	CIP Object to use			
		Class ID	Instance ID	Service code	Unit number
1	Change the parameter write mode of the NX Unit to Write mode.	0x74 NX Configuration object	0x01	0x37 Switch parameter write mode	0x01
2	Initialize the Unit operation settings for the NX Unit with unit number 1.	0x74 NX Configuration object	0x01	0x3D Initialize unit operation parameter	0x01
3	Restart the NX Unit with unit number 1.	0x74 NX Configuration object	0x01	0x35 Restart NX unit	0x01

A-9 Procedure to Use Secure Socket Service with Secure Socket Configuration Commands

This section describes the procedure to use secure socket services for the following use cases.

- Starting to use secure socket services
  Refer to *A-9-1 Settings for Starting Secure Socket Services* on page A-66.
- Replacing CPU Units
  Refer to *A-9-2 Procedure for Replacing the CPU Unit* on page A-68.

A-9-1 Settings for Starting Secure Socket Services

The following two procedures describe how to set up a new configuration.

- If you do not use a client certificate and a client private key
- If you use a client certificate and a client private key

For details on Secure Socket Configuration commands that are used in the procedures, refer to *A-10 Secure Socket Configuration Commands* on page A-73.

If you do not use a client certificate and a client private key

The setting procedure to start secure socket services when the client certificate and client private key are not used is as follows.

As a prerequisite, set the built-in EtherNet/IP of the CPU Unit as follows.

- If the server is on the Internet, configure the default gateway and routing table.
  If the server is specified by an item other than the IP address, such as "xxx.com", configure the DNS server settings.
- Configure NTP settings.
  The NTP settings are optional. It is recommended for matching with the server time.

Check with the network administrator of the installation site for the settings of the default gateway, routing table, DNS server, and NTP server.

The options for Secure Socket Configuration commands in this procedure are described in the following example.

- To connect the computer to the CPU Unit, an EtherNet/IP port is used. They are connected through Ethernet connection via a Hub or remote connection via USB.
- The IP address of the built-in EtherNet/IP Port 1 of the CPU Unit is set to 192.168.250.1.
- Set the session ID to 0 in the secure socket setting.

1 Configure the server and check the server's IP address, HOST name, and other settings.
Check with the server installer for details on how to check.

2 Configure the secure socket setting.
Use the Secure Socket Configuration commands to configure secure socket setting for the session ID. Set different session IDs for all connected destinations.

```
tlsconfig setSessionInfo /id 0 /ip:192.168.250.1
```

To enable secure socket communications log, execute the following command.

```
tlsconfig setLogLevel /enable /ip:192.168.250.1
```

3 Create a user program.
Create a session for secure socket communications with SktTCPConnect instruction to the server in step 1. Set the TLS session name for the session ID to *TLSSessionName*, which is the input variable of SktTLSConnect instruction. If "N" is *0*, TLS session name is *TLSSession0*.

Use SktTLSRead and SktTLSWrite instructions to process data communication with the server.

4 Download the user program using the synchronization function.
Download the user program from the computer to the CPU Unit.
After sufficiently confirming that the connection destination is correct, start operation.

If you use a client certificate and a client private key

The setting procedure to start secure socket services when the client certificate and client private key are used is as follows.
As a prerequisite, set the built-in EtherNet/IP of the CPU Unit as follows.

- If the server is on the Internet, configure the default gateway and routing table.
  If the server is specified by an item other than the IP address, such as "xxx.com", configure the DNS server settings.
- Configure NTP settings.
  The NTP settings are optional. It is recommended for matching with the server time.

Check with the network administrator of the installation site for the settings of the default gateway, routing table, DNS server, and NTP server.

The options for Secure Socket Configuration commands in this procedure are described in the following example.

- To connect the computer to the CPU Unit, an EtherNet/IP port is used. They are connected through Ethernet connection via a Hub or remote connection via USB.
- The IP address of the built-in EtherNet/IP Port of the CPU Unit is set to 192.168.250.1.
- The session ID set in the secure socket setting is 0.

1 Prepare the client private key, client certificate, and CA certificate.
In this procedure, the path and filename of the prepared client certificate is “C:\dir1\dir2\0\client.cert”. The path and filename of the client private key is “C:\dir1\dir2\0\client.key”.
Note that the prepared client certificate and client private key must be stored and managed by the customer.

2 Install the client certificate and CA certificate on the server.
Check with the server administrator for details such as whether installation on the server is required.

3 Configure the server and check the server's IP address, HOST name, and other settings.
Check with the server installer for details on how to check.

4 Configure the secure socket setting.
Use the Secure Socket Configuration commands to configure session information for the session ID.

```
tlsconfig setLogLevel /enable /ip:192.168.250.1
```

To enable secure socket communications log, execute the following command.

```
tlsconfig setLogLevel /enable /ip:192.168.250.1
```

5 Create a user program.
Create a session for secure socket communications with SktTCPConnect instruction to the server confirmed in step 3. Set the TLS session name for the session ID to *TLSSessionName*, which is the input variable of SktTLSConnect instruction. If "N" is *0*, TLS session name is *TLSSession0*.

Use SktTLSRead and SktTLSWrite instructions to process data communication with the server.

6 Download the user program using the synchronization function.
Download the user program from the computer to the CPU Unit.
After sufficiently confirming that the connection destination is correct, start operation.

A-9-2 Procedure for Replacing the CPU Unit

This section describes the following three procedures for replacing the CPU Unit.
- If you do not use a client certificate and a client private key
- If you have stored the client certificate and client private key
- If you have not stored the client certificate and client private key

When you replace the CPU Unit, be sure to perform the following steps before proceeding to the replacement procedure.
For more information about Secure Socket Configuration commands, refer to *A-10 Secure Socket Configuration Commands* on page A-73.
The options for Secure Socket Configuration commands in this procedure are described in the following example.
- To connect the computer to the CPU Unit, an EtherNet/IP port is used. They are connected through Ethernet connection via a Hub or remote connection via USB.
- The IP address of the built-in EtherNet/IP Port of the CPU Unit is set to 192.168.250.1.
- The session ID set in the secure socket setting is 2.

1 Back up the data in the Controller.
Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for details on Controller backups.

2 Read the secure socket setting.
Use the Secure Socket Configuration commands to save the secure socket setting to <PC_Folder>.

```
tlsconfig getAllSessionInfo /f /o <PC_Folder> /ip:192.168.250.1
```

Read and confirm the enable/disable status of the secure socket communications log.

```
tlsconfig getLogLevel /ip:192.168.250.1
```

3 Check that the client certificate and client private key are stored.
Check the read secure socket seting to ensure that the required client private key is stored.

If you do not use a client certificate and a client private key

The procedure for replacing the CPU Unit when the client certificate and client private key are not used is as follows.

The options for the Secure Socket Configuration commands in the replacement procedure are described in the following example.

- To connect the computer to the CPU Unit, an EtherNet/IP port is used. They are connected through Ethernet connection via a Hub or remote connection via USB.
- The IP address of the built-in EtherNet/IP Port of the CPU Unit is set to 192.168.250.1.
- The session ID in the secure socket setting before replacement is set to 2.

1 Replace to a new CPU Unit.

2 Check the secure socket setting.
Confirm the session ID that is being used by the secure socket setting before replacing the CPU Unit. Read the session ID with Secure Socket Configuration commands.

3 Configure the secure socket setting.

```
tlsconfig setLogLevel /enable /ip:192.168.250.1
```

To enable secure socket communications log, execute the following command.

```
tlsconfig setLogLevel /enable /ip:192.168.250.1
```

4 Check the secure socket setting.
Use the Secure Socket Configuration commands to view the secure socket setting and verify that it matches the session ID set in the <PC Folder> read in step 2 of *A-9-2 Procedure for Replacing the CPU Unit* on page A-68. In this procedure, the /o option is not used.

```
tlsconfig getLogLevel /ip:192.168.250.1
```

Read and confirm the enable/disable status of the secure socket communications log.

```
tlsconfig getLogLevel /ip:192.168.250.1
```

5 Restore data to the Controller.
Restore is performed using the backed up data.
Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for details on restoration on the Controller.

6 Check the operation.
Verify that the program and settings are restored and the Controller is working correctly.

If you have stored the client certificate and client private key

The procedure for replacing the CPU Unit when the client certificate and client private key have been stored is as follows.

The options for the Secure Socket Configuration commands in the replacement procedure are described in the following example.

- To connect the computer to the CPU Unit, an EtherNet/IP port is used. They are connected through Ethernet connection via a Hub or remote connection via USB.
- The IP address of the built-in EtherNet/IP Port of the CPU Unit is set to 192.168.250.1.
- The session ID in the secure socket setting before replacement is set to 2.
- The file name and path in the computer that stores the client certificate file used in the secure socket setting of session ID=2 is "C:\dir1\dir2\2\client.cert".
- The path and file name of the client private key file stored on the computer used in the secure client setting of session ID=2 is “C:\dir1\dir2\2\client.key”.

1 Replace to a new CPU Unit.

2 Check the secure socket setting.
Confirm the session ID that is being used by the secure socket setting before replacing the CPU Unit. Read the session ID with Secure Socket Configuration commands.
Prepare the client certificate and client private key for each session ID that are stored in the computer.

3 Configure the secure socket setting.
Use the Secure Socket Configuration commands to configure session information for each session ID.

```
tlsconfig setSessionInfo /id 2 /cert C:\dir1\dir2\2\client.cert /key C:\dir1
\dir2\2\client.key /ip:192.168.250.1
```

To enable secure socket communications log, execute the following command.

```
tlsconfig setLogLevel /enable /ip:192.168.250.1
```

4 Check the secure socket setting.
Use the Secure Socket Configuration commands to view the secure socket setting and verify that it matches the session ID set in the <PC Folder> read in step 2 of *A-9-2 Procedure for Replacing the CPU Unit* on page A-68. In this procedure, the /o option is not used.

```
tlsconfig getLogLevel /ip:192.168.250.1
```

Read and confirm the enable/disable status of the secure socket communications log.

```
tlsconfig getLogLevel /ip:192.168.250.1
```

5 Restore data to the Controller.
Restore is performed using the backed up data.
Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for details on restoration on the Controller.

6 Check the operation.
Verify that the program and settings are restored and the Controller is working correctly.

If you have not stored the client certificate and client private key

The procedure for replacing the CPU Unit when the client certificate and client private key have not been stored is as follows.

1 Create a client certificate and client private key.
Depending on whether you are creating a client certificate and client private key on the server or preparing the client private key and client certificate yourself, the procedures are different as follows.

Creating a client certificate and client private key on the server

1) Create a client certificate and client private key on the server and download them to the computer.
In this procedure, the path and filename of the downloaded client certificate is "C:\dir1\dir2\2\client.cert". The path and filename of the client private key is "C:\dir1\dir2\2\client.key".
Note that the prepared client certificate and client private key must be stored and managed by the customer.

Creating a client certificate and client private key yourself

1) Prepare the client certificate, client private key, and CA certificate.
In this procedure, the path and filename of the prepared client certificate is "C:\dir1\dir2\2\client.cert". The path and filename of the client private key is "C:\dir1\dir2\2\client.key".
Note that you must store and manage the prepared client certificate, client private key, and CA certificate yourself.
2) Install the client certificate and CA certificate on the server.
Check with the server administrator for details such as whether installation on the server is required.

2 Check the secure socket setting.
Confirm the session ID that is being used by the secure socket setting before replacing the CPU Unit. Read the session ID with Secure Socket Configuration commands.
Prepare the client certificate and client private key for each session ID that are stored in the computer.

3 Configure the secure socket setting.
Use the Secure Socket Configuration commands to configure session information for each session ID.

```
tlsconfig setSessionInfo /id 2 /cert C:\dir1\dir2\2\client.cert /key C:\dir1
\dir2\2\client.key /ip:192.168.250.1
```

To enable secure socket communications log, execute the following command.

```
tlsconfig setLogLevel /enable /ip:192.168.250.1
```

4 Restore data to the Controller.
Restore is performed using the backed up data.
Refer to the *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* for details on restoration on the Controller.

5 Check the operation.
Verify that the program and settings are restored and the Controller is working correctly.

A-10 Secure Socket Configuration Commands

Secure Socket Configuration commands are command line tools.
When a command is entered on the command line, the CPU Unit is temporarily connected online and the secure socket setting in the CPU Unit is updated.

Precautions for Correct Use

To reduce the risk of unauthorized access by a third party using the Secure Socket Configuration commands, consider setting operation authority verification on the CPU Unit. You can restrict the use of Secure Socket Configuration commands to administrators only.
For details on how to set operation authority verification, refer to *Operation Authority Verification* on the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)*.
Refer to *Operation Authority Verification* on page A-75 for operating specifications of Secure Socket Configuration commands when operation authority verification is set.

The functions of the Secure Socket Configuration commands are described in the table below.

Function	Description
TLS session setting	• You can register the TLS session information to the secure socket setting in the CPU Unit. You can also read and delete the registered TLS session information. Set one TLS session for one socket used in the secure socket communications. • You can transfer the client certificate and client private key as required.
Secure socket communications log setting	You can set to enable or disable the secure socket communications log. You can also read the set enable or disable status of secure socket communications log.

A-10-1 Operating Environment for Secure Socket Configuration Commands

The operating environment of the Secure Socket Configuration Commands on the computer is as follows.

Item	System requirement
Communications port	USB 2.0 port or Ethernet port

The operating environment other than above, such as the operating system, is the same as that for the Sysmac Studio. Refer to the *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for details on the operating environment of the Sysmac Studio.

A-10-2 Location and Starting Procedure of Secure Socket Configuration Commands

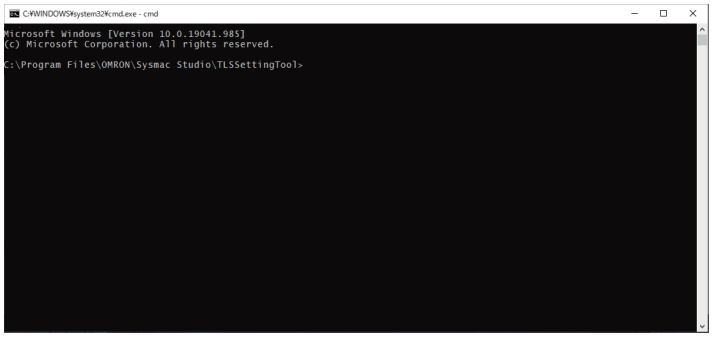
Location

The Secure Socket Configuration Commands are stored in the following folder under the Sysmac Studio installation folder.
.\TLSSettingTool\tlsconfig.exe

Procedure

To start the Secure Socket Configuration commands, proceed as follows.

1 From Windows Start menu, select **OMRON** – **Sysmac Studio** – **Tools** – **Secure Socket Configuration Command**.
The command prompt starts in the folder where tlsconfig.exe is located.



2 From the command line, run tlsconfig.exe.

A-10-3 Command and Option Formats

The table below describes the meaning of command and option symbols that are used in each command.

Symbol	Meaning
☐ (square)	Indicates single-byte space.
| (stroke)	Indicates separation between items for multiple items. E.g. "A|B|C" means that "A, B, or C".
{} (wave brackets)	An item must be selected out of ones within this symbol. Separation of items is indicated by "|". E.g. “{A|B|C}” indicates that "one of the A, B, or C must be specified".
[] (square brackets)	Item enclosed in this symbol can be omitted. E.g. "[A]" indicates that "A is specified as needed".
... (dot line)	More than one item of the previous one described before this symbol can be specified. When more than one item is specified, a single-byte space is used to separate the items. E.g. "A B..." indicates "A can be followed by several B".

Symbol	Meaning
_ (underline)	Indicates the default values when items are omitted. E.g.: “A|B” indicates that when neither A nor B was specified, A was specified.

The format of the command to be entered is as follows.

a. The order of the options is random.
b. A single-byte space is entered as an separator of options.
c. Case-sensitive for both commands and options.
d. Specify an option with "/" (slash).
e. An error occurs in the following cases:
   - You specified a command that does not exist.
   - You specified an option that does not exist.
   - You specified the same option.
   - The number of options does not match.
   - When there is an option to specify one from more than one, you specified more than one.

A-10-4 Common Specifications to All Commands

The function, specification of connection method and execution result displays that are common to all commands and options are described below.
<TLSSettingTool Folder> in the execution example indicates the <Sysmac Studio Installed Folder> \TLSSettingTool folder.

Operation Authority Verification

If the CPU Unit was configured to be operated with different operation authorities, the user is prompted to enter the password before the command is executed.
If the password is correct, the command is executed. If the password is wrong, an error is displayed.
The following dialog to enter password is displayed.

```
Operation authority: Administrator
Password:
```

An example is shown below.

- When operation authority is set
  - If the password is correct

```
<TLSSettingTool Folder>>tlsconfig setSessionInfo /id 1 /key C:\private\foo.private.key /cert
C:\certs\foo.cert.pem /ip:192.168.250.1
Operation authority: Administrator
Password:******

000: Success
<TLSSettingTool Folder>>
```

  - If the password is correct and the command execution result is displayed

```
<TLSSettingTool Folder>>tlsconfig getSessionInfo /id 0 /ip:192.168.250.1
Operation authority: Administrator
Password:******
Id=0
PrivateKey=private.key
Certificate=client.crt
Description=

000: Success
<TLSSettingTool Folder>>
```

- If the password is incorrect

```
<TLSSettingTool Folder>>tlsconfig setSessionInfo /id 1 /key C:\private\foo.private.key /cert
C:\certs\foo.cert.pem /ip:192.168.250.1
Operation authority: Administrator
Password:******
13: Operation authority verification error
<TLSSettingTool Folder>>
```

- When operation authority is not set

```
<TLSSettingTool Folder>>tlsconfig setSessionInfo /id 1 /key C:\private\foo.private.key /cert
C:\certs\foo.cert.pem /ip:192.168.250.1

000: Success
<TLSSettingTool Folder>>
```

Specifying Connection Method

You can specify the method to connect to the CPU Unit with the command option.
Use either one of the command options below to specify.

- /usb
  Specify this option for connecting to the USB port of the CPU Unit through direct connection via USB.
  If the CPU Unit does not support USB connection, an error occurs.
- /ip:xxx.xxx.xxx.xxx
  The xxx.xxx.xxx.xxx is the IP address of the connected CPU Unit.
  Specify this option for connecting to an Ethernet port of the CPU Unit through Ethernet connection via a hub or remote connection via USB.

Precautions for Correct Use

Direct connection via Ethernet is not supported.

Execution Result Displays

The execution results of the command are displayed as follows.

- Normal operation
  Commands that have functions to show the execution results display the results.
  000: Success
- Abnormal operation

Commands display the error code and error message.
The error code display format is as follows.
[error code]:[error message]
The [error code] is stored in the Windows environment variable ERRORLEVEL. ERRORLEVEL can be checked with echo %ERRORLEVEL%.

The error codes and error messages are given in the following table.

Error code	Error message	Description
1	Undefined command ; type "tlsconfig help"	Command that does not exist
2	Illegal argument	Incorrect argument
3	Communication error	Communication error with CPU Unit
4	Operating mode error	Command not permitted in RUN mode
5	Session setting already exists	Session setting already exists.
6	Session setting does not exist	Session setting does not exist.
7	Non-supported session ID	Session ID is not supported by the connected CPU Unit.
8	Invalid target file path/name	The path/file name of the specified file is invalid.
9	Target file not found	The specified file does not exist.
10	Output folder already exists	The specified destination folder already exists.
11	Can not create output file	File output failed.
12	Controller execution error	Execution error of CPU Unit processing for the command
13	Operation authority verification error	Operation authority verification error
14	Too large file	The specified file exceeded the maximum file size
15	Client certificate and key do not match or are broken	The client certificate and the private key do not match, or one or both of them are corrupted.

A-10-5 Command Specifications

The specifications of the Secure Socket Configuration commands are described below.

Command	Function	Reference
setSessionInfo	Sets the TLS session information of the specified session ID and registers it to the secure socket setting in the CPU Unit.	page A-78
delSessionInfo	Deletes the TLS session information of the specified session ID from the secure socket setting in the CPU Unit.	page A-80
delAllSessionInfo	Deletes the TLS session information of all session IDs from the secure socket setting in the CPU Unit.	page A-80
getSessionInfo	Reads the TLS session information of the specified session ID from the CPU Unit and displays it. Alternatively, reads the TLS session information and client certificate files from the CPU Unit and saves them in the computer.	page A-81

Command	Function	Reference
getAllSessionInfo	Reads the TLS session information of all session IDs from the CPU Unit, and displays it in order from the smallest session ID number. Alternatively, reads the TLS session information and client certificate files from the CPU Unit and saves them in the computer.	page A-83
setLogLevel	Enables or disables the secure socket communications log.	page A-85
getLogLevel	Reads the enable or disable state of the secure socket communications log.	page A-86
clearAllSettings	Initializes the secure socket setting.	page A-86
help	Displays the version of the Secure Socket Configuration commands and how they are used.	page A-87

setSessionInfo

- Format
  setSessionInfo□/id□n□[/key□xxxx□/cert□xxxx]□[/desc□xxxx]□[/f]□{/usb|/ip:xxx.xxx.xxx.xxx}
- Functions
  You can set TLS session information of the specified session ID to the secure socket setting in the CPU Unit.
  The TLS session information includes the TLS session name, the file name of the client certificate, the file name of the client private key, and a description of the session.
  The TLS session name is automatically set with the session ID specified in the /id option. “TLSSession” + ”session ID” is the session name. If 5 is specified for session ID, the TLS session name is “TLSSession5”.

  You must set client private keys and client certificates only when a server performs client authentication with the X.509 public key certificates.
  When the files of the client certificate and the client private key are specified in the /cert and /key options, the files are transferred from the computer to the CPU Unit.

  You can use the /f option to overwrite and update the client certificate and client private key for the session ID that is already set in the TLS session information.

Additional Information

a) The TLS session name is used as the input variable of SktTLSConnect (Establish TLS Session) instruction.
b) As an alternative method, you can also update the client certificate and client private key by deleting the registered session with the *delSessionInfo* command and then setting the session again with the *setSessionInfo* command.

- Restrictions
  This command can be used only when the CPU Unit is in PROGRAM mode. An error occurs when it is used in RUN mode.
- Option details
  - /id n
    n: Session ID
    This option specifies the session ID in TLS session information to register.

Specify a value from 0 to 59 for an NX102 CPU Unit. For other models, specify 0 to 29.
If the /f option is not specified and the specified session ID is already registered, an error occurs.

- /key xxxx
  xxxx: Path to the client private key file and file name
  Specify the path to the folder on the computer where the client private key file is located. The path also has a file name.
  You can specify either with a relative or an absolute path to the folder.
  If you do not want to set the client private key in the TLS session information, specify "none" in the path or omit the option.
  The TLS session information contains only the file name of the client private key and does not contain any path information.
  An error occurs in the following cases:
  - The client private key file does not exist at the specified location.
  - When the file size of the client private key exceeds 10KB
- /cert xxxx
  xxxx: Path to the client certificate file and the file name
  Specify the path to the folder on the computer where the client certificate file is located. The path also has a file name.
  You can specify either with a relative or an absolute path to the folder.
  If you do not want to set the client certificate in the TLS session information, specify “none” in the path or omit the option.
  The TLS session information contains only the file name of the client certificate and does not contain any path information.
  An error occurs in the following cases:
  - The client certificate file does not exist at the specified location.
  - The size of client certificate file exceeds 10 KB.
- /f
  Even when the client private key and client certificate are already set in the TLS session information of the session ID specified with the /id, they are overwritten with the client private key and client certificate that are specified with the /key and /cert.
  If both of the /key and /cert are not specified, the set client private key and client certificate are deleted.
  If the /desc is not specified, the comments set with the /desc will be cleared.
- /desc
  Description of the session
  You can comment on TLS session information.
  The following characters can be used in comments: 0-9, A-Z, a-z, - (hyphen), _ (underscore), ((parenthesis), and) (closing parenthesis).
  The maximum length is 32 characters.

- Execution results
  None
- Execution examples
  a) When session ID=1, and client certificate and client private key are not set in TLS session information

```
<TLSSettingTool Folder>>tlsconfig setSessionInfo /id 1 /key none /cert none /ip:192.168.250.1
<\TLSSettingTool Folder>>
```

b) When session ID=1, and client certificate = client.crt and client private key = private.key are set in the TLS session information

```
<TLSSettingTool Folder>>tlsconfig setSessionInfo /id 1 /key C:\privates\private.key /cert C:\certs
\client.crt /ip:192.168.250.1
<TLSSettingTool Folder>>
```

delSessionInfo

- Format
  delSessionInfo□/id□n□{/usb|/ipxxx.xxx.xxx.xxx}
- Functions
  You can delete TLS session information of the specified session ID from the secure socket setting in the CPU Unit.
- Restrictions
  This command can be used only when the CPU Unit is in PROGRAM mode. An error occurs when it is used in RUN mode.
- Option details
  - /id n
    n: Session ID
    This option specifies the session ID of TLS session to delete.
    Specify a value from 0 to 59 for an NX102 CPU Unit. For other models, specify 0 to 29.
    If the specified session ID is not already registered in the secure socket setting, an error occurs.
- Execution results
  None
- Execution examples
  When you want to delete TLS session information of session ID=1

```
<TLSSettingTool Folder>>tlsconfig delSessionInfo /id 1 /ip:192.168.250.1
<TLSSettingTool Folder>>
```

delAllSessionInfo

- Format
  delAllSessionInfo□{/usb|/ip:xxx.xxx.xxx.xxx}
- Functions
  You can delete TLS session information of all session IDs from the secure socket setting in the CPU Unit.
  The secure socket communications log enable or disable setting will not be changed.
  When you execute the command, the warning message "This command deletes all session settings of destination Controller. Confirm the settings to be deleted first, and enter "Yes".[Enter]" is displayed.
  If you enter "Yes", all TLS session information is deleted and the message "All settings are deleted." is displayed.
  If an entry other than “Yes” is made, the message "Operation is canceled.” is displayed and TLS session information will not be deleted from the secure socket setting in the CPU Unit.
  Yes is case sensitive. An entry of yes, YES, etc. will cause an error.
- Restrictions

This command can be used only when the CPU Unit is in PROGRAM mode. An error occurs when it is used in RUN mode.

- Option details
  None
- Execution results
  None
- Execution examples
  In the examples below, what it displayed by the command is underlined.
  a) When you delete all TLS session information from the secure socket setting of the CPU Unit (if operation authority is not set)

```
<TLSSettingTool Folder>>tlsconfig delAllSessionInfo /ip:192.168.250.1
This command deletes all session settings of destination Controller.
Confirm the settings to be deleted first, and enter "Yes".
>Yes
All settings are deleted.
000: Success
<TLSSettingTool Folder>>
```

  b) When you cancel the command execution (if operation authority is not set)

```
<TLSSettingTool Folder>>tlsconfig delAllSessionInfo /ip:192.168.250.1
This command deletes all session settings of destination Controller.
Confirm the settings to be deleted first, and enter "Yes".
>No
Operation is canceled.
000: Success
<TLSSettingTool Folder>>
```

getSessionInfo

- Format
  getSessionInfo□/id□n□[/o□path_to_SessionInfo]□[/f]□ {/usb|/ip:xxx.xxx.xxx.xxx}
- Functions
  You can read TLS session information (TLS session name, client certificate file name, client private key file name, and a description of the session) of the specified session ID from the CPU Unit and display it.

  If you specify the /o option, you can save the TLS session information and client certificate file in the specified folder in the computer, instead of displaying them. However, the client private key files cannot be saved.
  The TLS session information is saved in “TLSSessionN.txt” text file (N is the specified session ID).
  The client certificate file is saved with the file name specified when the TLS session information is set.
  The files of TLS session information and client certificate are saved in the folder with the session ID name in the specified folder.
- Restrictions
  None
- Option details
  - /id n
    n: Session ID

Specify the session ID of the TLS session information to be read.
Specify a value from 0 to 59 for an NX102 CPU Unit. For other models, specify 0 to 29.

- /o path_to_SessionInfo
  path_to_SessionInfo: Path to the folder to save the read TLS session information and client certificate files
  If this option is specified, TLS session information is not displayed.
  The path of the folder can be either a relative path or an absolute path.
  Characters and formats that can be used for the specified folder path conform to Windows specifications.
  - The drive letter ("D:" or the like) can be used for the pathname.
  - You must specify the path to the storage in the computer. UNC (Universal Naming Convention) cannot be used.

  An error occurs in the following cases:
  - The specified folder already has a folder with the same name as the session ID specified.
  - You do not have the access right to the specified folder.
    You should be given the appropriate authority or an appropriate folder with the right to access should be specified.
- /f
  If the folder specified with the /o option exists, files in the folder will be deleted, and then the files of TLS session information and client certificate will be saved.
  If the /o option is not specified, specification for this option is ignored.

- Execution results
  TLS session information of the specified session ID is output.
  If the client private key and client certificate are not set, "none" is output.

  - Normal operation

```
Id = session ID
PrivateKey=File name of the private key file
Certificate=File name of the certificate file
Description= Description of the session
(Blank line)
```

  - Abnormal operation
    The following message is output.

```
[error code]:[error message]
```

    The [error code] is stored in the Windows environment variable ERRORLEVEL.

- Execution examples
  a) When you want to display TLS session information of session ID=0

```
<TLSSettingTool Folder>>tlsconfig getSessionInfo /id 0 /ip:192.168.250.1
Id=0
PrivateKey=private.key
Certificate=client.crt
Description=

000: Success
<TLSSettingTool Folder>>
```

  b) When you want to save TLS session information of session ID=2 in C:\Dir1\Dir2 folder

```
<TLSSettingTool Folder>>tlsconfig getSessionInfo /id 2 /o C:\Dir1\Dir2 /ip:192.168.250.1
000: Success
<TLSSettingTool Folder>>
```

TLSSension2.txt and client.crt are saved in C:\Dir1\Dir2\2 folder.
The contents of TLSSession2.txt are as follows.

```
Id=2
PrivateKey=private.key
Certificate=client.crt
Description=
```

getAllSessionInfo

- Format
  getAllSessionInfo□[/o□path_to_SessionInfo]□[/f]□ {/usb|/ip:xxx.xxx.xxx.xxx}
- Functions
  You can read TLS session information (TLS session name, client certificate file name, and client private key file name) of all session IDs from the CPU Unit and display it in order from the smallest session ID number.
  If session ID of 1, 3, 10 and 20 are registered in the TLS session information in the CPU Unit, the TLS session information is displayed in the order of 1, 3, 10 and 20.

  If you specify the /o option, you can save the TLS session information and client certificate file in the specified folder in the computer, instead of displaying them. However, the client private key files cannot be saved.
  The TLS session information is saved in “TLSSessionN.txt” text file (N is the specified session ID).
  The client certificate file is saved with the file name specified when the TLS session information is set.
  The files of TLS session information and client certificate are saved in the folder for each Session ID in the specified folder.
- Restrictions
  None
- Option details
  - /o path_to_SessionInfo
    path_to_SessionInfo: Path to the folder to save the read TLS session information and client certificate files
    TLS session information and client certificate files are stored in the folder with the folder name that has the Session ID followed by the above path.
    If this option is specified, TLS session information is not displayed.
    The path of the folder can be either a relative path or an absolute path.
    Characters and formats that can be used for the specified folder path conform to Windows specifications.
    - The drive letter ("D:" or the like) can be used for the pathname.
    - You must specify the path to the storage in the computer. UNC (Universal Naming Convention) cannot be used.

    An error occurs in the following cases:
    - The specified folder already has a folder with the same name as the session ID specified.

- You do not have the access right to the specified folder.
  You should be given the appropriate authority or an appropriate folder with the right to access should be specified.

- /f
  If the folder specified with the /o option exists, files in the folder will be deleted, and then the files of TLS session information and client certificate will be saved.

- Execution results
  In normal operation, all of the registered TLS session information is output.

```
Count=Serial number
Id = session ID
PrivateKey=”File name of the private key file”
Certificate=”File name of the certificate file”
Description=”Description of the session”
(Blank line)
Count=Serial number
Id = session ID
PrivateKey=”File name of the private key file”
Certificate=”File name of the certificate file”
Description=”Description of the session”
(Blank line)
:
```

  - Serial number
    Outputs the serial number of 1, 2, 3, and… in the order of the session to be output.
  - Session ID
    Outputs session ID (decimal).
  - File name of the private key file
    Outputs the file name set by the setSessionInfo command. The path is not included in file name. If it is not set, “none” is output.
  - File name of the certificate file
    Outputs the file name set by the setSessionInfo command. The path is not included in file name. If it is not set, “none” is output.

- Execution examples
  a) When you want to display TLS session information of all session IDs
     An example when two sessions of session ID=0 and 1 are registered

```
<TLSSettingTool Folder>>tlsconfig getAllSessionInfo /ip:192.168.250.1
Count=1
Id=0
PrivateKey=private0.key
Certificate=client0.crt
Description=
Count=2
Id=1
PrivateKey=private1.key
Certificate=client1.crt
Description=
000: Success
<TLSSettingTool Folder>>
```

  b) When you want to save TLS session information of all session IDs in C:\Dir1\Dir2 folder

```
<TLSSettingTool Folder>>tlsconfig getAllSessionInfo /o C:\Dir1\Dir2 /ip:192.168.250.1
000: Success
<TLSSettingTool Folder>>
```

The read TLS session information and client certificate files are saved in two separate folders: TLSSession0.txt and client0.crt are saved in C:\Dir1\Dir2\0\ folder and TLSSession1.txt and client1.crt are saved in C:\Dir1\Dir2\1\ folder.
The image of the folder is as follows.

```
C:\Dir1\Dir2\
        0\
                TLSSession0.txt
                client0.crt
        1\
                TLSSession1.txt
                client1.crt
```

The contents of TLSSession0.txt are as follows.
Id=0
PrivateKey=private0.key
Certificate=client0.crt
The contents of TLSSession1.txt are as follows.
Id=1
PrivateKey=private1.key
Certificate=client1.crt

setLogLevel

- Format
  setLogLevel□{/enable|/disable}□{/usb|/ip:xxx.xxx.xxx.xxx}
- Functions
  You can set to enable or disable the secure socket communications log.
  Enable: Starts output of secure socket communications log.
  Disable: Stops output of secure socket communications log.
- Restrictions
  This command can be used only when the CPU Unit is in PROGRAM mode. An error occurs when it is used in RUN mode.
- Option details
  /enable: Enables secure socket communications log
  /disable: Disables secure socket communications log
  Specify either of enable or disable.
  Selecting neither or both of the two will result in an error.
- Execution results
  None
- Execution examples
  a) When you enable secure socket communications log

```
<TLSSettingTool Folder>>tlsconfig setLogLevel /enable /ip:192.168.250.1
000: Success
<TLSSettingTool Folder>>
```

b) When you disable secure socket communications log

```
<TLSSettingTool Folder>>tlsconfig setLogLevel /disable /ip:192.168.250.1
000: Success
<TLSSettingTool Folder>>
```

getLogLevel

- Format
  getLogLevel□{/usb|/ip:xxx.xxx.xxx.xxx}
- Functions
  You can read the enable or disable status of secure socket communications log.
  disable: Secure socket communications log is disabled
  enable: Secure socket communications log is enabled
- Restrictions
  None
- Option details
  None
- Execution results
  None
- Execution examples
  a) When the result of reading secure socket communications log status was enable

```
<TLSSettingTool Folder>>tlsconfig getLogLevel /ip:192.168.250.1
enable
000: Success
```

  b) When the result of reading secure socket communications log status was disable

```
<TLSSettingTool Folder>>tlsconfig getLogLevel /ip:192.168.250.1
disable
000: Success
```

clearAllSettings

- Format
  clearAllSettings□{/usb|/ip:xxx.xxx.xxx.xxx}
- Functions
  You can initialize the secure socket setting.
  - This command clears all TLS session information.
  - This command disables secure socket communications log (stops output).
- Restrictions
  This command can be used only when the CPU Unit is in PROGRAM mode. An error occurs when it is used in RUN mode.
- Option details
  None
- Execution results

None

- Execution examples

  In the examples below, what it displayed by the command is underlined.

  a) To execute the command

```
<TLSSettingTool Folder>>tlsconfig clearAllSettings /ip:192.168.250.1
This command clears all settings of destination Controller.
Confirm the settings to be cleared first, and enter "Yes".
>Yes
All settings are cleared.
000: Success
<TLSSettingTool Folder>>
```

  b) To cancel command execution

```
<TLSSettingTool Folder>>tlsconfig clearAllSettings /ip:192.168.250.1
This command clears all settings of destination Controller.
Confirm the settings to be cleared first, and enter "Yes".
>No
Operation is canceled.
000: Success
<TLSSettingTool Folder>>
```

help

- Format

  help

- Functions

  You can display the version information and how to use the commands.

- Restrictions

  None

- Option details

  None

- Execution results

  None

- Execution examples

```
<TLSSettingTool Folder>>tlsconfig help
Secure Socket Configuration Tool
tlsconfig.exe Version 1.00.00
Copyright (c) OMRON Corporation 2021. All Rights Reserved.
Usage: tlsconfig command [option1,option2....]
command:
setSessionInfo /id n [/key KEY_FILE_NAME /cert CERT_FILE_NAME] /desc [DESCRIPTION] [/f]
{/usb|/ip:IP_ADDRESS}
delSessionInfo /id n {/usb|/ip:IP_ADDRESS}
delAllSessionInfo {/usb|/ip:IP_ADDRESS}
getSessionInfo /id n [/o path_to_SessionInfo] [/f] {/usb|/ip:IP_ADDRESS}
getAllSessionInfo [/o path_to_SessionInfo] [/f] {/usb|/ip:IP_ADDRESS}
setLogLevel {/enable|/disable} {/usb|/ip:IP_ADDRESS}
getLogLevel {/usb|/ip:IP_ADDRESS}
clearAllSettings {/usb|/ip:IP_ADDRESS}
help
<TLSSettingTool Folder>>
```

When Commands are Omitted

- Format
  None
- Functions
  Same as the help command.
- Restrictions
  None
- Option details
  None
- Execution results
  None
- Execution examples

A-11 Version Information

This appendix shows the supported functions which have been changed or added through version upgrades of the CPU Units.

● Additions and Changes to Functional Specifications

The following table lists additions and changes to the functional specifications, each with the corresponding CPU unit version and Sysmac Studio version.

Function		Addition/ Change	Reference	Unit version	Sysmac Studio version
CIP routing		Addition	page 1-20	1.01	1.02
Packet Filter (Simple)		Addition	page 4-9	1.30	1.23
Packet Filter		Addition/ Change	page 4-7	*1	1.50
Support for mounting a CJ1W-EIP EtherNet/IP Unit		Addition	page 1-6	1.01	1.02
Offsets for structure members	Optional	Addition	page A-57		
	CJ	Addition	page A-57	1.02	1.03
CIP objects	Identity object	Change	page 7-49	1.01	---
	NX Configuration object	Addition	page 7-52	1.30	---
	TCP/IP Interface object	Change	page 7-73	1.02	---
Tag data links	Packet intervals (RPI)	Change	page 6-6	1.03	1.04
	Permissible communications band	Change			
CIP message communications	CIPOpenWithDataSize instruction	Addition	page 7-4	1.06	1.07
	Client function	Addition*2	page 7-16	1.11	1.15
Socket services	Number of supported sockets	Change	page 8-10	1.03	1.04
	SktSetOption instruction	Addition	page 8-12	1.12	1.16
	TCP/UDP message	Addition	page 8-32	1.30	1.23
	Secure socket services	Addition	page 8-35	*3	1.46
FTP client		Addition	page 11-1	1.08	1.09
Troubleshooting	Tag Data Link Connection Timeout	Addition	*4	1.04	1.05
	Number of Tag Sets for Tag Data Links Exceeded	Addition	*4	1.30	1.23
Connection settings		Addition	page A-4	1.09	1.10
TCP/IP settings	Operation for an IP address conflict	Addition	page 4-2	*5	
Modbus TCP Master Function		Addition	page 9-1	1.30	1.23

*1. Refer to *Packet Filter* on page 4-7 for the CPU Unit models and unit versions that support the Packet Filter.

*2. An extension structure is supported as the data type of variables to contain the request path (IOI).
*3. Refer to *8-9 Secure Socket Services* on page 8-35 for the CPU Unit models and unit versions that support the secure socket services.
*4. Refer to *NJ/NX-series Troubleshooting Manual (Cat. No. W503)*.
*5. This function can be used with the Sysmac Studio and CPU Units which support OPC UA. Refer to the *NJ/NX-series CPU Unit OPC UA User's Manual (Cat. No. W588)* for information on the models and unit versions of the CPU Units that support OPC UA, and the corresponding Sysmac Studio versions.

I

Index

Index

Numerics

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

I

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 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 ASIA PACIFIC PTE. LTD.
438B Alexandra Road, #08-01/02 Alexandra
Technopark, Singapore 119968
Tel: (65) 6835-3011 Fax: (65) 6835-2711

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:

Cat. No. W506-E1-31 0123