OMRON

Machine Automation Controller NJ-series

Startup Guide for CPU Unit

SYSMAC always in control

W513-E1-01

Introduction

Thank you for purchasing an NJ-series CPU Unit and the Sysmac Studio.

This *NJ-series Startup Guide for CPU Unit* (hereafter referred to as “this Guide”) describes the startup procedures that are required to use an NJ-series CPU Unit for the first time and the basic operating instructions for the Sysmac Studio. A simple sequence control example is used for the discussion. You can perform the procedures that are presented in this Guide to quickly gain a basic understanding of the NJ-series CPU Units and the Sysmac Studio. This Guide contains information about useful NJ-series CPU Unit and Sysmac Studio features.

This Guide does not contain safety information and other details that are required for actual use of an NJ-series Controller. Thoroughly read and understand the manuals for all of the devices that are used in this Guide to ensure that the system is used safely. Review the entire contents of these materials, including all safety precautions, precautions for safe use, and precautions for correct use.

For the startup and operating instructions for motion control, refer to the *NJ-series Startup Guide for Motion Control* (Cat. No. W514).

Intended Audience

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

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

Applicable Products

This Guide covers the following products.

- CPU Units of NJ-series Machine Automation Controllers
- Sysmac Studio Automation Software

Special Information

The icons that are used in this Guide are described below.

 **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.

Read and Understand this Manual

Please read and understand this manual before using the product. Please consult your OMRON representative if you have any questions or comments.

CPU Units of NJ-series Machine Automation Controllers

Warranty and Limitations of Liability

WARRANTY

OMRON's exclusive warranty is that the products are free from defects in materials and workmanship for a period of one year (or other period if specified) from date of sale by OMRON.

OMRON MAKES NO WARRANTY OR REPRESENTATION, EXPRESS OR IMPLIED, REGARDING NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR PARTICULAR PURPOSE OF THE PRODUCTS. ANY BUYER OR USER ACKNOWLEDGES THAT THE BUYER OR USER ALONE HAS DETERMINED THAT THE PRODUCTS WILL SUITABLY MEET THE REQUIREMENTS OF THEIR INTENDED USE. OMRON DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED.

LIMITATIONS OF LIABILITY

OMRON SHALL NOT BE RESPONSIBLE FOR SPECIAL, INDIRECT, OR CONSEQUENTIAL DAMAGES, LOSS OF PROFITS OR COMMERCIAL LOSS IN ANY WAY CONNECTED WITH THE PRODUCTS, WHETHER SUCH CLAIM IS BASED ON CONTRACT, WARRANTY, NEGLIGENCE, OR STRICT LIABILITY.

In no event shall the responsibility of OMRON for any act exceed the individual price of the product on which liability is asserted.

IN NO EVENT SHALL OMRON BE RESPONSIBLE FOR WARRANTY, REPAIR, OR OTHER CLAIMS 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 OR REPAIR.

Application Considerations

SUITABILITY FOR USE

OMRON shall not be responsible for conformity with any standards, codes, or regulations that apply to the combination of products in the customer's application or use of the products.

At the customer's request, OMRON will provide applicable third party certification documents identifying ratings and limitations of use that apply to the products. This information by itself is not sufficient for a complete determination of the suitability of the products in combination with the end product, machine, system, or other application or use.

The following are some examples of applications for which particular attention must be given. This is not intended to be an exhaustive list of all possible uses of the products, nor is it intended to imply that the uses listed may be suitable for the products:

- Outdoor use, uses involving potential chemical contamination or electrical interference, or conditions or uses not described in this manual.
- Nuclear energy control systems, combustion systems, railroad systems, aviation systems, medical equipment, amusement machines, vehicles, safety equipment, and installations subject to separate industry or government regulations.
- Systems, machines, and equipment that could present a risk to life or property.

Please know and observe all prohibitions of use applicable to the products.

NEVER USE THE PRODUCTS FOR AN APPLICATION INVOLVING SERIOUS RISK TO LIFE OR PROPERTY WITHOUT ENSURING THAT THE SYSTEM AS A WHOLE HAS BEEN DESIGNED TO ADDRESS THE RISKS, AND THAT THE OMRON PRODUCTS ARE PROPERLY RATED AND INSTALLED FOR THE INTENDED USE WITHIN THE OVERALL EQUIPMENT OR SYSTEM.

PROGRAMMABLE PRODUCTS

OMRON shall not be responsible for the user's programming of a programmable product, or any consequence thereof.

Disclaimers

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 model numbers when published ratings or features are changed, or when significant construction changes are made. However, some specifications of the products may be changed without any notice. When in doubt, special model numbers may be assigned to fix or establish key specifications for your application on your request. Please consult with your OMRON representative at any time to confirm actual specifications of purchased products.

DIMENSIONS AND WEIGHTS

Dimensions and weights are nominal and are not to be used for manufacturing purposes, even when tolerances are shown.

PERFORMANCE DATA

Performance data given in this manual 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 users must correlate it to actual application requirements. Actual performance is subject to the OMRON Warranty and Limitations of Liability.

ERRORS AND OMISSIONS

The information in this manual has been carefully checked and is believed to be accurate; however, no responsibility is assumed for clerical, typographical, or proofreading errors, or omissions.

Sysmac Studio Automation Software

● Warranty

(1) Warranty Period
The warranty period for this software is one year from either the date of purchase or the date on which the software is delivered to the specified location.

(2) Scope of Warranty

a) Customers who agree to the terms of use for this software and discover a defect in the software (a significant difference from the information that is provided in the *Sysmac Studio Version 1 Operation Manual* (Cat. No. W504)) can return their copy of the software to OMRON for a replacement copy of the software without the defect. OMRON may also elect to provide a method to download a copy of the software without the defect from an OMRON website. If a problem is discovered with the storage media containing the software and the media is returned to OMRON, OMRON shall provide a replacement storage media containing the software free of charge.

b) If OMRON is unable to eliminate the defect from the software for any reason, OMRON shall return the amount paid for the software to the customer.

● Limitations of Liability

(1) The purchase price refund and exchange defined in the preceding article represent the limits of the warranty for this software. OMRON shall not be held responsible for any direct, indirect, or consequential damages or losses to the customer as a result of any defect in this software.

(2) OMRON shall not be held responsible for any defects resulting in the modification of this software by any party other than OMRON.

(3) OMRON shall not be held responsible for software developed based on this software by any party other than OMRON or for the results of that software.

● Application of the Software

Do not use this software for any purpose other than those described in the *Sysmac Studio Version 1 Operation Manual* (Cat. No. W504).

● Changes to Specifications

Specifications and accessories for this software may be changed as needed to improve the product or for any other reason.

● Scope of Service

The price of this software does not include service costs, such as dispatching technical staff.

Precautions

- When building a system, check the specifications for all devices and equipment that will make up the system and make sure that the OMRON products are used well within their rated specifications and performances. Safety measures, such as safety circuits, must be implemented in order to minimize the risks in the event of a malfunction.
- Thoroughly read and understand the manuals for all devices and equipment that will make up the system to ensure that the system is used safely. Review the entire contents of these materials, including all safety precautions, precautions for safe use, and precautions for correct use.
- Confirm all regulations, standards, and restrictions that the system must adhere to.

Trademarks

- Sysmac and SYSMAC are trademarks or registered trademarks of OMRON Corporation in Japan and other countries for OMRON factory automation products.
- Windows, Windows 98, Windows XP, Windows Vista, and Windows 7 are registered trademarks of Microsoft Corporation in the USA and other countries.
- Intel, the Intel logo, and Intel Atom are the trademarks of Intel Corporation in the USA and other countries.
- EtherCAT® is registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.
- The SD logo is a trademark of SD-3C, LLC.

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

Software Licenses and Copyrights

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/.

Related Manuals

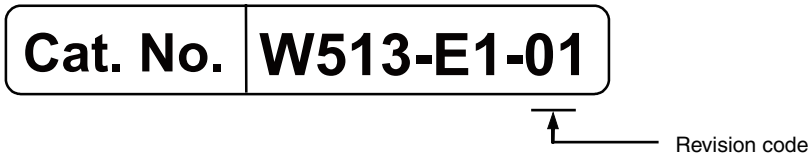
The following manuals are related to the NJ-series Controllers. Use these manuals for reference.

Manual name	Cat. No.	Model numbers	Application	Description
NJ-series CPU Unit Hardware User's Manual	W500	NJ501-□□□□	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 a Controller built with an NJ501 CPU Unit. • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection Use this manual together with the *NJ-series CPU Unit Software User's Manual* (Cat. No. W501).
NJ-series CPU Unit Software User's Manual	W501	NJ501-□□□□	Learning how to program and set up an NJ-series CPU Unit. Mainly software information is provided.	The following information is provided on a Controller built with an NJ-series CPU Unit. • CPU Unit operation • CPU Unit features • Initial settings • Programming based on IEC 61131-3 language specifications Use this manual together with the *NJ-series CPU Unit Hardware User's Manual* (Cat. No. W500).
NJ-series Instructions Reference Manual	W502	NJ501-□□□□	Learning detailed specifications on the basic instructions of an NJ-series CPU Unit.	The instructions in the instruction set (IEC 61131-3 specifications) are described. When programming, use this manual together with the *NJ-series CPU Unit Hardware User's Manual* (Cat. No. W500) and *NJ-series CPU Unit Software User's Manual* (Cat. No. W501).
NJ-series CPU Unit Built-in EtherCAT Port User's Manual	W505	NJ501-□□□□	Using the built-in EtherCAT port on an NJ-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. Use this manual together with the *NJ-series CPU Unit Hardware User's Manual* (Cat. No. W500) and *NJ-series CPU Unit Software User's Manual* (Cat. No. W501).

Manual name	Cat. No.	Model numbers	Application	Description
GX-series EtherCAT Slave Units User's Manual	W488	GX-ID□□□□ GX-OD□□□□ GX-OC□□□□ GX-MD□□□□ GX-AD□□□□ GX-DA□□□□ GX-EC□□□□ XWT-ID□□ XWT-OD□□	Leaning how to connect GX-series EtherCAT Slave Units.	Provides the specifications of and describes application methods for GX-series EtherCAT Slave Units.
NJ-series Troubleshooting Manual	W503	NJ501-□□□□	Learning about the errors that may be detected in an NJ-series Controller.	Concepts on managing errors that may be detected in an NJ-series Controller and information on individual errors are described. Use this manual together with the *NJ-series CPU Unit Hardware User's Manual* (Cat. No. W500) and *NJ-series CPU Unit Software User's Manual* (Cat. No. W501).
Sysmac Studio Version 1 Operation Manual	W504	SYSMAC -SE2□□□	Learning about the operating procedures and functions of the Sysmac Studio.	The operating procedures of the Sysmac Studio are described.

Revision History

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



Revision code	Date	Revised content
01	November 2011	Original production

Features of an NJ-series CPU Unit

The SYSMAC NJ-series Controllers are next-generation machine automation controllers that provide the functionality and high-speed performance that are required for machine control. They provide the safety, reliability, and maintainability that are required of industrial controllers.

The NJ-series Controllers provide the functionality of previous OMRON PLCs, and they also provide the functionality that is required for motion control. Synchronized control of I/O devices on high-speed EtherCAT can be applied to vision systems, motion equipment, discrete I/O, and more.

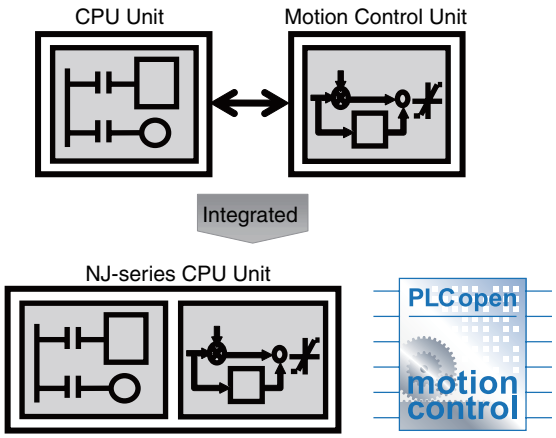
High-speed, High-precision Control

An Intel® Atom™ processor enables high-speed execution of the user program. Ladder diagram instructions are executed as quickly as 1.9 ns. Double-precision floating-point calculation instructions are executed as quickly as 26 ns.

Integrated Sequence Control and Motion Control

An NJ-series CPU Unit can perform both sequence control and motion control. This eliminates the communications time loss between the CPU Unit and the Position Control Units and simplifies the communications setup.

Furthermore, a diverse, PLCopen-compliant Motion Control Function Block provides the ability to perform complex cam motion and multi-axis coordinated control operations.

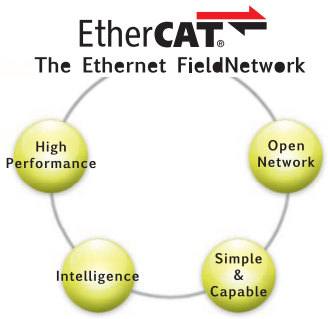


Standard-feature EtherCAT Control Network Support

All CPU Units provide an EtherCAT master port for EtherCAT communications.

EtherCAT is an advanced industrial network system that achieves faster, more-efficient communications. It is based on Ethernet. Each node achieves a short fixed communications cycle time by transmitting Ethernet frames at high speed.

Standard-feature EtherCAT allows you to connect all of the devices required for machine control (e.g., I/O systems, Servo Drives, Inverters, and machine vision) to the same network.



Standard-feature EtherNet/IP Communications Port

All CPU Units provide an EtherNet/IP port for EtherNet/IP communications. EtherNet/IP is a multivendor industrial network that uses Ethernet. You can use it for networks between Controllers or as a field network. The use of standard Ethernet technology allows you to connect to many different types of general-purpose Ethernet devices.

Programming Languages Based on the IEC 61131-3 International Standard

The NJ-series Controllers support language specifications that are based on IEC 61131-3. To these, OMRON has added our own improvements.

You can use motion control instructions that are based on PLCopen standards and an instruction set (POUs) that follows IEC standards.

Programming with Variables to Eliminate Worrying about the Memory Map

You access all data through variables in the same way as for the advanced programming languages that are used on computers.

When you create variables, memory in the CPU Unit is automatically assigned to them so that you do not have to be concerned with the memory map. This keeps software modifications to a minimum even when there are changes in the hardware configuration.

Multitasking

Series of processes including I/O refreshing and program execution are assigned to tasks, and then the execution conditions and the order of execution are set for the tasks. Many tasks can be combined to flexibly build a control system that is matched to the application.

A Wealth of Security Features

You can use the many security features of the NJ-series Controllers, including operation authority settings and restriction of program execution with IDs.

Complete Controller Monitoring

The CPU Unit monitors events in all parts of the Controller, including mounted Units and EtherCAT slaves. Troubleshooting methods for errors that are generated as events are displayed on the Sysmac Studio or on an NS-series PT. Events are also recorded in logs.

Sysmac Studio Automation Software

The Sysmac Studio provides an integrated development environment that covers not only the Controller, but also covers peripheral devices and devices on EtherCAT. You can use integrated operations regardless of the device. You can use the Sysmac Studio in all phases of Controller application, from designing through debugging, simulations, commissioning, and changes during operation.

A Wealth of Simulation Features

The many simulation features include execution, debugging, and task execution time estimates on a virtual controller.

CONTENTS

Section 4 Programming and Debugging

Section 5 Useful Functions

Appendices

1

System Configuration and Startup Procedures

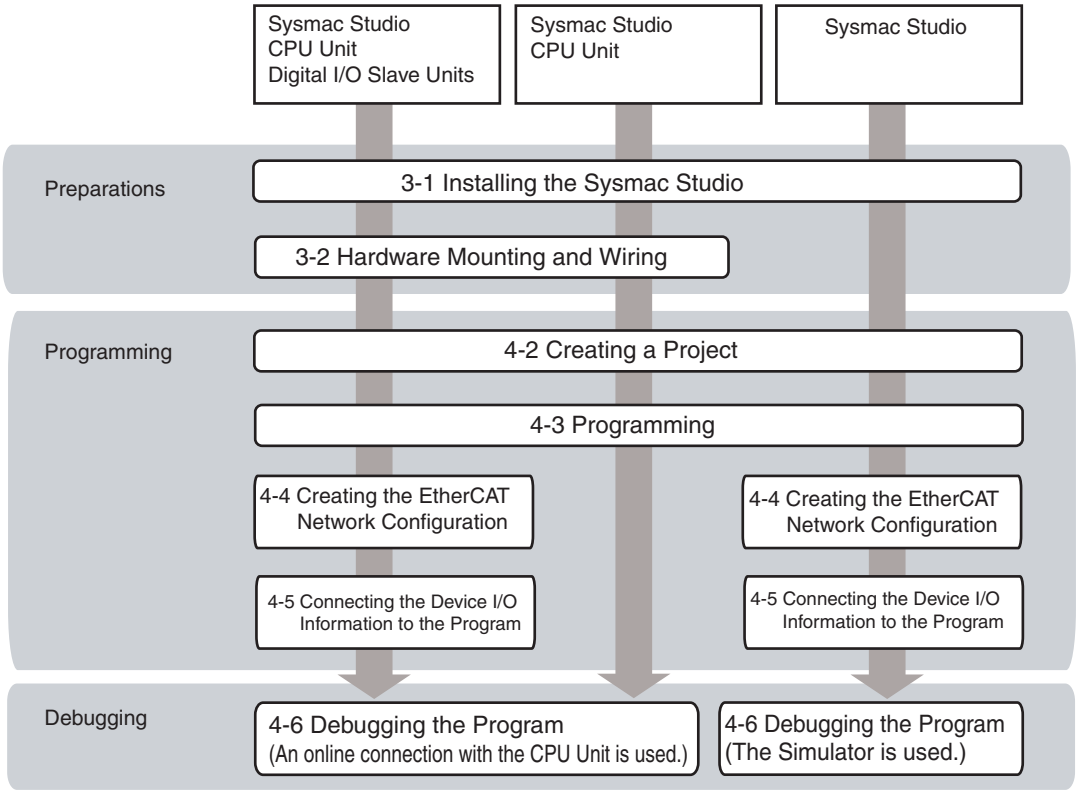
This section describes the startup procedures that are presented, the system configuration that is used, and the operation of the program that is created in this Guide.

1-1 Startup Procedures

This Guide describes the basic procedures from programming to debugging. All operations from programming to debugging can be performed under the following system configurations.

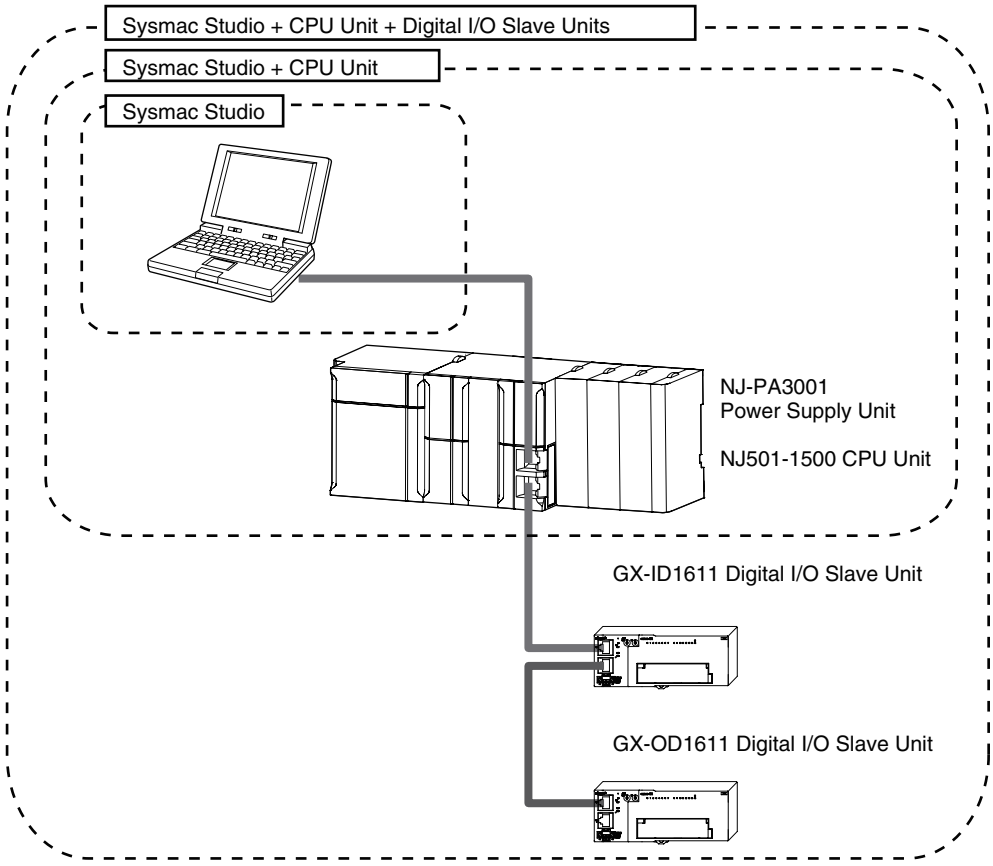
System configuration	Operation details
• Sysmac Studio • CPU Unit • Digital I/O Slave Units	You create the program on the Sysmac Studio, go online with the CPU Unit, and then debug the program. During programming, you create the program and connect the I/O information for the Digital I/O Slave Units with the program.
• Sysmac Studio • CPU Unit	You create the program on the Sysmac Studio, go online with the CPU Unit, and then debug the program. Digital I/O Slave Units are not used, so programming involves only creating the program.
• Sysmac Studio	You perform all tasks from programming to debugging by using just the Sysmac Studio. During programming, you create the program and connect the I/O information for the Digital I/O Slave Units with the program. Debugging is performed with the Simulator. The Simulator allows you to debug on the computer a program that was created for the actual system configuration.

The startup procedures for each of these system configurations is outlined in the following figure.



1-2 System Configuration and Configuration Devices

The following figure represents the system configurations that are used in this Guide. If a CPU Unit is used, connect a computer with the Sysmac Studio installed to the peripheral USB port on the CPU Unit. If Digital I/O Slave Units are used, connect the Digital I/O Slave Units to the CPU Unit's built-in EtherCAT port.



● Configuration Devices

The models of the devices that are used in each of the above system configurations are given below. When selecting devices for an actual application, refer to the device manuals.

Device name	Model numbers	Manual name
NJ-series CPU Unit	NJ501-1500 (unit version 1.00)	NJ-series CPU Unit Hardware User's Manual (Cat. No. W500)
NJ-series Power Supply Unit	NJ-PA3001	
EtherCAT Communications Cables	XS5W-T421-CMD-K	
Digital I/O Slave Units	GX-ID1611 (version 1.1) and GX-OD1611 (version 1.1)	GX-series EtherCAT Slave Units User's Manual (Cat. No. W488)
Unit power supplies*1	S8JX Series	---
USB cable	Commercially available USB cable*2	

*1. The Unit power supplies are used for the Digital I/O Slave Units.

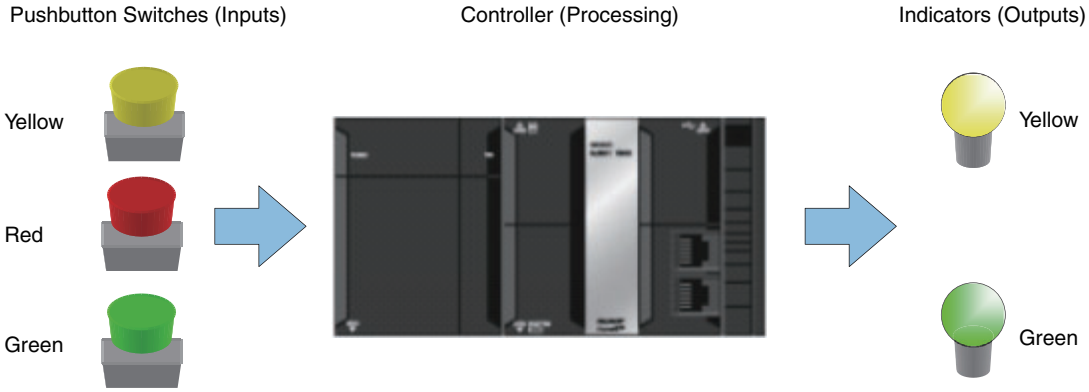
*2. Use a USB 2.0 (or 1.1) cable (A connector - B connector), 5.0 m max.

● Automation Software

Product	Number of licenses	Model numbers
Sysmac Studio Standard Edition version 1.00	None (DVD only)	SYSMAC-SE200D
	1 license	SYSMAC-SE201L

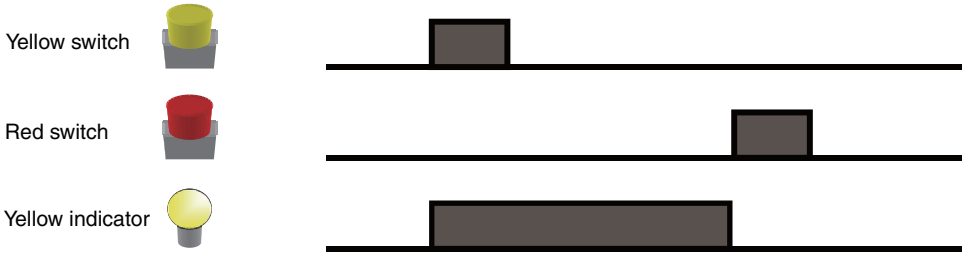
1-3 User Program to Create

The startup procedures are used to create a program for sequence control with switches and indicators. This programming example assumes that pushbutton switches are wired as inputs to a Digital I/O Slave Unit and that indicators are wired to a Digital I/O Slave Unit as outputs. The switches control the indicators. However, because we only need to confirm the operation of the variables in the program, you do not need to prepare any actual switches or indicators for this example.



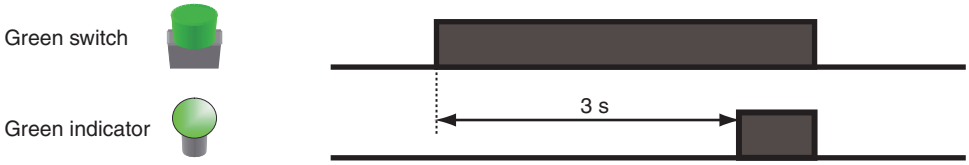
Controlling the Yellow Indicator (Self-holding Rung)

- When the yellow pushbutton switch is turned ON, the yellow indicator lights.
- When the yellow pushbutton switch is turned OFF, the yellow indicator does not go out.
- When the red pushbutton switch is turned ON, the yellow indicator goes out.



Controlling the Green Indicator (ON-delay Timer)

- When the green pushbutton switch is turned ON, the green indicator lights 3 seconds later.
- When the green pushbutton switch is turned ON and then turned OFF again within 3 seconds, the green indicator does not light.
- When the green pushbutton switch is turned OFF while the green indicator is lit, the green indicator goes out.



2

Fundamentals of Programming

This section describes the fundamental elements of programming an NJ-series CPU Unit: POUs, variables, tasks, programming languages, and instructions.

2-1 POUs (Program Organization Units)

A POU (program organization unit) is a unit that is defined in the IEC 61131-3 user program execution model. You combine POUs to build a complete user program.

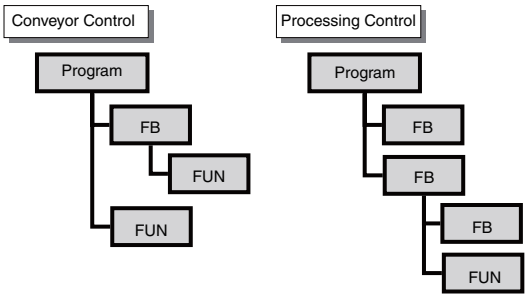
There are three types of POUs, as described below.

POU configuration element	Description
Programs	A program corresponds to a main routine. It is the main type of POU that is used for algorithms. You can place any instruction, function, or function block in the algorithm of a program.
Function Blocks (FBs)	A function block can output different values even with the same inputs. Function blocks are executed when they are called from a program or another function block. To use a function block in a program, an instance of the function block must be placed in the program. You can retain the values of internal variables. Therefore, you can retain status, such as for timers and counters.
Functions (FUNs)	A function always outputs the same values for the same inputs. Functions are executed when they are called from a program, another function, or a function block.

● Easy-to-Read Programming

Programs can be organized in layers by calling POUs from other POUs. For example, you can increase the readability of your programs by structuring them according to units of control.

The following figure shows the structure for a conveyor control program and processing control program as examples.

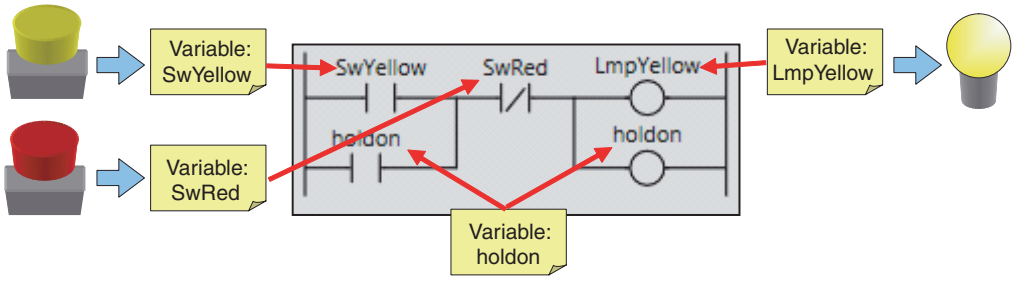


● Reusable Programming

Function blocks and functions are used to divide programs into smaller, more manageable objects. If processes are divided up into function blocks, you can call instances of those function blocks to reuse them in other devices that require those same processes.

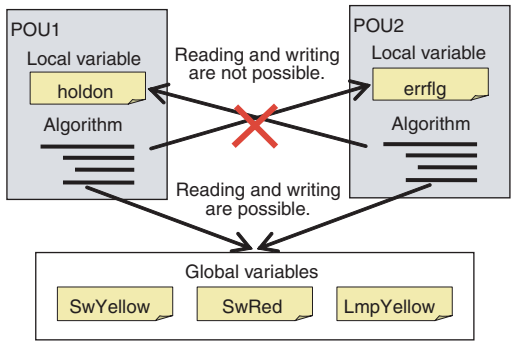
2-2 Variables

Variables store I/O data for exchange with external devices or temporary data that is used for internal POU processing. In the NJ-series System, variables are used to exchange I/O information with external devices, to perform data calculations, and to perform other processes. This allows for programming that does not depend on hardware memory addresses.



● Local Variables and Global Variables

You can read and write a local variable only from the POU in which you defined it. However, you can read and write global variables from any POU (i.e., any program, function, or function block). This Guide defines the variables that are required to access the pushbutton switches and indicators as global variables. The variables that are used in self-holding rungs are defined as local variables.



● Variable Data Types

The Data Type attribute defines the type of data and range of data that are expressed by a variable. To define a variable, you must specify its data type.

NJ-series Controllers provide the predefined basic data types that are listed in the following table. There are also derivative data types, which are defined by the user.

Classification	Definition	Data types
Boolean	A data type with a value of either TRUE or FALSE.	BOOL
Bit strings	A data type that represents a value as a bit string.	BYTE, WORD, DWORD, and LWORD
Integers	A data type that represents an integer value.	SINT, INT, DINT, LINT, USINT, UINT, UDINT, and ULINT
Real numbers	A data type that represents a real number.	REAL and LREAL
Durations	A data type that represents a time duration (days, hours, minutes, seconds, and milliseconds).	TIME
Times of day	A data type that represents a specific time of day (hour, minutes, and seconds).	DATE
Dates	A data type that represents a date (year, month, and day).	TIME_OF_DAY
Dates and times	A data type that represents a date and time (year, month, day, hour, minutes, seconds, and milliseconds).	DATE_AND_TIME
Text strings	A data type that contains a value that represents a text string.	STRING

2-3 Tasks

Tasks are used to assign execution conditions and execution priorities to programs and to I/O refreshing. Programs are executed by assigning them to a task.

There are two types of tasks, as described in the following table.

Type of task	Definition
Primary periodic task	The primary periodic task is executed once every task period. The primary periodic task has the highest execution priority. It executes processes with high speed and high precision.
Periodic task	A periodic task is executed once every task period. These tasks have a lower execution priority than the primary periodic task. Periodic tasks are executed during the unused time between executions of the primary periodic task.

The program in this Guide is executed in the primary periodic task. The Sysmac Studio default value of 1 ms is used as the task period. This setting does not need to be changed.

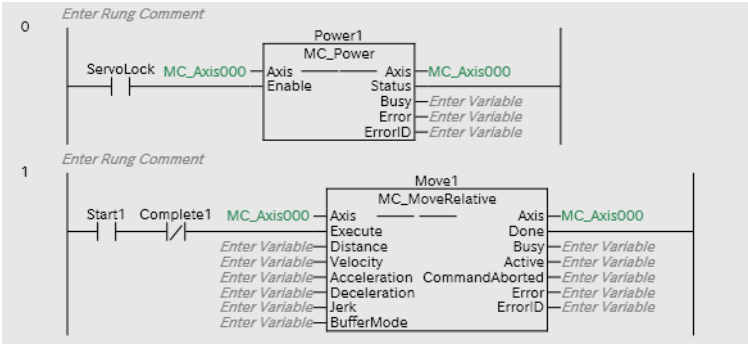
2-4 Programming Languages

The languages that are used to express the algorithms in POUs are called the programming languages.

There are two different programming languages that you can use for an NJ-series Controller: ladder diagram language (LD) and structured text language (ST).

● Ladder Diagrams

The ladder diagram language (LD) is a graphical programming language that is written in a form that appears similar to electrical circuits. Each object for processing, including functions and function blocks, is represented as a diagram. Those objects are connected together with lines to build the algorithm. Algorithms that are written in the ladder diagram language are called ladder diagrams.



● Structured Text

The ST language is a high-level language for industrial controls (mainly PLCs). It is defined by the IEC 61131-3 standard.

The standard control statements, operators, and functions make the ST language ideal for mathematical processing, which is difficult to write in ladder diagrams.

```
1
2       // Determine TableNo
3       FOR i:=0 TO ItemNum DO
4
5           IF (MinNo[i] <= ItemBox[i]) AND (ItemBox[i] <= MaxNo[i]) THEN   // Normal
6               TableNo[i] := ItemBox[i];
7               RangeOK[i] := TRUE;
8
9           ELSIF (ItemBox[i] > MaxNo[i]) THEN                              // Upper
10              TableNo[i] := MaxNo[i];
11              RangeOK[i] := FALSE;
12
13          ELSE                                                            // Lower
14              TableNo[i] := MinNo[i];
15              RangeOK[i] := FALSE;
16
17          END_IF;
18
19      END_FOR;
```

The programming examples in this Guide use ladder diagrams because they are ideal for sequence control.

2-5 Instructions

Instructions are the smallest unit of the processing elements that are provided by OMRON for use in POU algorithms.

Instructions are classified as shown below.

Type of instruction	Definition
Ladder diagram structure elements (inputs and outputs)	These instructions take the form of a program input or program output. These are used in ladder diagrams.
FB instructions	These instructions are created with a function block. An instruction that retains status within the instruction, such as timers or counters, is written as an FB instruction. To use an FB instruction in a program, an instance of the instruction must be placed in the program.
FUN instructions	These instructions are created with a function. Instructions that always output the same values for the same inputs are written as function instructions.
ST language statements	These include instructions that take the form of ST constructs, such as IF or FOR constructs, and simple ST statements, such as assignments and operators. These instructions are used in the ST language.

3

Before You Begin

This section describes the installation of the Sysmac Studio and the process of hardware mounting and wiring.

3-1 Installing the Sysmac Studio

The Sysmac Studio is the Support Software that you use for an NJ-series Controller. On it, you can create the Controller configuration and settings, you can write the programs, and you can debug and simulate operation.

Use the following procedure to install the Sysmac Studio.

1. Set the Sysmac Studio installation disk into the DVD-ROM drive.

   The setup program is started automatically and the Select Setup Language Dialog Box is displayed.

2. Select the language to use, and then click the **OK** Button.

   The Sysmac Studio Setup Wizard is started.

3. Follow the instructions given by the Setup Wizard to complete the installation.

4. Restart the computer when the installation is completed.

Additional Information

- The system requirements for the Sysmac Studio are given in the following table.

OS	CPU		RAM	Display
Windows XP SP3, Windows Vista, or Windows 7 (32-bit or 64-bit edition)	Minimum	Personal computer with Celeron 540 (1.8 GHz) processor	2 GB	XGA 1,024 × 768, 16 million colors
	Recommended	Personal computer with Core i5 M520 (2.4 GHz) processor or the equivalent	2 GB	WXGA 1,280 × 800, 16 million colors

- Refer to the *Sysmac Studio Version 1 Operation Manual* (Cat. No. W504) if you are unable to install the Sysmac Studio with the above instructions.

Precautions for Correct Use

If CX-One version 4 or lower is installed, the installation is cancelled and the Sysmac Studio cannot be installed. In that case, uninstall the CX-One before you install the Sysmac Studio.

3-2 Assembling the Hardware

Connect and wire all of the devices that are used in the system configuration.

This section gives an overview of the assembly procedures. Refer to the device manuals for detailed procedures and safety precautions.

Precautions for Safe Use

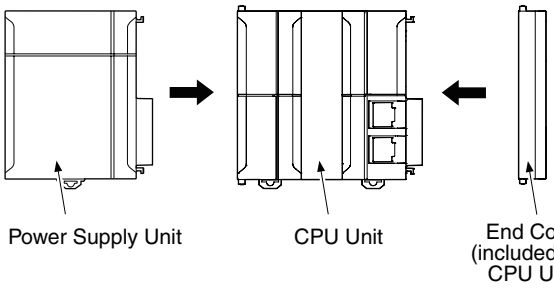
Always turn OFF the power supply to the Controller and the Units before you attempt any of the following.

- Mounting or removing any Units
- Assembling a Rack
- Setting DIP switches or rotary switches
- Connecting cables or wiring the system
- Connecting or disconnecting the connectors

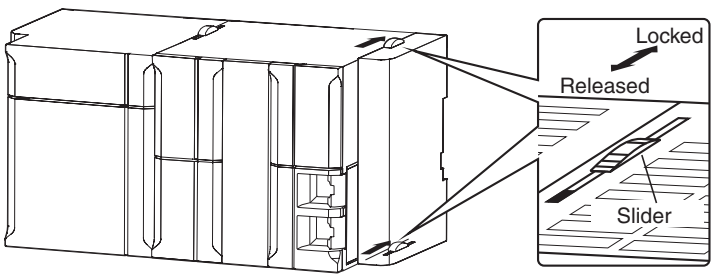
The Power Supply Unit continues to supply power to the Controller for up to several seconds after the power supply is turned OFF. The PWR indicator remains lit as long as power is supplied. Make sure that the PWR indicator is not lit before you perform any of the above operations.

3-2-1 Mounting the Units

Connect the Power Supply Unit, CPU Unit, and End Cover.

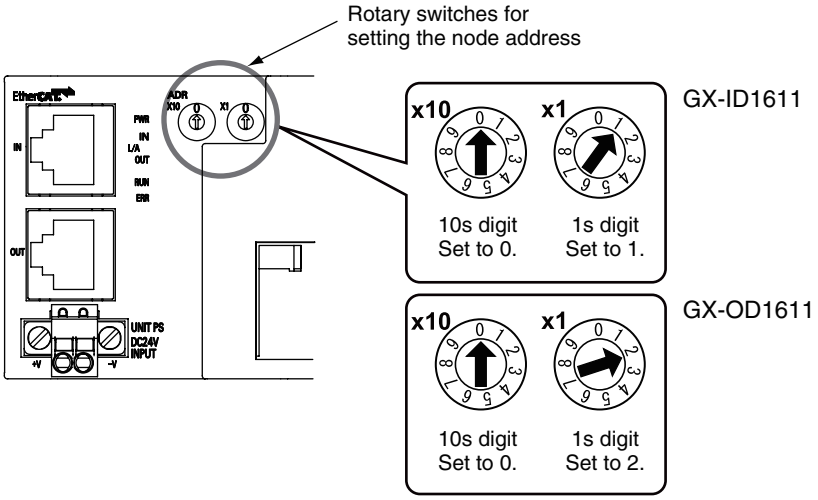


After joining the connectors between the Units, use the sliders at the top and bottom of each Unit to lock the Units together. Lock the sliders firmly into place.



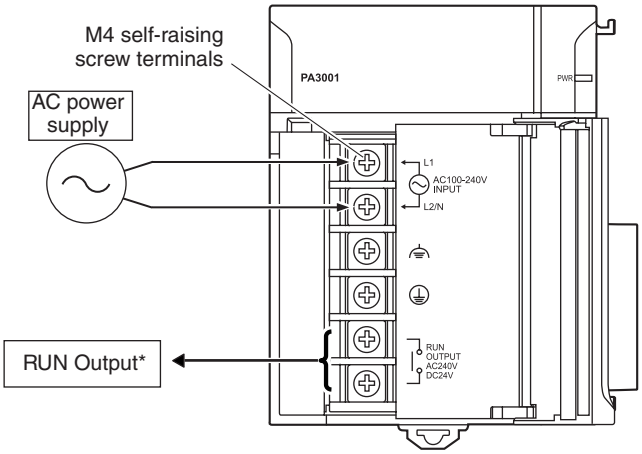
3-2-2 Setting the Node Address for the Digital I/O Slave Units

Set the node address for the Digital I/O Slave Units as shown below.



3-2-3 Wiring the Power Supply

Wire the Power Supply Unit to the power supply.



* The RUN output is ON when the CPU Unit is in RUN mode.
It is OFF when the CPU Unit is in PROGRAM mode or when a major fault level Controller error occurs.

Additional Information

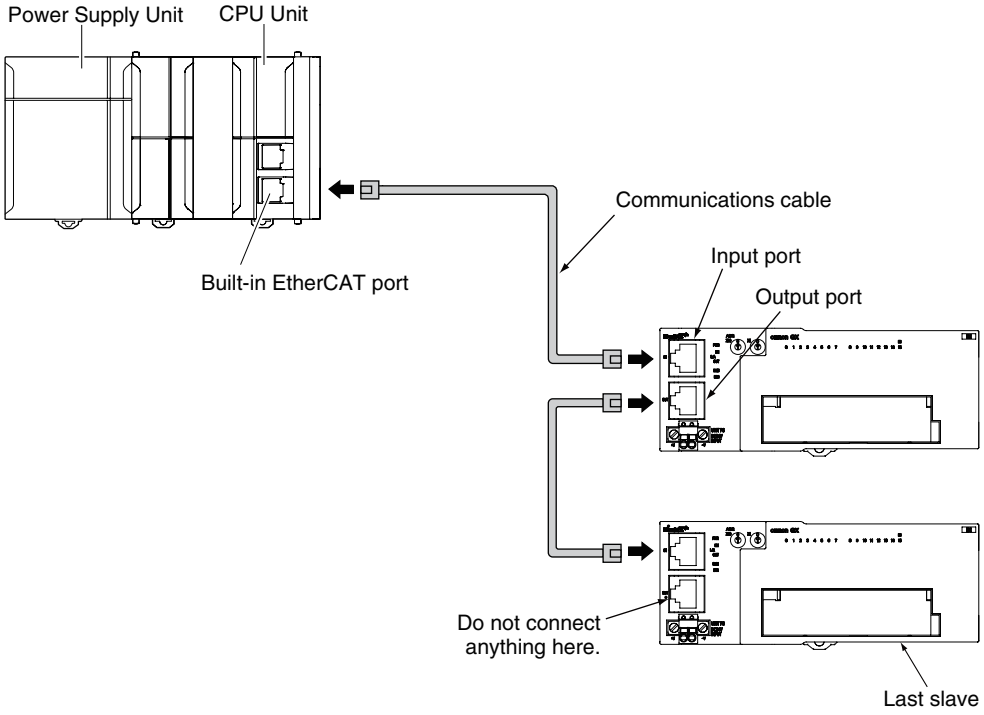
This Guide uses an NJ-PA3001 AC Power Supply Unit. An NJ-PD3001 DC Power Supply Unit can also be used.

3-2-4 Laying EtherCAT Communications Cables

Connect the EtherCAT slave communications cable to the built-in EtherCAT port as shown in the following figure.

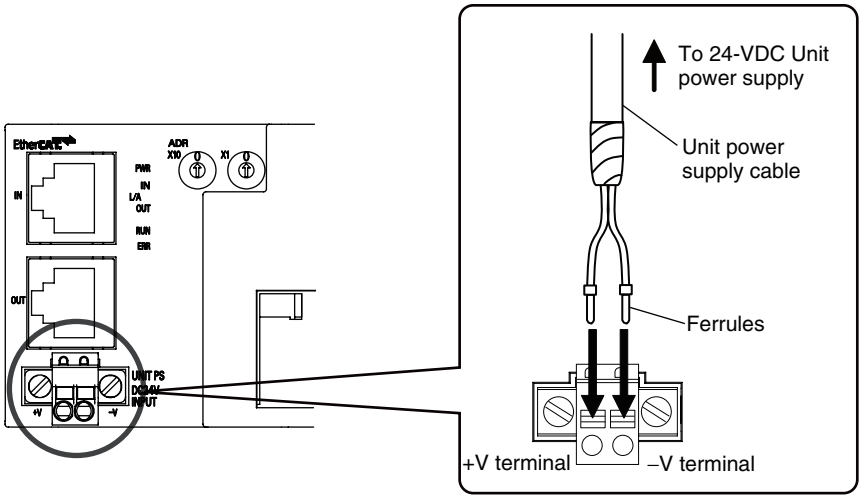
Connect the communications cable from the built-in EtherCAT port to the input port on the first slave, and then connect the communications cable to the next slave to the output port on the first slave.

Do not connect anything to the output port of the slave at the end of the network.



3-2-5 Wiring the Digital I/O Slave Units to the Power Supply

Connect the cable from the Unit power supply for the Slaves (24 VDC) to the power supply connector on each Slave.



4

Programming and Debugging

This section describes the basic procedures for programming and debugging.

4-1 Procedures

This section provides the basic operating procedures for programming and debugging.

NJ-series Controllers support programming with variables, so there is no need for concern about memory addresses. Therefore, hardware and software can be designed independently and developed in parallel.

This Guide describes the programming procedures that are used when the physical system is not yet connected so that you will understand the concepts of programming with variables.

STEP 1. Create a Project (page 4-3)

Create a project file.

STEP 2. Start Programming (page 4-6)

Define the global variables and create the POUs.

STEP 2-1 Define the global variables (page 4-6).
STEP 2-2 Write the algorithms (page 4-8).

STEP 3. Create the EtherCAT Network Configuration (page 4-18)

Create the Digital I/O Slave Unit configuration that will connect to the CPU Unit's built-in EtherCAT port.

STEP 4. Connect the Device I/O Information to the Program (page 4-20)

Assign the I/O information of the Digital I/O Slave Units to the program variables.

STEP 5. Debug the Program (page 4-21)

Transfer the project to the CPU Unit and check operation with online debugging. If an actual CPU Unit is not used, run a simulation in the Sysmac Studio to check the operation of the program with offline debugging.

STEP 5-1 Prepare for online debugging (page 4-21). Prepare for offline debugging (page 4-27).
STEP 5-2 Debug the program logic (page 4-30).
STEP 5-3 Use a data trace to confirm the operation of the indicators (page 4-39).
STEP 5-4 Modify the logic as needed with online editing (page 4-43).

4-2 Creating a Project

Start the Sysmac Studio and create a project.

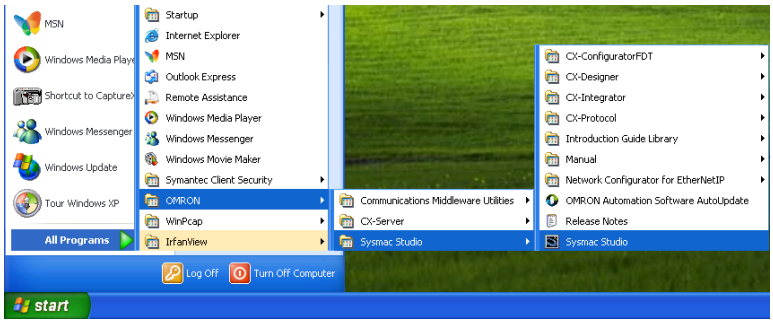
Starting the Sysmac Studio

Start the Sysmac Studio.

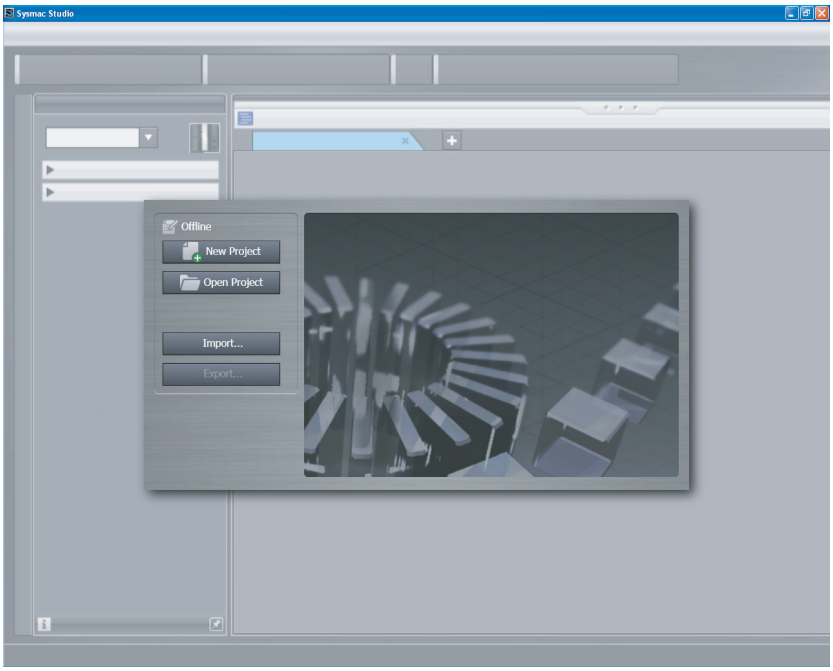
1 Use one of the following methods to start the Sysmac Studio.

- Double-click the Sysmac Studio shortcut icon on your desktop.

- Select ***All Programs*** – ***OMRON*** – ***Sysmac Studio*** – ***Sysmac Studio*** from the Windows Start Menu.



The Sysmac Studio starts and the following window is displayed.



Creating a Project

Create a project in Sysmac Studio.

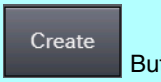
1 Click the **New Project** Button in the Project Window.

Click the 

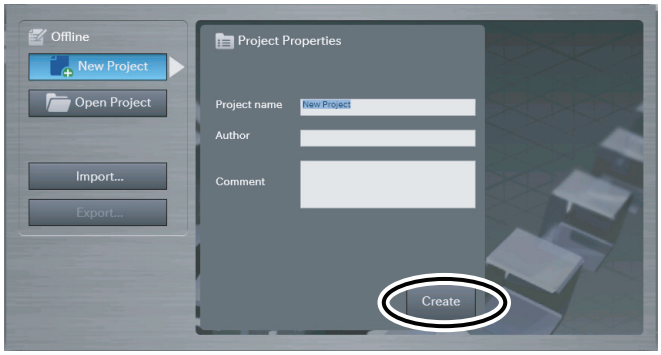
Button.

The Project Properties Dialog Box is displayed.

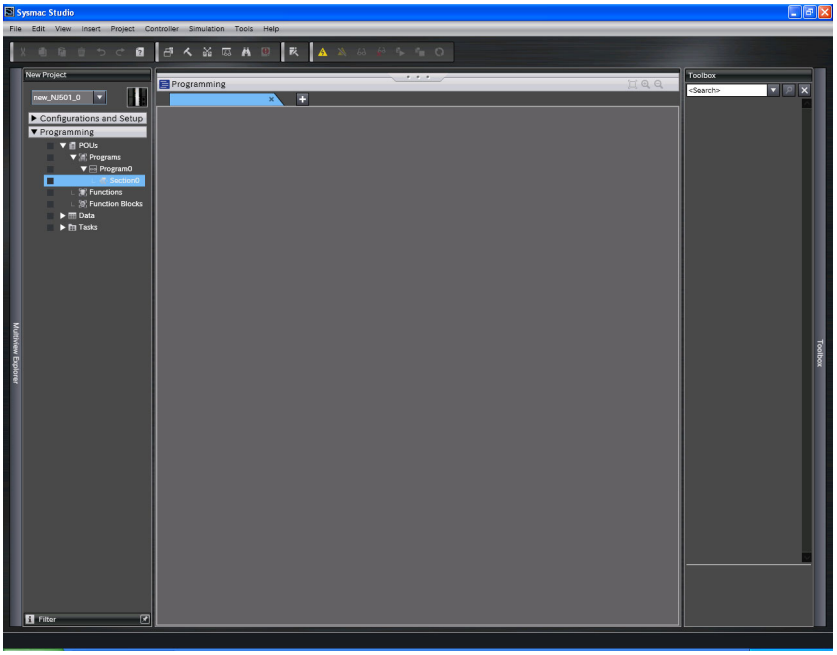
2 Click the **Create** Button in the Project Properties Dialog Box.

Click the 

Button.



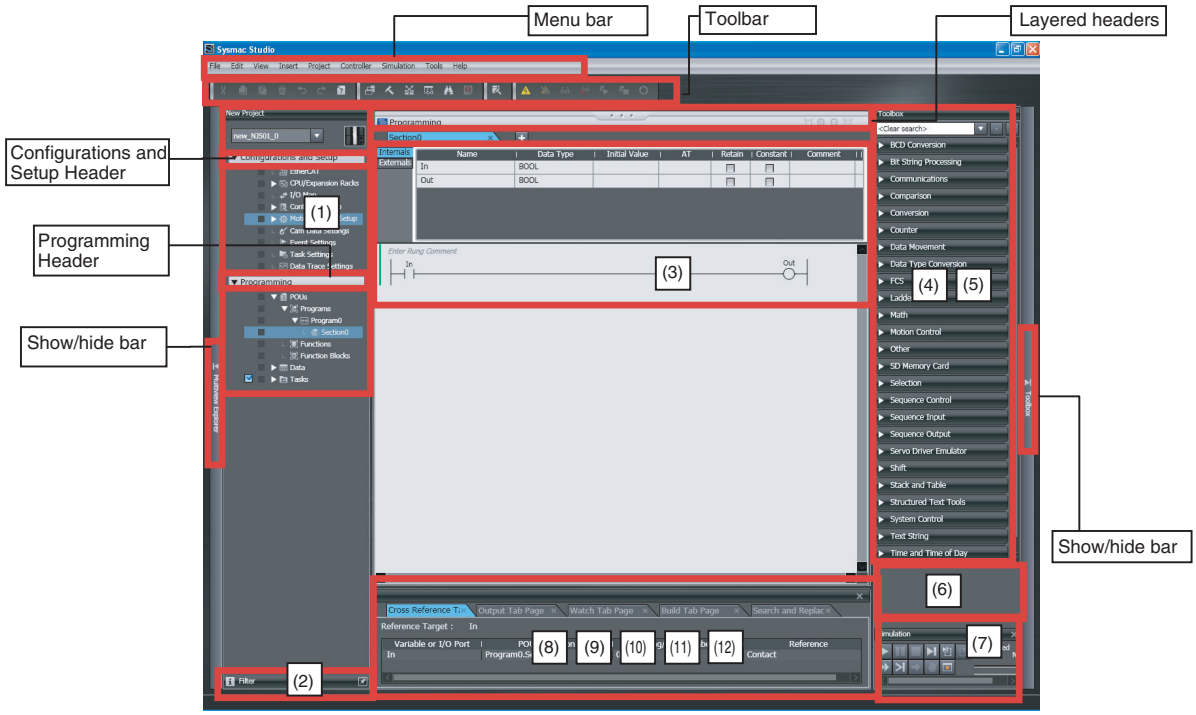
A project file is created and the following window is displayed.



This concludes the procedure to create a project file.

Additional Information

The names and functions of the parts of the Sysmac Studio Window are shown in the following figure.



No.	Item	Function
(1)	Multiview Explorer	This pane is your access point for all Sysmac Studio data. It is separated into *Configurations and Setup* and *Programming* Layers.
(2)	Filter Pane	The Filter Pane allows you to search for color codes and for items with an error icon. The results are displayed in a list.
(3)	Edit Pane	The Edit Pane is used to display and edit the data for any of the items. It is separated into *Configurations and Setup* and *Programming* Layers.
(4)	Toolbox	The Toolbox shows the objects that you can use to edit the data that is displayed in the Edit Pane.
(5)	Search and Replace Pane	In this pane, you can search for and replace strings in the data in the Programming Layer.
(6)	Controller Status Pane	The Controller Status Pane shows the current operating status of the Controller. The Controller Status Pane is displayed only while the Sysmac Studio is online with the Controller.
(7)	Simulation Pane	The Simulation Pane is used to set up, start, and stop the Simulator for the Controller.
(8)	Cross Reference Tab Page	A Cross Reference Tab Page displays a list of where variables, data types, I/O ports, functions, and function blocks are used in the Sysmac Studio.
(9)	Output Tab Page	The Output Tab Page shows the results of building.
(10)	Watch Tab Page	The Watch Tab Page shows the monitor results of the Simulator or online Controller.
(11)	Build Tab Page	The Build Tab Page shows the results of program checks and building.
(12)	Search and Replace Results Tab Page	The Search and Replace Results Tab Page shows the results when ***Search All*** or ***Replace All*** is executed.

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

4-3 Programming

The following programming is created.

- **Controlling the Yellow Indicator (Self-holding Rung)**
  - When the yellow pushbutton switch is turned ON, the yellow indicator lights.
  - When the yellow pushbutton switch is turned OFF, the yellow indicator does not go out.
  - When the red pushbutton switch is turned ON, the yellow indicator goes out.
- **Controlling the Green Indicator (ON-delay Timer)**
  - When the green pushbutton switch is turned ON, the green indicator lights 3 seconds later.
  - When the green pushbutton switch is turned ON and then turned OFF again within 3 seconds, the green indicator does not light.
  - When the green pushbutton switch is turned OFF while the green indicator is lit, the green indicator goes out.

Additional Information

The Sysmac Studio provides an environment for programming with variables. This enables you to program without having to think about the actual system configurations.

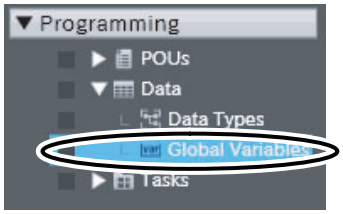
4-3-1 Defining the Global Variables

Define the global variables that are required to access the pushbutton switches and indicators. The global variables to define are listed in the following table.

Variable name	Data type	Application
SwYellow	BOOL	Yellow pushbutton switch
SwRed	BOOL	Red pushbutton switch
SwGreen	BOOL	Green pushbutton switch
LmpYellow	BOOL	Yellow indicator
LmpGreen	BOOL	Green indicator

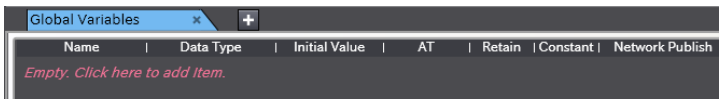
1 Double-click **Global Variables** under **Programming** – **Data** in the Multiview Explorer.

Double-click **Global Variables** in the Multiview Explorer.



2 Click inside the global variable table.

Click inside the global variable table.



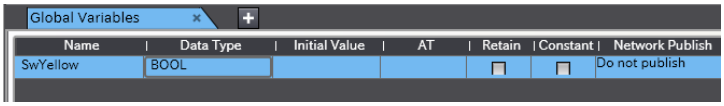
Define the *SwYellow* variable.

3 Enter the variable name in the *Name* Field.

Enter *SwYellow*.

▼

Press the Enter **Enter** Key to confirm.



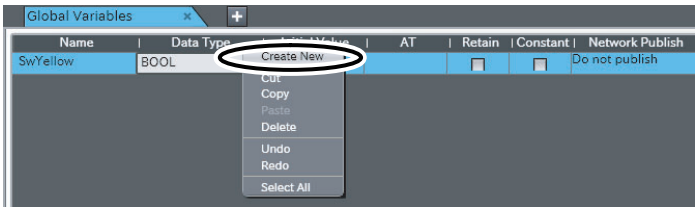
The data type field does not need to be changed because the default BOOL data type is used for this example.

This concludes the definition of the *SwYellow* variable.

Next, define the *SwRed* variable.

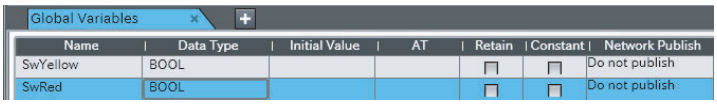
4 Press the **Insert** Key in the global variable table, or right-click in the global variable table and select ***Create New*** from the menu.

Press the Insert **Insert** Key in the global variable table, or right-click in the global variable table and select ***Create New*** from the menu.



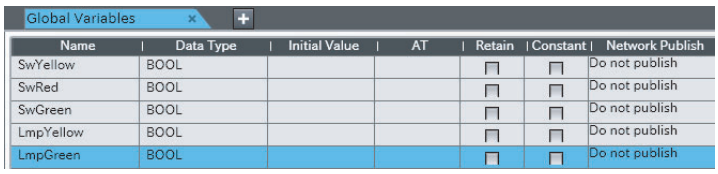
5 Enter the variable name in the *Name* Field.

Enter *SwRed*, and then press the Enter **Enter** Key to confirm.



The *SwGreen*, *LmpYellow*, and *LmpGreen* variables are defined by following this same procedure.

6 Define the *SwGreen*, *LmpYellow*, and *LmpGreen* variables by following this same procedure.



This concludes the definition of the global variables.

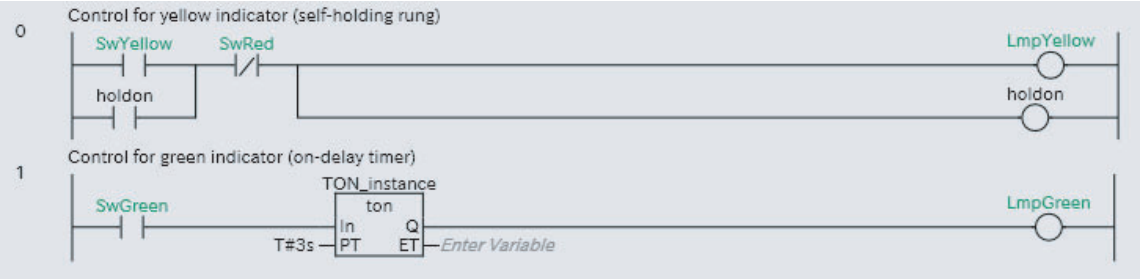
Additional Information

- You can read and write global variables from any POU (i.e., any program, function, or function block). If global variables are read and rewritten from various places in the project, you can use cross references to see where the global variables are used. Refer to *A-1 Using Cross References* for information on using cross references.
- Keyboard entry, name entry assistance, and data type input assistance allow you to edit variables quickly and accurately. Refer to *A-2 Useful Functions for Editing Variable Tables* for details.

4-3-2 Writing the Algorithm

Create the program algorithms. The algorithm is written in Section0 of Program0. Program0 is automatically created when you create a project.

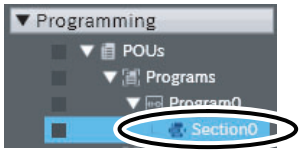
Create the algorithm.



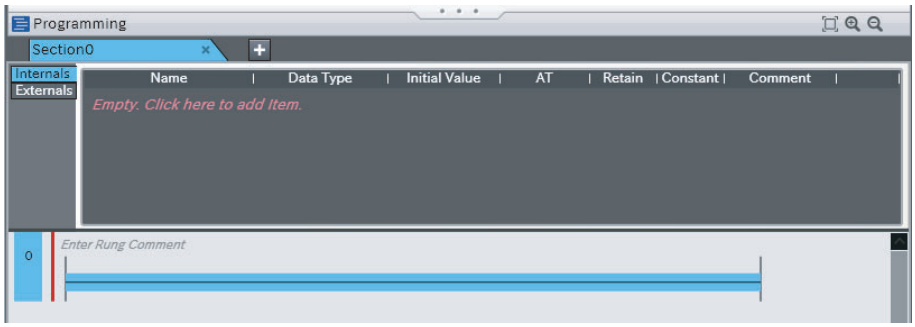
Opening the Ladder Editor

Open the Ladder Editor to create the algorithms required.

1 Double-click **Section0** under **Programming** – **POUs** – **Programs** – **Program0** in the Multiview Explorer.



The Ladder Editor for Section0 is displayed.



Creating the Algorithm for Controlling the Yellow Indicator (Self-holding Rung)

Create the algorithm.

- When the yellow pushbutton switch is turned ON, the yellow indicator lights.
- When the yellow pushbutton switch is turned OFF, the yellow indicator does not go out.
- When the red pushbutton switch is turned ON, the yellow indicator goes out.

The yellow indicator should stay lit even when the yellow pushbutton switch is turned OFF. In this case, a self-holding rung is used.

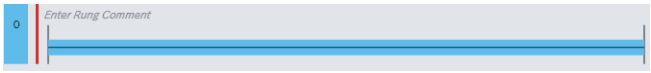
Here, shortcut keys are used to create the algorithm. The Sysmac Studio also provides the following ways to create algorithms.

- Dragging circuit parts from the Toolbox
- Right-clicking connecting lines and selecting ***Insert Circuit Part*** from the menu
- Selecting connecting lines and then selecting circuit parts from ***Insert*** – ***Circuit Parts*** in the Main Menu

1 Add an input.

Enter variable *SwYellow* as an N.O. input.

Click the connecting line for rung 0.



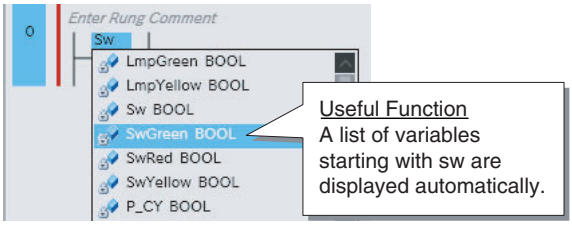
▼

Press the C **C** Key to insert an input.

The N.O. input is inserted.

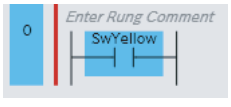
▼

Enter *sw*.



▼

Press the ↓ ↓ **Down Arrow** Key twice to select *SwYellow BOOL*, and then press the Enter Enter **Enter** Key twice to confirm.



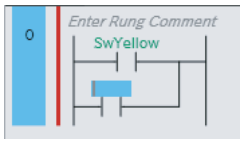
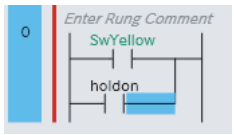
Additional Information

When you enter a rung object, the format is always checked and any mistakes are displayed as errors. If there are any errors, a red line is displayed between the rung number and the left bus bar.

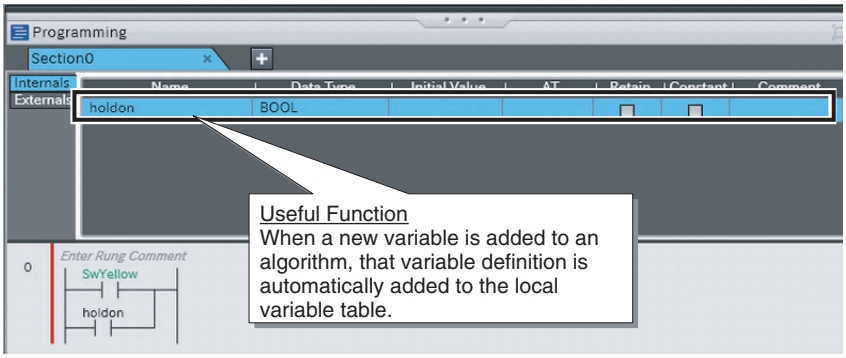
Place the mouse over the red line to view information on the rung error.

2 Add a program input in an OR structure.

Connect the *holdon* variable with an OR connection.

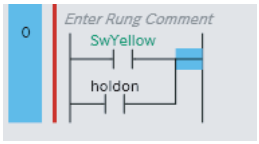
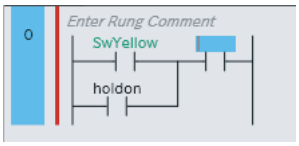
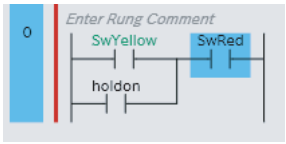
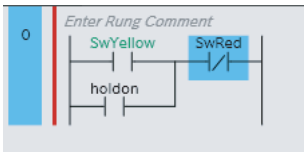
Click the *SwYellow* variable input.	
▼	
Press the W **W** Key.	 An N.O. input is inserted in an OR structure.
▼	
Enter *holdon*, and then press the Enter Enter **Enter** Key twice to confirm.	

When the *holdon* variable is inserted into the Ladder Editor, the *holdon* variable is automatically added to the local variable table for Program0.



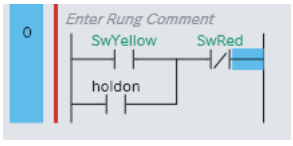
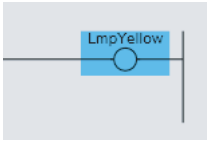
3 Add a program input in an AND structure.

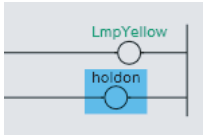
Connect the *SwRed* variable as an N.C. input with an AND connection.

Click the connecting line where you want to insert the AND instruction.	
▼	
Press the C **C** Key.	 An N.O. input is inserted in an AND structure.
▼	
Enter *swred*, and then press the Enter Enter **Enter** Key twice to confirm.	
▼	
Click the input for the *SwRed* variable and press the / **/** Key.	 The N.O. input is changed to an N.C. input.

4 Add an output.

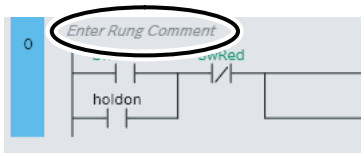
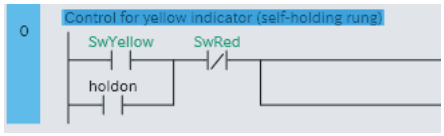
Connect the *LmpYellow* variable and *holdon* variable as outputs.

Click the connecting line where you want to insert the output.	
▼	
Press the O **O** Key to insert an output.	 The output is inserted.
▼	
Enter *LmpYellow*, and then press the Enter Enter **Enter** Key twice to confirm.	
▼	

Click the *LmpYellow* output and press the O **O** Key to insert an output.	
▼	
Enter *holdon*, and then press the Enter Enter **Enter** Key twice to confirm.	

5 Enter a rung comment.

Enter a comment for Rung0.

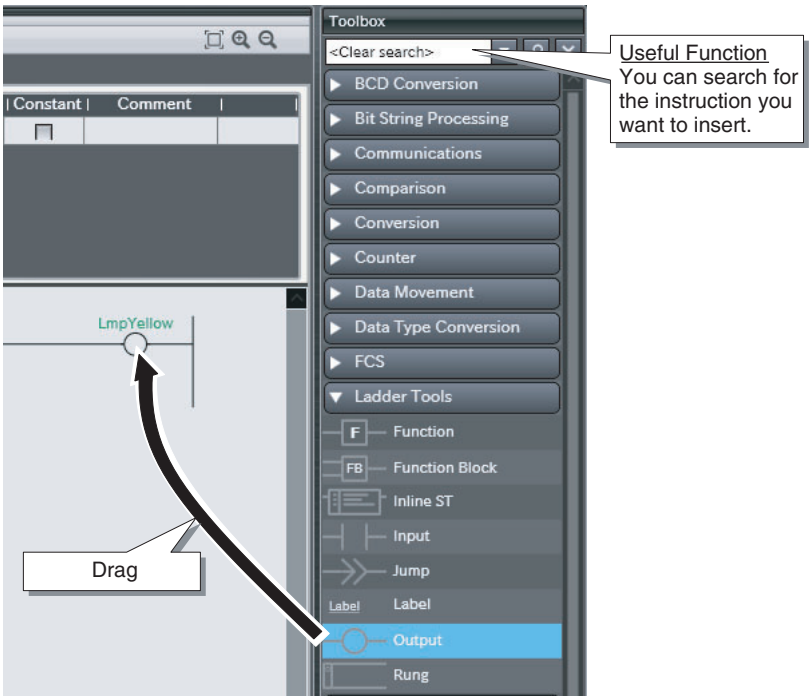
Click ***Enter Rung Comment***.	
▼	
Enter *Control for yellow indicator (self-holding rung)* and press the Enter **Enter** Key twice to confirm.	

This concludes the creation of the algorithm for controlling the yellow indicator.

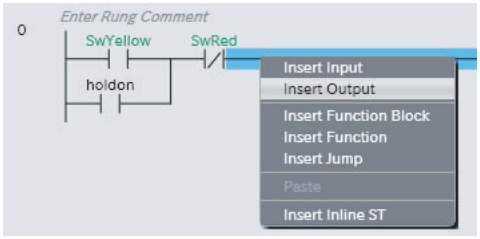
Additional Information

You can also insert ladder rungs without using the shortcut keys, as shown below.

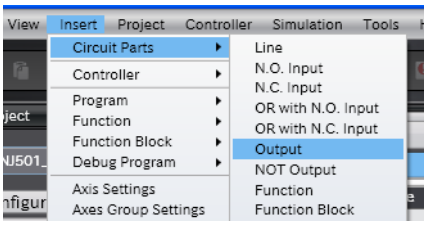
- Method 1: Drag a circuit part from **Ladder Tools** in the Toolbox.



- Method 2: Right-click a connecting line and select the circuit part from the menu to insert it.



- Method 3: Click a connecting line and then select a circuit part from ***Insert*** – ***Circuit Parts*** in the Main Menu.



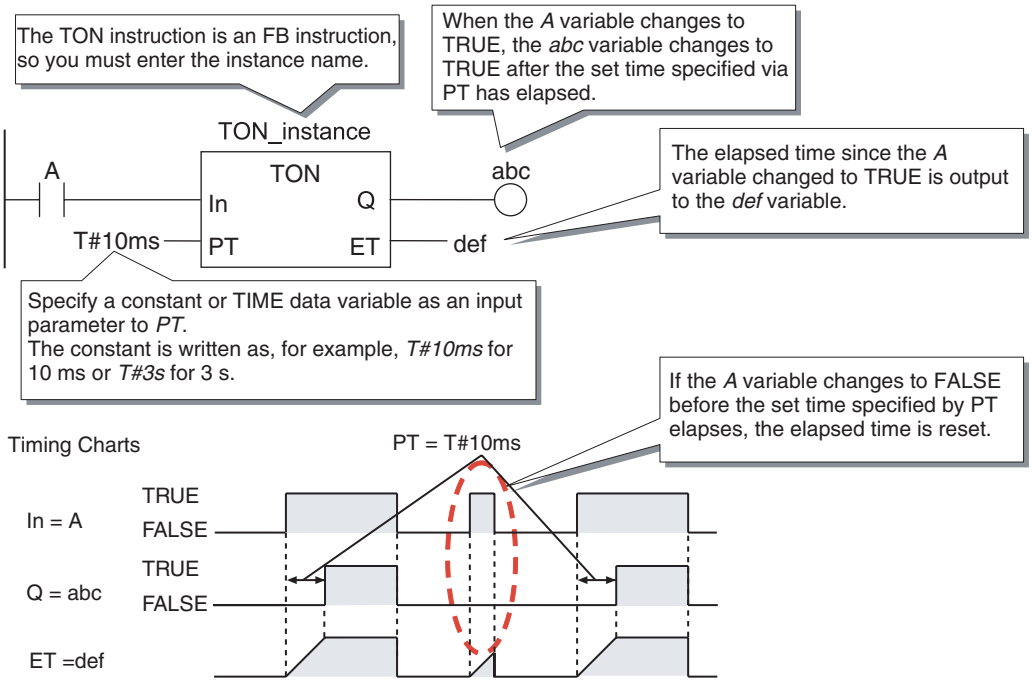
Creating the Algorithm for Controlling the Green Indicator (ON-delay Timer)

Create the algorithm.

- When the green pushbutton switch is turned ON, the green indicator lights 3 seconds later.
- When the green pushbutton switch is turned ON and then turned OFF again within 3 seconds, the green indicator does not light.
- When the green pushbutton switch is turned OFF while the green indicator is lit, the green indicator goes out.

To create a delay from the time the green pushbutton switch is pressed until the green indicator lights, use the On-Delay Timer instruction (TON).

The On-Delay Timer instruction (TON) operates as shown in the following figure.



The On-Delay Timer instruction (TON) contains the following variables.

Parameter	Item	I/O	Description
In	Timer input	Inputs	TRUE: Timer start signal FALSE: Timer reset signal
PT	Set time		Time from when timer starts until *Q* changes to TRUE
Q	Timer output	Outputs	TRUE: Timer output ON FALSE: Timer output OFF
ET	Elapsed time		Elapsed time since timer started

Additional Information

NJ-series CPU Units support the following five timer instructions.

You can select instructions according to the applications to increase the readability of your programs as well as programming productivity.

Instruction	Name	Operation
TON	On-Delay Timer	The TON instruction outputs TRUE when the set time elapses after the timer starts.
TOF	Off-Delay Timer	The TOF instruction outputs FALSE when the set time elapses after the timer starts.
TP	Timer Pulse	The TP instruction outputs TRUE while the set time elapses after the timer starts.
AccumulationTimer	Accumulation Timer	The AccumulationTimer instruction totals the time that the timer input is TRUE.
Timer	Hundred-ms Timer	The Timer instruction outputs TRUE when the set time elapses after the timer starts. Set the time in increments of 100 ms. The timing accuracy is 100 ms. Use this instruction to shorten instruction execution time.

1 Insert a rung.

Click rung 0.

▼

Press the R **R** Key.

Or, right-click and select ***Insert Rung Below*** from the menu.*

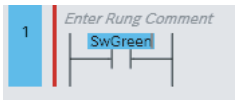
A rung is inserted below the selected rung.

* To insert above the selected rung, click the rung, and then hold down the **Shift** Key and press the **R** Key. Or, right-click and select ***Insert Rung Above*** from the menu.

2 Insert the FB instruction.

Insert the On-Delay Timer instruction.

Add the N.O. input for the *SwGreen* variable.

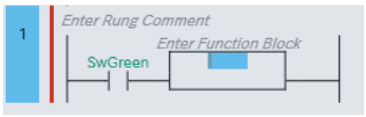


▼

Click the connecting line at the location where you want to insert the instruction.

▼

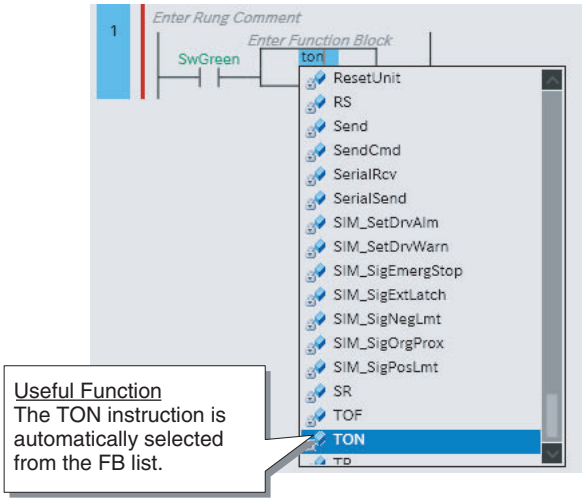
Press the F **F** Key.*



An FB with no function block definition name is inserted.

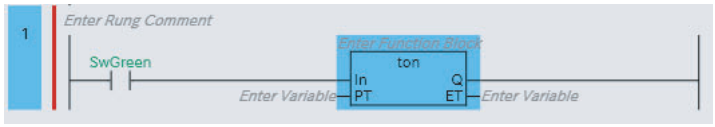
▼

Enter *ton*.



▼

Press the Enter Enter **Enter** Key twice to confirm.

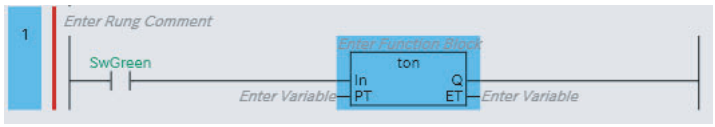


* Press the **I** Key to insert a function.

3 Enter the instance name of the FB instruction.

Enter the instance name of the inserted On-Delay Timer instruction. Enter *TON_instance* for the instance name.

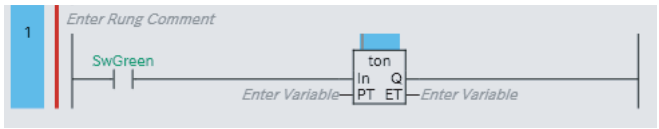
Click the TON instruction.



▼

Press the Enter **Enter** Key.

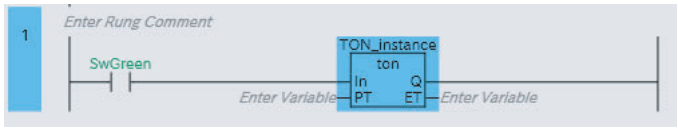
Or, right-click and select ***Enter Instance Variable Name*** from the menu.



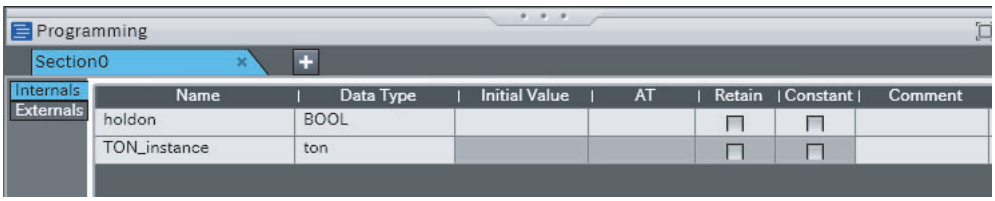
This allows you to enter an instance name.

▼

Enter *TON_instance*, and then press the Enter **Enter** Key to confirm.



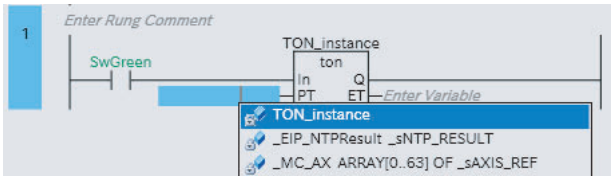
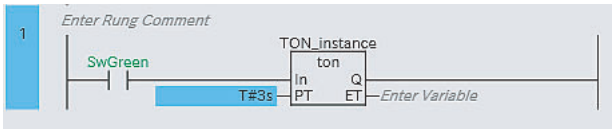
After the instance name is entered, *TON_instance* is automatically added to the local variable table of Program0.



4 Enter the parameters.

Enter the parameters for the On-Delay Timer instruction.

Enter the input parameter for input variable *PT*. We want the indicator to light after three seconds, so enter *T#3s* here.

Double-click ***Enter Variable*** for the PT input variable.	 This allows you to enter an input parameter.
▼	
Enter *T#3s*, and then press the **Enter** Key to confirm.	

This concludes entering the input parameter for input variable *PT*.

For this algorithm, the elapsed time from the start of the timer is not required, so no entries are needed for output variable *ET* (elapsed time).

Connect the *LmpGreen* output to the right of output variable *Q* to complete entering the parameter for the On-Delay Timer instruction.

Enter a rung comment to complete the algorithm for controlling the green indicator.

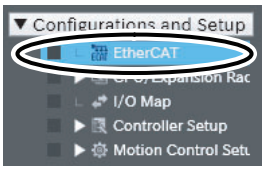
Additional Information

Refer to *A-3 Frequently Used Programming Operations* for the procedure to add differential type inputs, always TRUE inputs, and other common tasks.

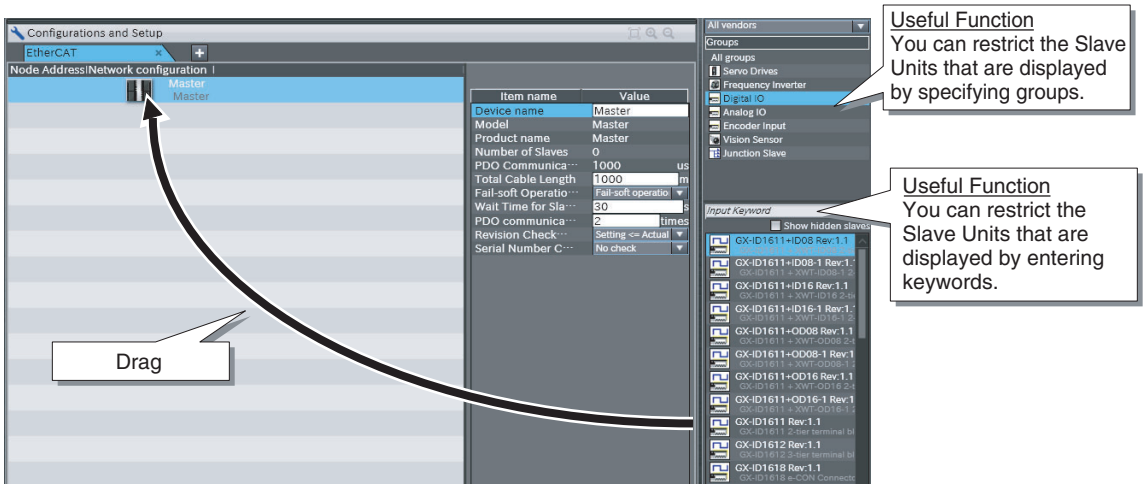
4-4 Creating the EtherCAT Network Configuration

Register the Digital I/O Slave Units (GX-ID1611 and GX-OD1611) in the EtherCAT network configuration.

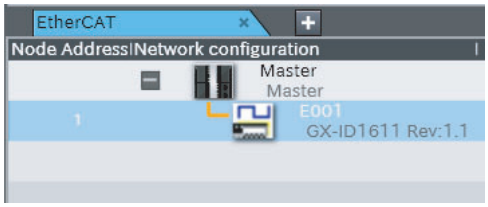
1 Double-click **EtherCAT** under **Configurations and Setup** in the Multiview Explorer. Or, right-click **EtherCAT** under **Configurations and Setup** and select ***Edit*** from the menu.



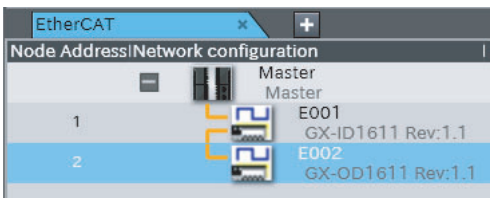
2 Drag **GX-ID1611 Rev:1.1** from the Toolbox to the master on the EtherCAT Tab Page. Or, select the master on the EtherCAT Tab Page, and then double-click **GX-ID1611 Rev:1.1** in the Toolbox.



The GX-ID1611 is added under the master.



3 In the same way, drag **GX-OD1611 Rev:1.1** from the Toolbox to the GX-ID1611 on the EtherCAT Tab Page. Or, select the GX-ID1611 on the EtherCAT Tab Page, and then double-click **GX-OD1611 Rev:1.1** in the Toolbox.



This concludes the creation of the network configuration.

Additional Information

If the physical EtherCAT network configuration is already connected, you can automatically create the virtual network configuration in the Sysmac Studio based on the physical network configuration. For details, refer to *A-4 Creating an Online Network Configuration* (page A-15).

4-5 Connecting the Device I/O Information to the Program

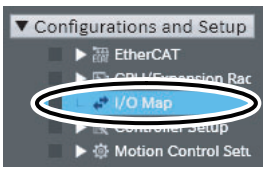
Variables that are used to access data in devices such as EtherCAT slaves and CJ-series Units are called device variables.

In this section, we will set the global variables we defined to access the pushbutton switches and indicators to device variables for the Digital I/O Slave Units. These settings create the connections between the I/O information of the Digital I/O Slave Units and the program.

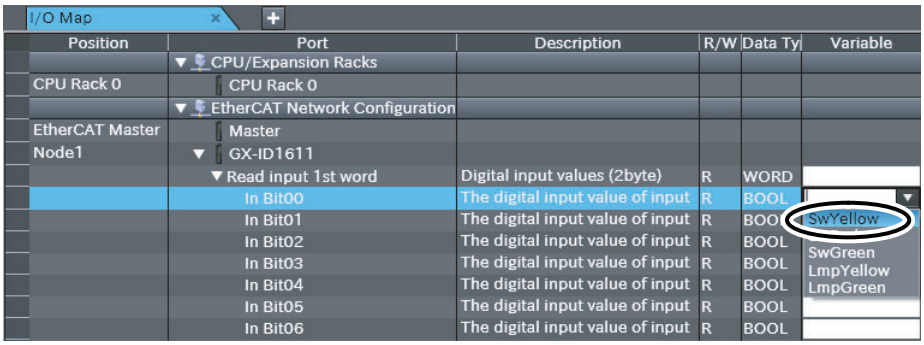
Slave location	Slave name	I/O port	Description	Global variables set for the device variable
Node 1	GX-ID1611	In Bit00	Input bit 00	SwYellow
		In Bit01	Input bit 01	SwRed
		In Bit02	Input bit 02	SwGreen
Node 2	GX-OD1611	Out Bit00	Output bit 00	LmpYellow
		Out Bit01	Output bit 01	LmpGreen

Device variables are set on the I/O Map.

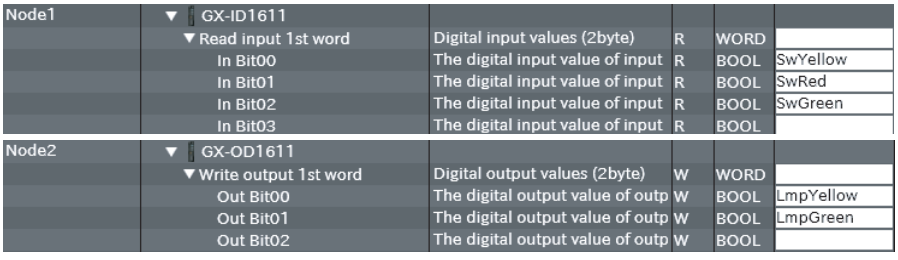
1 Double-click **Axis Settings** under **Configurations and Setup** – **I/O Map** in the Multiview Explorer. Or, right-click **I/O Map** under **Configurations and Setup** and select ***Edit*** from the menu.



2 Double-click the box in the Variable Column for **In Bit00** under **Node1**, **GX-ID1611** on the I/O Map, and select the SwYellow global variable.



3 Set the other global variables to device variables in the same way.



This concludes the setting of device variables for Digital I/O Slave Units.

4-6 Debugging the Program

Online debugging is the process of placing the Sysmac Studio online with a CPU Unit to check its operation. Offline debugging is the process of checking the operation of a CPU Unit with the Simulator in the Sysmac Studio on a computer, without going online with the CPU Unit. Use offline debugging if you do not have an physical CPU Unit to use for testing.

The actual debugging operations in this Guide are the same for both online and offline debugging, but require different steps for preparation.

Additional Information

- There are some differences in debugging between online and offline debugging. For details, refer to *A-5 Differences between Online and Offline Debugging* (page A-17).
- You must build programs whenever you create or change them. Building is the process of converting your project programs into a format that is executable on the CPU Unit. The Sysmac Studio will automatically build the programs when you change them. Therefore, the procedure to build the programs is omitted from this Guide. If you perform no operations for five seconds after you change data types, global variables, or the local variables or the algorithm of a program, building the programs is started automatically.

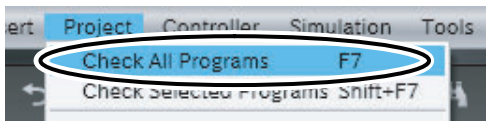
4-6-1 Preparations for Online Debugging

To prepare for online debugging, you must check the program and then transfer the project to the CPU Unit.

● Perform a Program Check

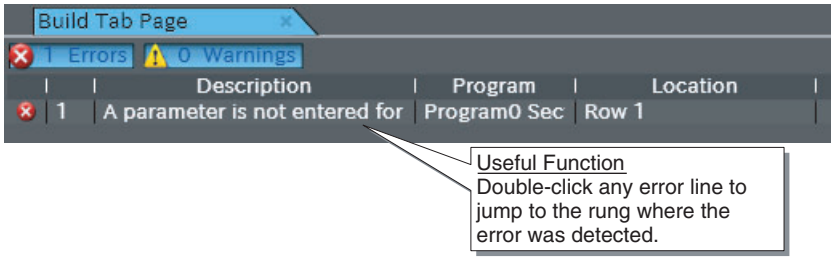
1 Check all programs.

Select ***Check All Programs*** from the Project Menu.



The results of the program check are displayed in the Build Tab Page.

If there are any errors, correct them.

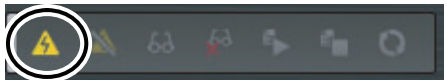


Going Online

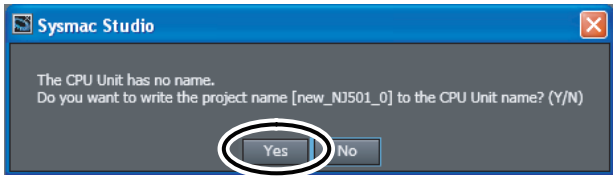
1 Turn ON the power supply to the Controller.

2 If Digital I/O Slave Units are connected, turn ON the power supply to the Digital I/O Slave Units.

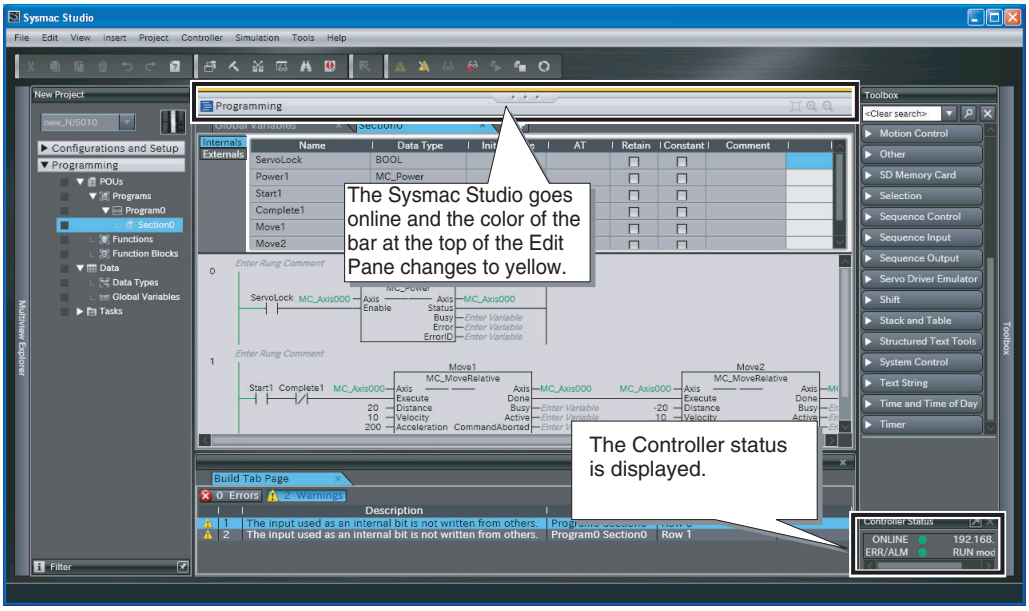
3 Use one of the following methods to go online.

Method 1: Select ***Online*** from the Controller Menu.	
Method 2: Click the  Button on the tool-bar.	
Method 3: Press the Ctrl **Ctrl** Key + the W **W** Key.	

4 The following message is displayed if no CPU Unit name is set for the Controller. Click the **Yes** Button.

Click the Yes Button.	

The Sysmac Studio goes online.

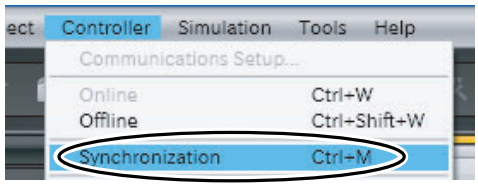


Transferring the Project

You must transfer the project to the CPU Unit. The synchronize operation is used to transfer the project.

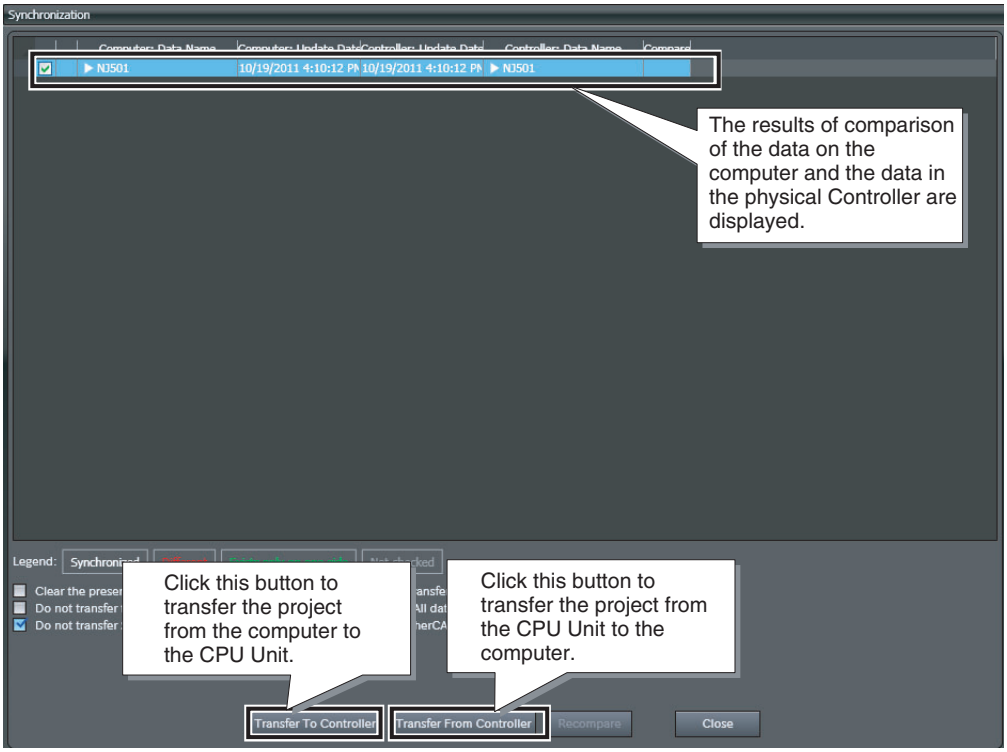
Here, “synchronize” means to automatically compare the data for the Sysmac Studio on the computer with the data in the physical Controller and transfer the data in the direction that is specified by the user.

1 Use one of the following methods to display the Synchronize Pane.

Method 1: Select ***Synchronization*** from the Controller Menu.	
Method 2: Click the Button on the toolbar.	
Method 3: Press the Ctrl Key + the M Key.	

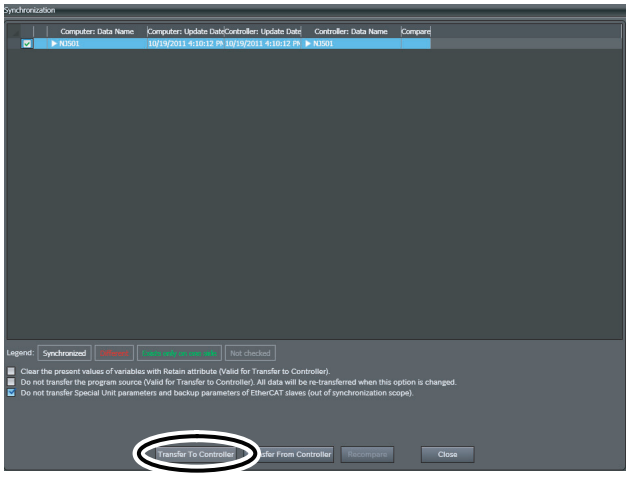
Comparison of the data on the computer and the data in the physical Controller is started.

The comparison results are displayed after the comparison is completed.



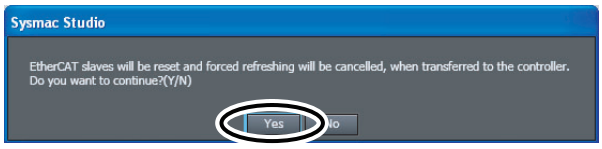
2 Click the **Transfer to Controller** Button.

Click the Transfer To Controller Button.

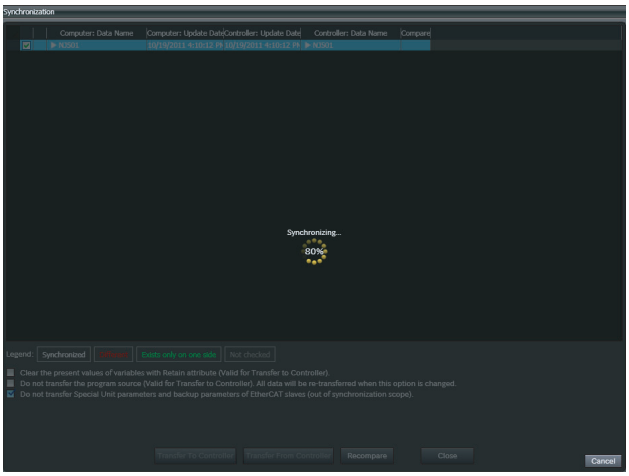


3 Click the **Yes** Button.

Click the Yes Button.

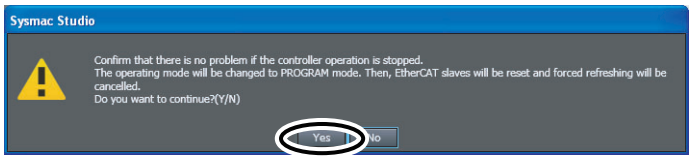


The operating mode changes to PROGRAM mode, and the Sysmac Studio starts transferring the project to the CPU Unit. During the transfer, a progress bar appears in the Synchronize Pane.



4 The following dialog box is displayed when the transfer is completed. Click the **Yes** Button.

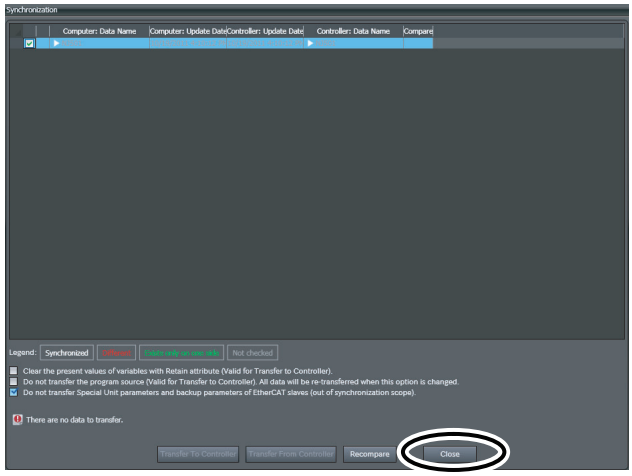
Click the Yes Button.



The operating mode changes back to RUN mode.

5 Click the **Close** Button in the lower right corner of the Synchronize Pane.

Click the Close Button.



The Synchronize Pane closes.

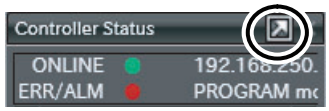
Precautions for Correct Use

The Sysmac Studio will automatically build the programs if you change data types, global variables, or the local variables or the algorithm of a program. You cannot transfer the project to the CPU Unit during execution of the build operation. Transfer the project after the build operation is completed. The progress of building the programs is displayed in a progress bar at the lower right of the window.

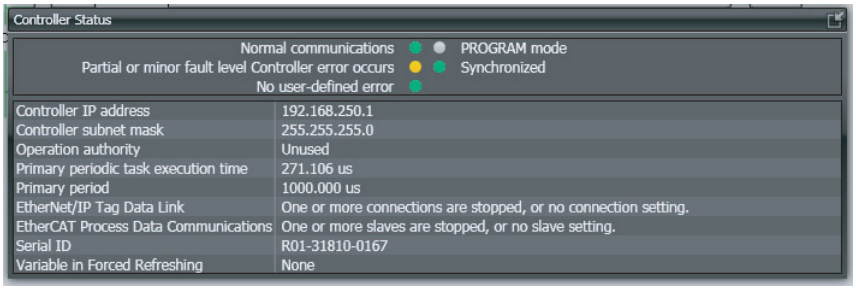
● Checking for Controller Errors

1 Open the Detailed View of the Controller Status Pane.

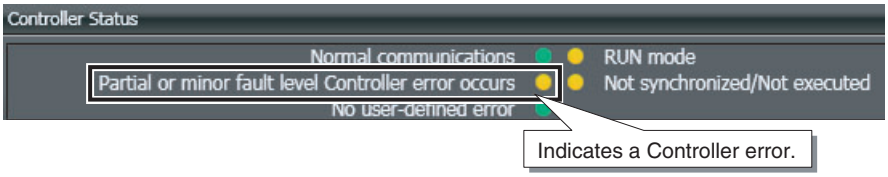
Click the Button.



The Detailed View of the Controller Status Pane is displayed.

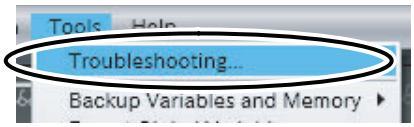


2 If a Controller error has occurred, open the Troubleshooting Window.

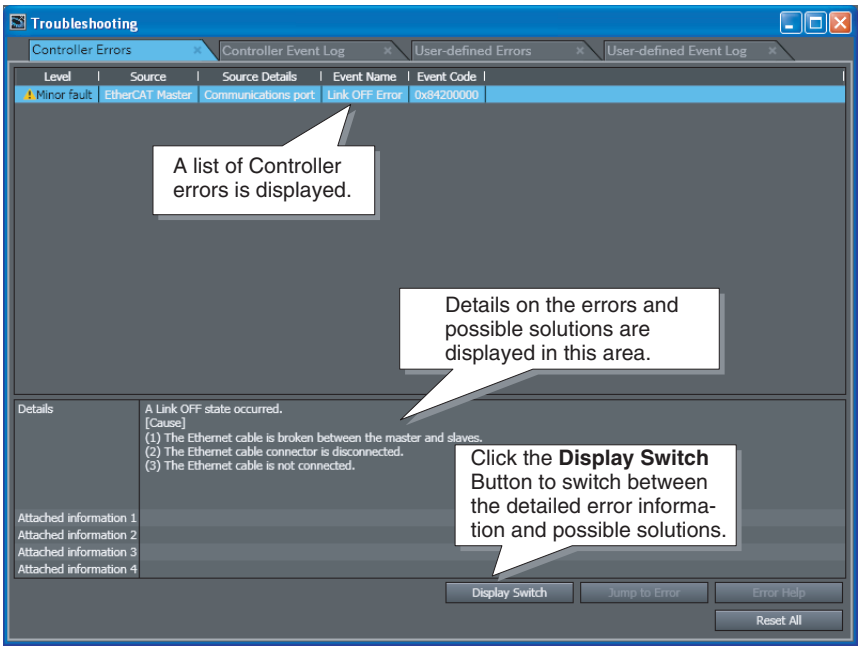


Select ***Troubleshooting*** from the Tools Menu.

Or, click the Button on the toolbar.



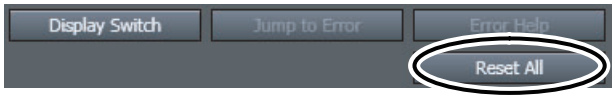
The Troubleshooting Window is displayed.



3 Refer to the error details and troubleshooting information to solve the problems and eliminate all errors.

4 Click the **Reset All** Button to clear all errors.

Click the Reset All Button.



All errors are reset.

If the cause of the error is not removed, the error will occur again.

Precautions for Correct Use

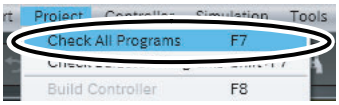
- If an EtherCAT communications cable is not properly connected or if power is not supplied to a Digital I/O Slave Unit, a minor fault level Controller error (a Link OFF Error or Network Configuration Verification Error) will occur. If you are sure that all EtherCAT communications cables are properly connected, first check to make sure that power is being supplied to the Digital I/O Slave Units before you reset the errors.
- If a major fault level Controller error has occurred, you must cycle the power to the Controller.

4-6-2 Preparations for Offline Debugging

To prepare for offline debugging, you must check the program and then start the Simulator.

● Performing a Program Check

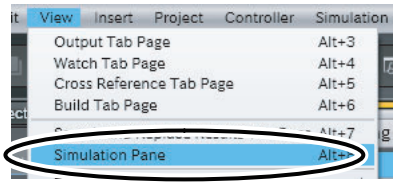
1 Check all programs.

Select ***Check All Programs*** from the Project Menu.	

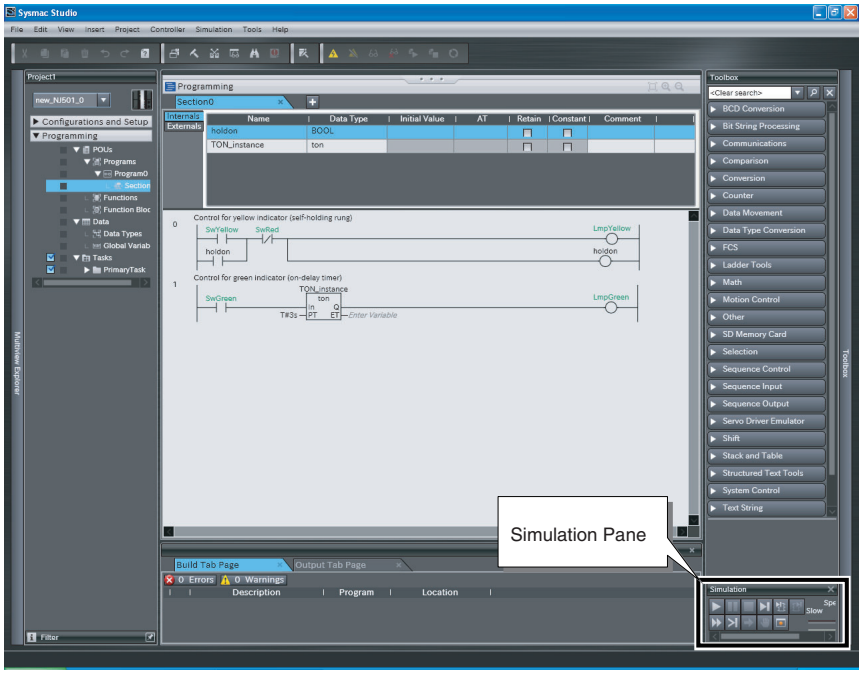
The results of the program check are displayed in the Build Tab Page. If there are any errors, correct them.

● Starting the Simulator

1 Use one of the following methods to display the Simulation Pane.

Method 1: Select ***Simulation Pane*** from the View Menu.	
Method 2: Press the Alt **Alt** Key + the 8 **8** Key.	

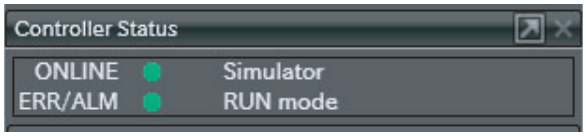
The Simulation Pane is displayed.



2 Use one of the following methods to start the Simulator.

Method 1: Select ***Run*** from the Simulation Menu.	
Method 2: Click the  Button in the Simulation Pane.	
Method 3: Press the  **F5** Key.	

When the Simulator has started, the Controller Status Pane appears as shown in the following figure.



This concludes the procedure for starting Simulator.

Precautions for Correct Use

The Sysmac Studio will automatically build the programs if you change data types, global variables, or the local variables or the algorithm of a program. You cannot start the Simulator while the build operation is in progress. Start the Simulator after the build operation is completed. The progress of building the programs is displayed in a progress bar at the lower right of the window.

Additional Information

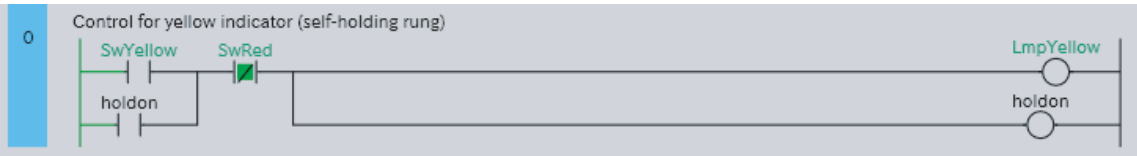
- The Simulator operates slower than the operation in the CPU Unit. Therefore, in the Simulator, the time from the execution of a timer instruction until the set time expires is slower than the time required on the CPU Unit.
- Use one of the following methods to stop simulation.

Method 1: Select ***Stop*** from the Simulation Menu.	
Method 2: Click the Button in the Simulation Pane.	
Method 3: Press the **Shift** Key + the **F5** Key.	

4-6-3 Debugging Program Logic

Next we will perform debugging to verify that the logic in our program operates as intended.

This section describes how to debug program logic using the algorithm that controls the yellow indicator as an example.



Use the functions listed in the following table to debug the logic. If you perform online debugging with a system configuration that uses Digital I/O Slave Units, you can use forced refreshing to change an input value. If you perform offline debugging or if the system configuration does not use Digital I/O Slave Units, you can use the Set/Reset Menu to change an input value.

Function name	Description
Monitoring	You can monitor the TRUE/FALSE status of program inputs and outputs and the present values of variables in the Controller. You can control BOOL variables in the Ladder Editor, ST Editor, Watch Tab Pages, or I/O Map. For this example, we will use monitoring in the Ladder Editor and on the Watch Tab Page.
Forced refreshing	Forced refreshing allows you to refresh external inputs and outputs with user-specified values from the Sysmac Studio to debug the system. If you perform online debugging with a system configuration that uses Digital I/O Slave Units, you can use forced refreshing to change the input value from a push-button switch.
Controller BOOL variables (Set/Reset)	You can change the value of any BOOL variable to TRUE or FALSE. If you perform offline debugging or if the system configuration does not use Digital I/O Slave Units, you can use the Set/Reset Menu to change the input value from a pushbutton switch.

Additional Information

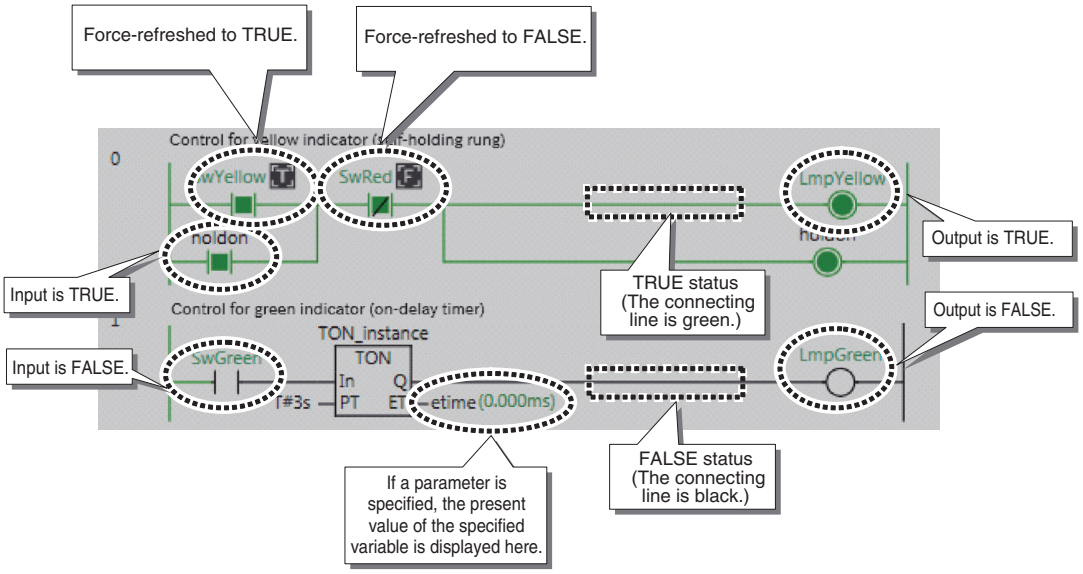
- Inputs from Input Slave Units that are not connected to switches or other input devices are always OFF. Therefore, use forced refreshing to check operation. The Set/Reset Menu cannot be used to change the values of variables. When you perform online debugging and EtherCAT slaves or CJ-series Units are actually connected, use forced refreshing to check operation.
- When a device variable is set in the I/O Map, use forced refreshing to check operation with the Simulator. However, operations from the Simulator cannot overwrite the values of variables from switches or other actual inputs. Therefore, use the Set/Reset Menu to check operation.

Monitoring

This section describes how to monitor program execution in the Ladder Editor and on the Watch Tab Page.

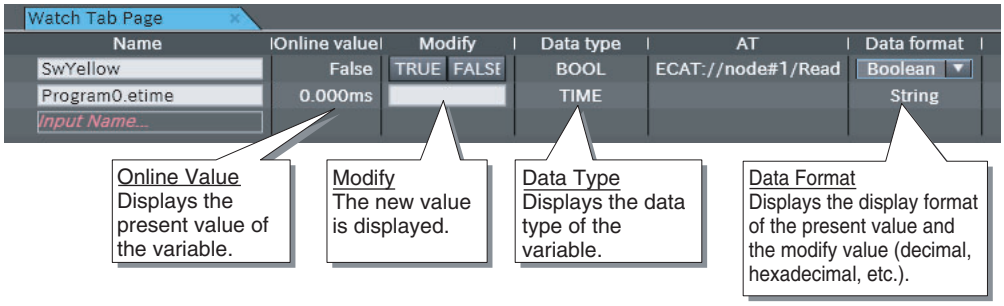
● Monitoring on the Ladder Editor

If the Ladder Editor is displayed when the Sysmac Studio is online with the Controller or when the Simulator is running, the rungs in the Ladder Editor are displayed in the monitored state. In monitored state, you can check the TRUE/FALSE status of inputs and outputs, the TRUE/FALSE status of the circuits, the forced status, the present values of variables set for parameters, and other status.



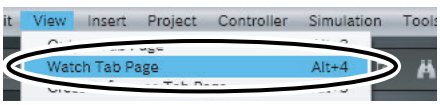
● Monitoring on a Watch Tab Page

You can check the present value of one or more variables on the Watch Tab Page.



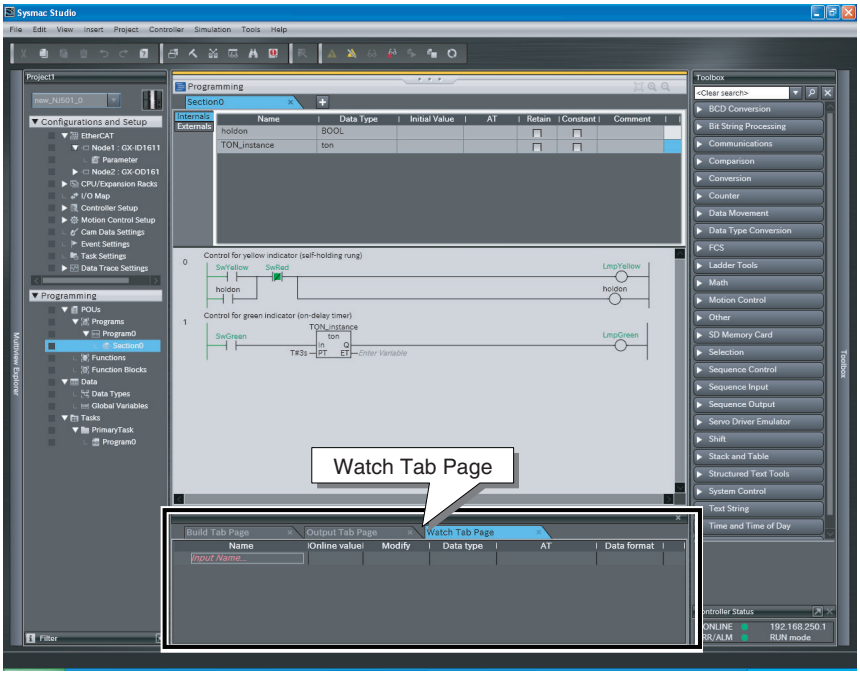
You use the following procedure to register a variable to the Watch Tab Page. The *SwYellow* variable is used as an example.

1 Use one of the following methods to display the Watch Tab Page.

Method 1: Select ***Watch Tab Page*** from the View Menu.	
Method 2: Click the  Button.	

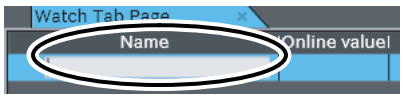
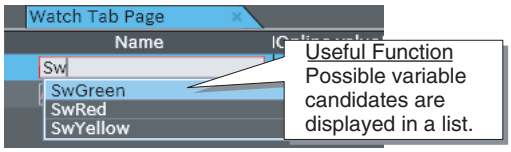
Method 3: Press the Alt **Alt** Key + the 4 **4** Key.	

The Watch Tab Page is displayed at the bottom of the Sysmac Studio Window.



2 Use one of the following methods to register a variable on the Watch Tab Page.

Method 1: Input the variable name.

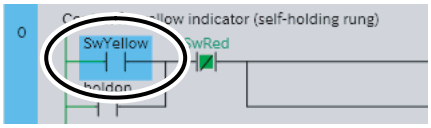
Click ***Input Name.***	
▼	
Enter *Sw*.	
▼	
Press the ↓ ↓ **Down Arrow** Key twice to select *SwYellow*, and then press the Enter **Enter** Key to confirm.	 The *SwYellow* variable is registered on the Watch Tab Page.

The following table lists input examples for registering other variables.

Variable	Description	Input example
Variable: *LmpYellow*	Global variable	LmpYellow
Variable: *holdon*	*Program0* local variable	Program0.holdon
Variable *Q*	The output variable for the *TON_instance* local variable in *Program0*.	Program0.TON_instance.Q

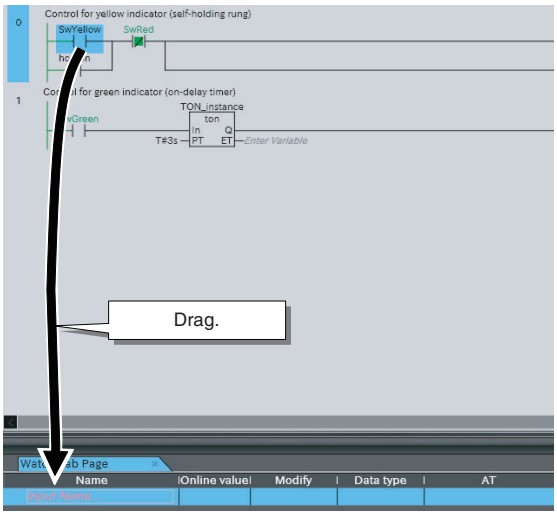
Method 2: Drag and drop.

Click the input for the *SwYellow* variable in the Ladder Editor.



▼

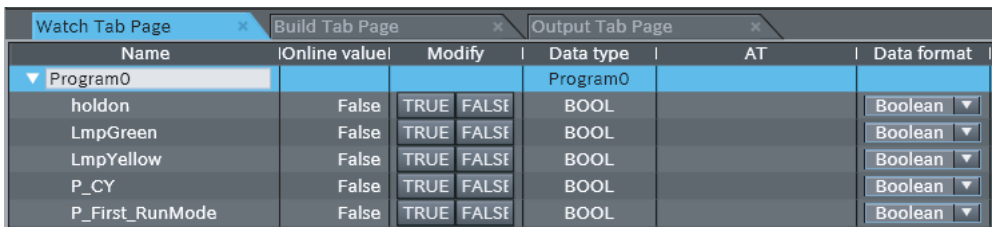
Drag the variable onto ***Input Name.***



The *SwYellow* variable is registered on the Watch Tab Page.

Additional Information

You can register a program name (or instance name for a function block) to register all the local variables for that program in a hierarchy.



Forced Refreshing

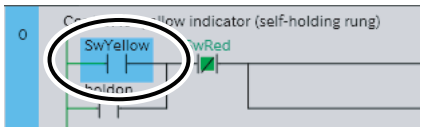
If you perform online debugging with a system configuration that uses Digital I/O Slave Units, you can use forced refreshing to change the input value from a pushbutton switch to debug the program.

● Forcing the Input Value of the Yellow Pushbutton Switch to TRUE

Use the following procedure to perform forced refreshing in the Ladder Editor.

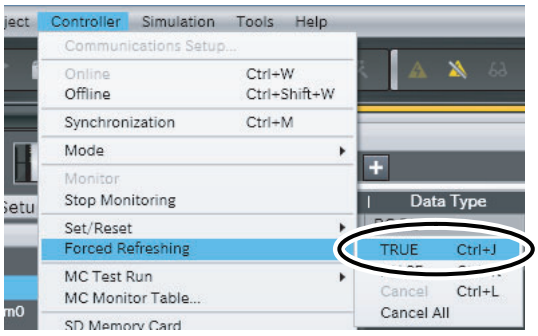
1 Select *SwYellow*, the device variable for the yellow pushbutton switch.

Click the input for the *SwYellow* variable in the Ladder Editor.

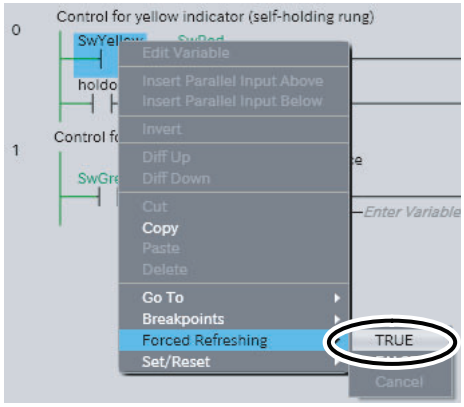


2 Use one of the following methods to force the input value of the yellow pushbutton switch to TRUE.

Method 1: Select ***Forced Refreshing*** – ***TRUE*** from the Controller Menu.



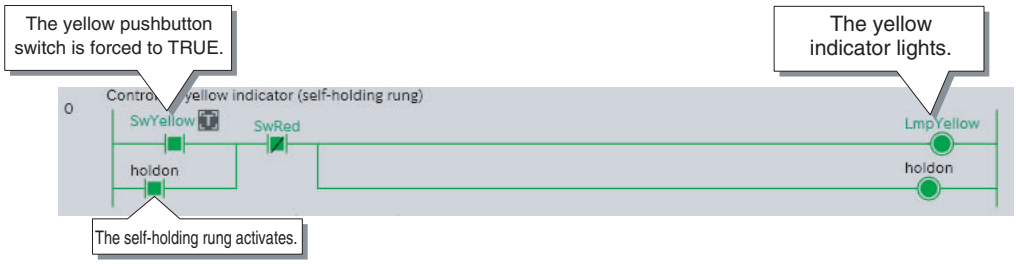
Method 2: Right-click and select ***Forced Refreshing*** – ***TRUE*** from the menu.



Method 3: Press the Ctrl **Ctrl** Key + the J **J** Key.*

* Press the **Ctrl** Key + the **K** Key to force the input to FALSE.

The input value from the yellow pushbutton switch is forced to TRUE, and the yellow indicator lights. The self-holding rung also activates at this time.

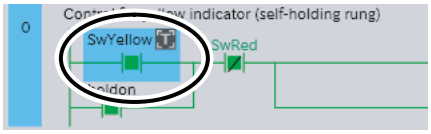


● Canceling the Forced Status for the Input Value of the Yellow Pushbutton Switch

Use the following procedure to cancel forced status in the Ladder Editor.

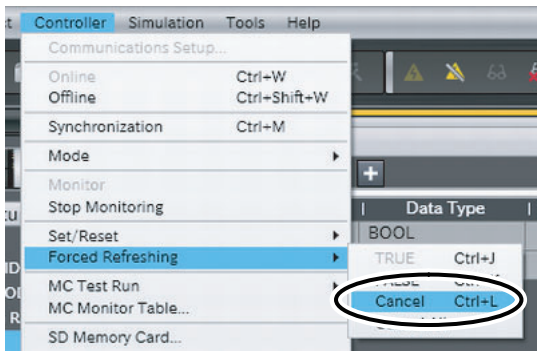
1 Select *SwYellow*, the device variable for the yellow pushbutton switch.

Click the input for the *SwYellow* variable in the Ladder Editor.

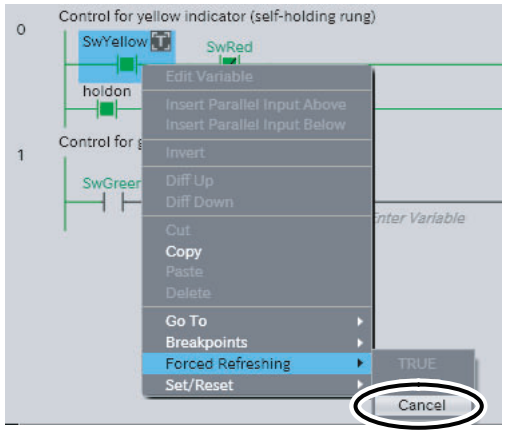


2 Use one of the following methods to release the forced input value of the yellow pushbutton switch.

Method 1: Select ***Forced Refreshing*** – ***Cancel*** from the Controller Menu.

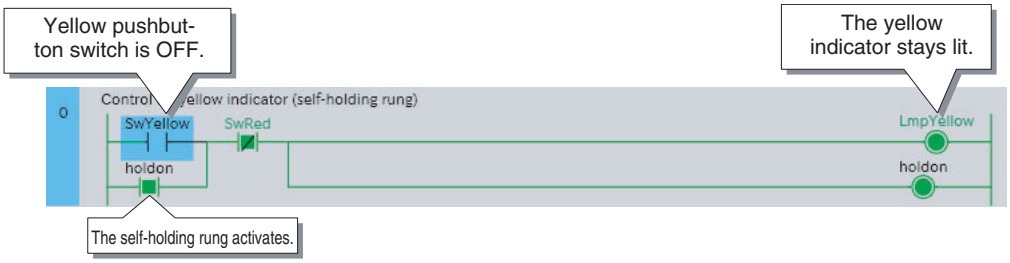


Method 2: Right-click and select ***Forced Refreshing*** – ***Cancel*** from the menu.



Method 3: Press the Ctrl **Ctrl** Key + the L **L** Key.

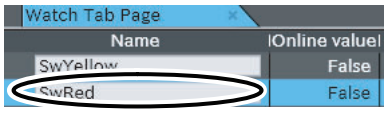
The input value changes back to FALSE when the forced status for the input value of the yellow pushbutton switch is released. In the configuration described in this Guide, the pushbutton switch is not wired and therefore the input is always FALSE. Because the self-holding rung is activated, the yellow indicator will stay lit.



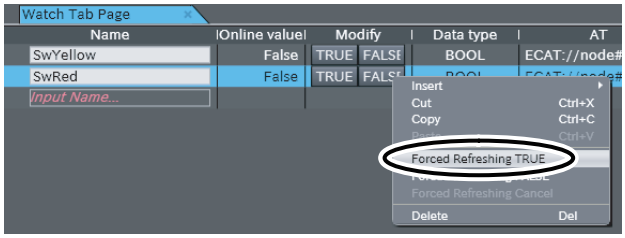
● Forcing the Input Value of the Red Pushbutton Switch to TRUE

You perform forced refreshing on a Watch Tab Page. First, register the *SwRed* variable on the Watch Tab Page.

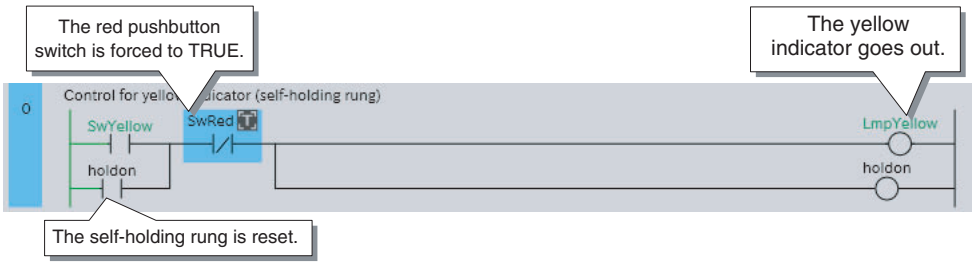
1 Select *SwRed*, the device variable for the red pushbutton switch.

Click the *SwRed* variable on the Watch Tab Page.	

2 Force the input from the red pushbutton switch to TRUE.

Right-click and select ***Forced Refreshing TRUE*** from the menu.	

When the input value from the red pushbutton switch is forced to TRUE, the self-holding rung is reset and the yellow indicator goes out.

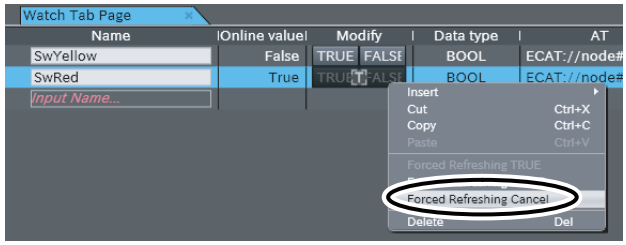


● Canceling the Forced Status for the Input Value of the Red Pushbutton Switch

Use the following procedure to cancel forced status on a Watch Tab Page

1 Cancel the forced status for the input value of the red pushbutton.

Right-click and select ***Forced Refreshing Cancel*** from the menu.



Additional Information

- You can perform forced refreshing in the above Ladder Editor, or on Watch Tab Page or on the I/O Map. On the I/O Map, you can perform forced refreshing without defining variables or creating an algorithm. Forced refreshing on the I/O Map is therefore convenient to check the I/O wiring of EtherCAT slaves or CJ-series Units. Refer to *A-6 Forced Refreshing on the I/O Map* for the procedure.
- Select ***Forced Refreshing*** – ***Cancel All*** from the Controller Menu to cancel all of the forced status.

Controller BOOL Variables (Set/Reset)

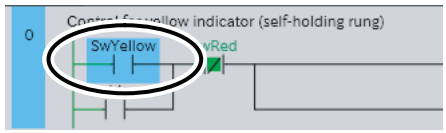
If you perform offline debugging or if the system configuration does not use Digital I/O Slave Units, you can use the Set/Reset Menu to change the input value from a pushbutton switch.

● Setting/Resetting the Input Value of the Yellow Pushbutton Switch

Use the following procedure to set or reset variables in the Ladder Editor.

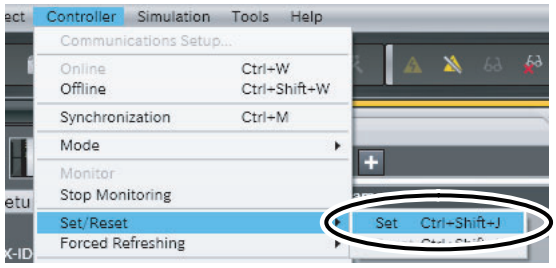
1 Select *SwYellow*, the device variable for the yellow pushbutton switch.

Click the input for the *SwYellow* variable in the Ladder Editor.

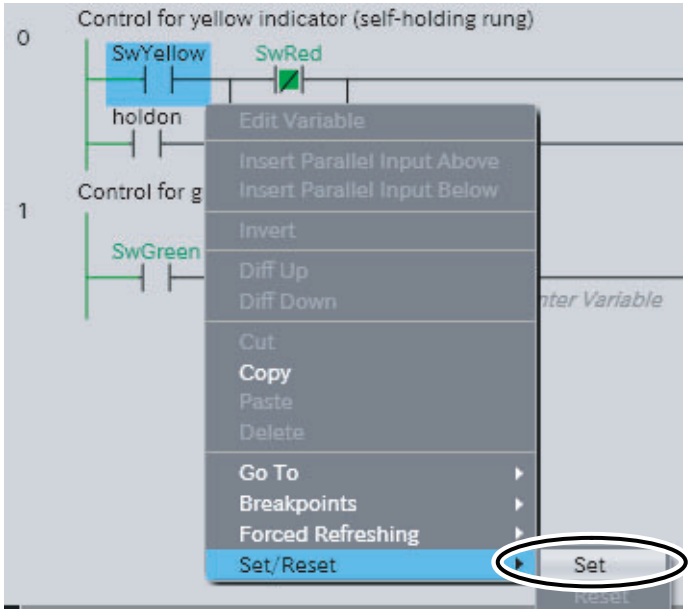


2 Use one of the following methods to reset or set the input from the yellow pushbutton switch.

Method 1: Select ***Set/Reset*** – ***Set or Reset*** from the Controller Menu.



Method 2: Right-click and select ***Set/Reset*** – ***Set*** or ***Reset*** from the menu.



Method 3:
Set:

Press the Ctrl **Ctrl** Key + the Shift **Shift** Key + the J **J** Key.

Reset:

Press the Ctrl **Ctrl** Key + the Shift **Shift** Key + the K **K** Key.

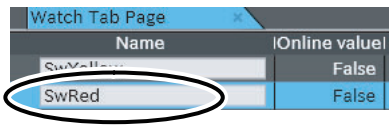
When the input from the yellow pushbutton switch is set, the *LmpYellow* variable changes to TRUE. Then, even if the input from the yellow pushbutton switch is reset, the *LmpYellow* variable remains TRUE due to the operation of the self-holding rung.

● Setting/Resetting the Variable for the Red Pushbutton Switch

Use the following procedure to set or reset variables in the Watch Tab Page. First, register the *SwRed* variable on the Watch Tab Page.

1 Select *SwRed*, the device variable for the red pushbutton switch.

Click the *SwRed* variable on the Watch Tab Page.



2 Select TRUE in the *Modify* Column to change the variable to TRUE. Select FALSE in the *Modify* Column to change the variable to FALSE.

- Set

Click the TRUE Button.

- Reset

Click the FALSE Button.

If the input from the red pushbutton switch is set while the self-holding rung is in operation, the self-holding rung is reset and the *LmpYellow* variable changes to FALSE.

4-6-4 Using a Data Trace to Confirm the Operation of the Indicators

Use a data trace to check the ON-delay timer operation of the green indicator.

The procedure is the same for both online and offline debugging.

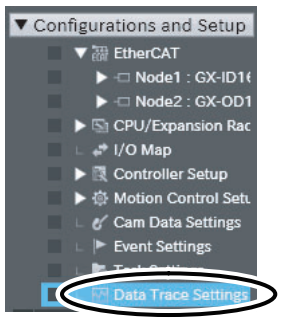
Setting Up the Data Trace

Start the data trace, and then set the trace type, trigger condition, and variable to sample.

● Starting the Data Trace

1 Create the data trace settings.

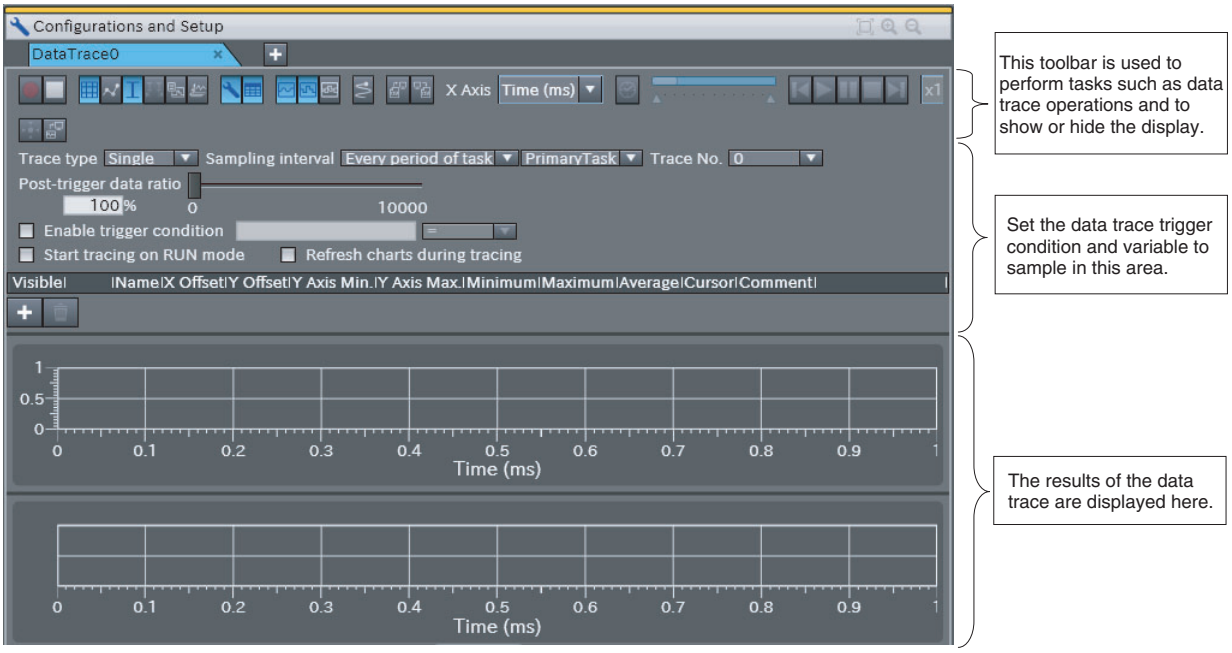
Right-click **Data Trace Settings** under **Configurations and Setup** in the Multiview Explorer and select ***Add*** – ***Data Trace*** from the menu.



DataTrace0 is added to the Multiview Explorer.

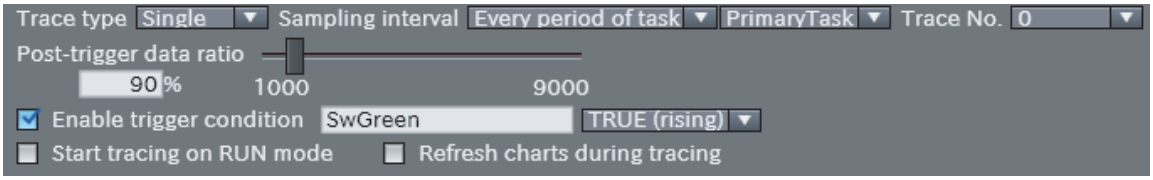
2 Double-click **DataTrace0**.

The Data Trace Tab Page is displayed.



● Setting the Trace Type and Trigger Condition

You must set the Trace Type and Trigger Condition. Set the trace type and trigger condition as shown below.



Parameter	Set value	Description
Trace type	Single Trace	This trace type samples the data before and after the trigger condition is met. A total of 10,000 samples are taken per variable. For this example, we want to sample the data before and after the trigger condition is met. Therefore, set the trace type to *Single*.
Sampling interval	Specified task period of primary task	These setting is used to sample the value once every primary periodic task period. For this example, the task period of the primary periodic task is set to 1 ms. Therefore, 10 seconds worth of data can be sampled. (1 ms × 10,000 samples)
Post-trigger data ratio	90%	A value of 90% means that 10% of the sampling data is from before the trigger condition is met and 90% of the sampling data is from after the trigger condition is met.
Trigger condition	Selected, when variable *SwGreen* changes to TRUE	This sets the trigger condition to when the *SwGreen* variable changes to TRUE.

The procedure for setting the trace type and trigger condition is given below.

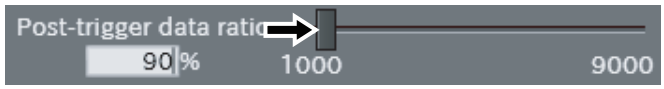
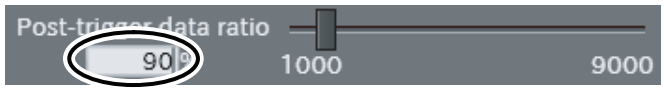
1 Select *Single* from the *Trace Type* Box.

Click the box and select *Single*.	

2 Select *Every period of task* and *PrimaryTask* from the Sampling interval Boxes.

Click the first box and select *Every period of task*. Click the second box and select *Primary Task*.	

3 Use one of the following methods to set the post-trigger data ratio to 90%.

Method 1: Drag the Post-trigger Data Ratio Slider to 90%.	
Method 2: Enter *90* directly into the Post-trigger Data Ratio Text Box.	

4 Enable the trigger condition, and set the trigger condition to when the *SwGreen* variable changes to TRUE.

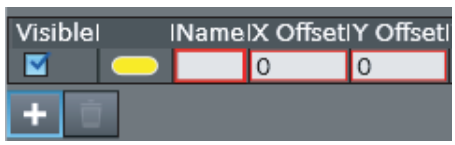
Select the *Enable Trigger Condition* Check Box. ▼ Enter *SwGreen* in the text box. ▼ Click the box and select *TRUE (rising)*.	

● Setting the Variables to Trace

You must set the variable to trace.

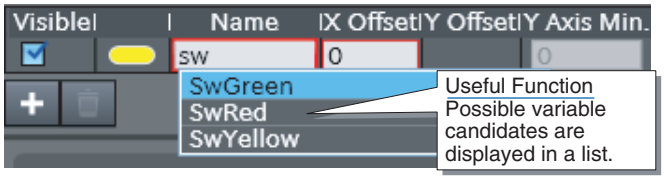
For this example, the *SwGreen* and *LmpGreen* variables are set as the variables to trace.

1 Add a trace variable line to the list.

Click the  Button.	

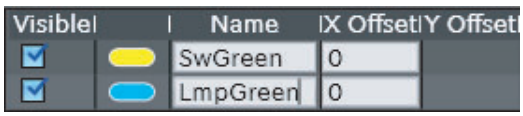
2 Add the *SwGreen* variable.

Click the *Name* Box and enter *swgreen*.



The *SwGreen* variable is registered.

3 Add the *LmpGreen* variable in the same way.



Performing a Data Trace

Start the data trace and check the results.

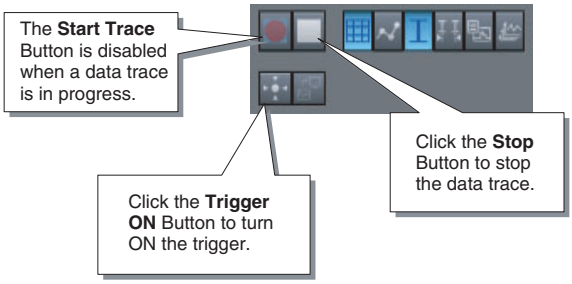
First, register the *SwGreen* and *LmpGreen* variables on the Watch Tab Page.

● Starting the Data Trace

1 Click the **Start Trace** Button.

Click the Button.

When the data trace is started, the toolbar on the Data Trace Tab Page is displayed as follows:



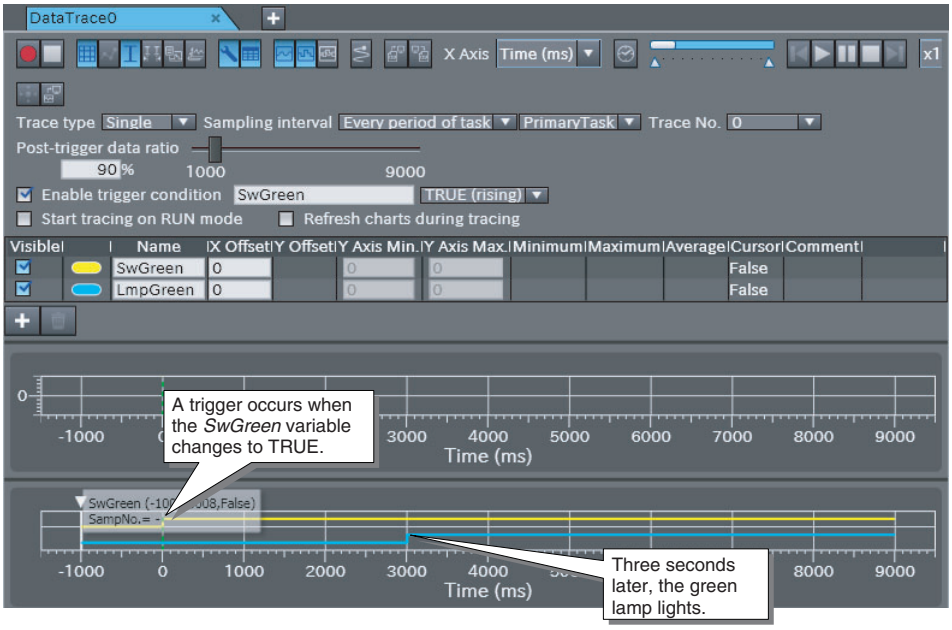
2 Change the device variable for the green pushbutton switch, *SwGreen*, to TRUE in the Watch Tab Page.

If you perform online debugging with a system configuration that uses Digital I/O Slave Units, use forced refreshing to change the variable to TRUE. If you perform offline debugging or if the system configuration does not use Digital I/O Slave Units, use the Set/Reset Menu to change the variable to TRUE.

When the trigger condition is met and the number of samples reaches 10,000, the results of the data trace are displayed.

● Displaying Data Trace Results

The results are displayed on the Data Trace Tab Page as shown in the following figure.



Additional Information

You can use the various tool buttons on the Data Trace Tab Page to show or hide certain information.
For the above figure, perform the following settings:
Hide: Grid, Cursor, Trace Settings, Analog Chart
Show: Legend

4-6-5 Modifying the Logic with Online Editing

Online editing allows you to change or add parts of the user program within the CPU Unit directly from the Sysmac Studio.

You can select any of the following to perform online editing.

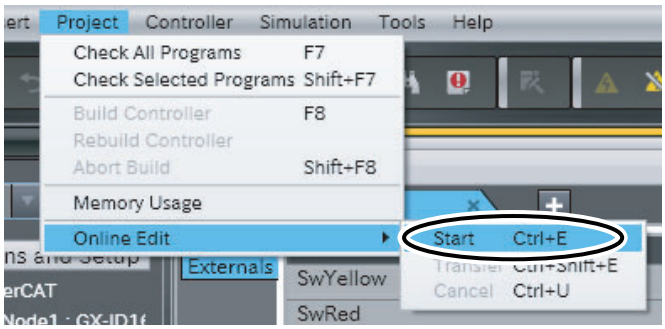
- Ladder section
- POU (program, function, or function block) written in ST
- Global variables

For this example, we will change the set time for the On-delay Timer instruction used to control the green lamp from 3 seconds to 5 seconds. The procedure is the same for both online and offline debugging.

1 Open *Section0* of *Program0*.

2 Use one of the following methods to start online editing.

Method 1: Select ***Online Edit*** – ***Start*** from the Project Menu.

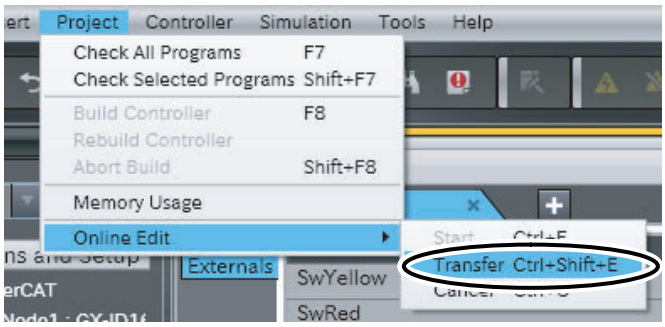


Method 2: Press the Ctrl **Ctrl** Key + the E **E** Key.

3 In the Ladder Editor, change the parameter for the On-Delay timer instruction input variable *PT* to *T#5s*.

4 Use one of the following methods to transfer the changes made.

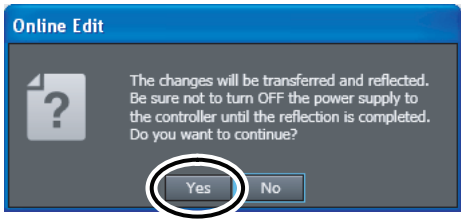
Method 1: Select ***Online Edit*** – ***Transfer*** from the Project Menu.



Method 2: Press the Ctrl **Ctrl** Key + the Shift **Shift** Key + the E **E** Key.

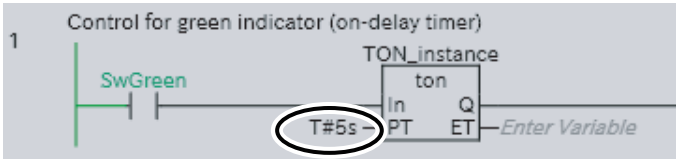
5 A confirmation dialog box is displayed. Click the **Yes** Button.

Click the Yes Button.



The logic in the user program in the CPU Unit is changed.

This concludes the process for online editing.



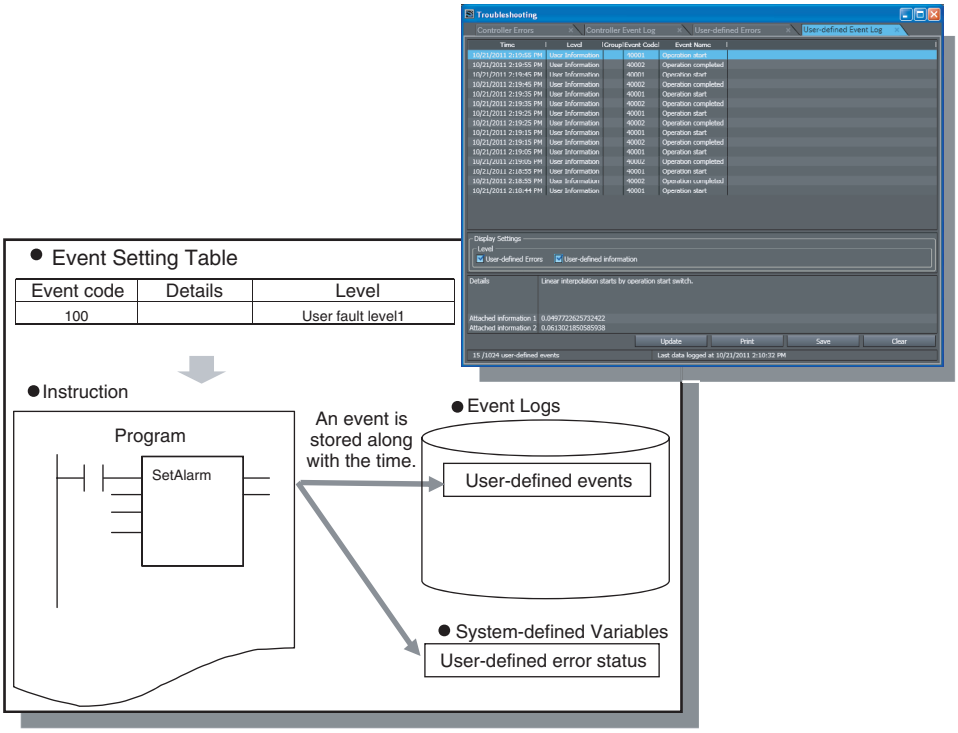
5

Useful Functions

This section describes various useful functions that you can use with a NJ-series CPU Unit or the Sysmac Studio.

5-1 Registering and Managing Application Events

You can register and manage custom events as user-defined events. User-defined events are registered in the Event Setting Table of the Sysmac Studio, and are triggered by instructions for user-defined events. Triggered user-defined events can be viewed in the Sysmac Studio or on an NS-series PT that is compatible with the NJ-series Controllers.



Characteristics of User-defined Events

User-defined events have the following characteristics:

- They can be divided up by event level (8 levels of user-defined errors and user-defined information) based on their purpose.
- You can specify a group name to represent the location or type of the event.
- They can be logged and managed along with Controller event errors and information defined in the NJ-series Controllers.
- You can view these event logs on a timeline from the Sysmac Studio or from an NS-series PT that is compatible with NJ-series Controllers.

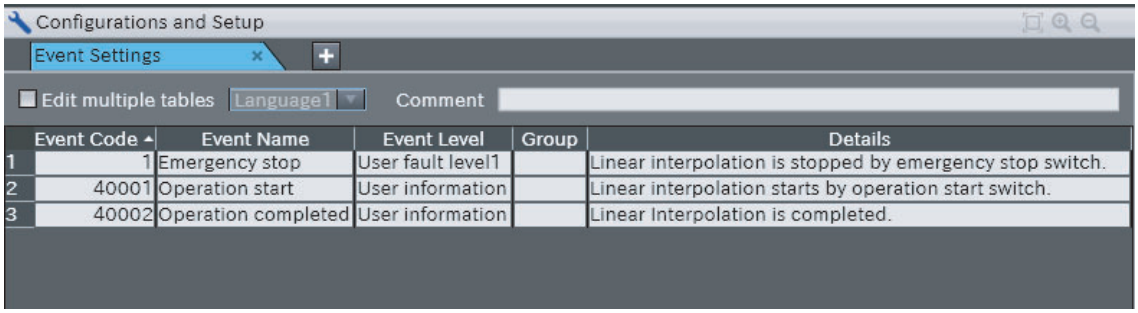
Using the User-defined Event Log

The procedures for setting user-defined events and viewing them are given below.

● Setting User-defined Events

1 Double-click **Event Settings** under **Configurations and Setup** in the Multiview Explorer.

2 Register the user-defined events in the Event Settings Table.

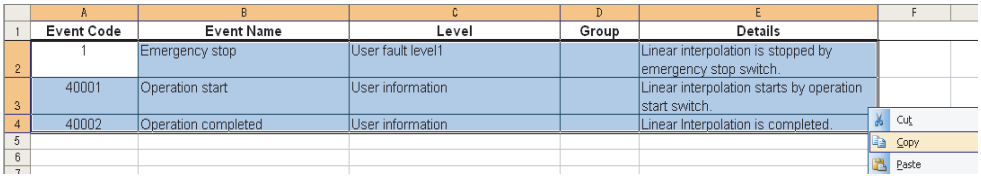


	Event Code	Event Name	Event Level	Group	Details
1	1	Emergency stop	User fault level1		Linear interpolation is stopped by emergency stop switch.
2	40001	Operation start	User information		Linear interpolation starts by operation start switch.
3	40002	Operation completed	User information		Linear Interpolation is completed.

Additional Information

You can edit the contents of the Event Settings Table with Microsoft Excel.

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

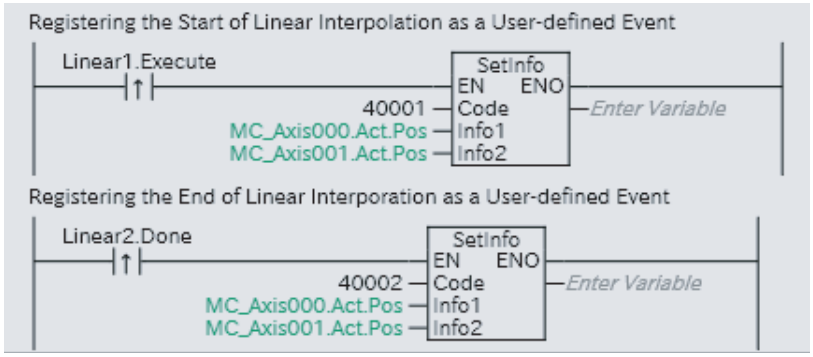


	A	B	C	D	E	F
1	Event Code	Event Name	Level	Group	Details	
2	1	Emergency stop	User fault level1		Linear interpolation is stopped by emergency stop switch.	
3	40001	Operation start	User information		Linear interpolation starts by operation start switch.	
4	40002	Operation completed	User information		Linear Interpolation is completed.	
5						
6						

3 Program the Create User-defined Information or Create User-defined Error instruction.

User-defined errors: Use the Create User-defined Error (SetAlarm) instruction.

User-defined information: Use the Create User-defined Information (SetInfo) instruction.



Precautions for Correct Use

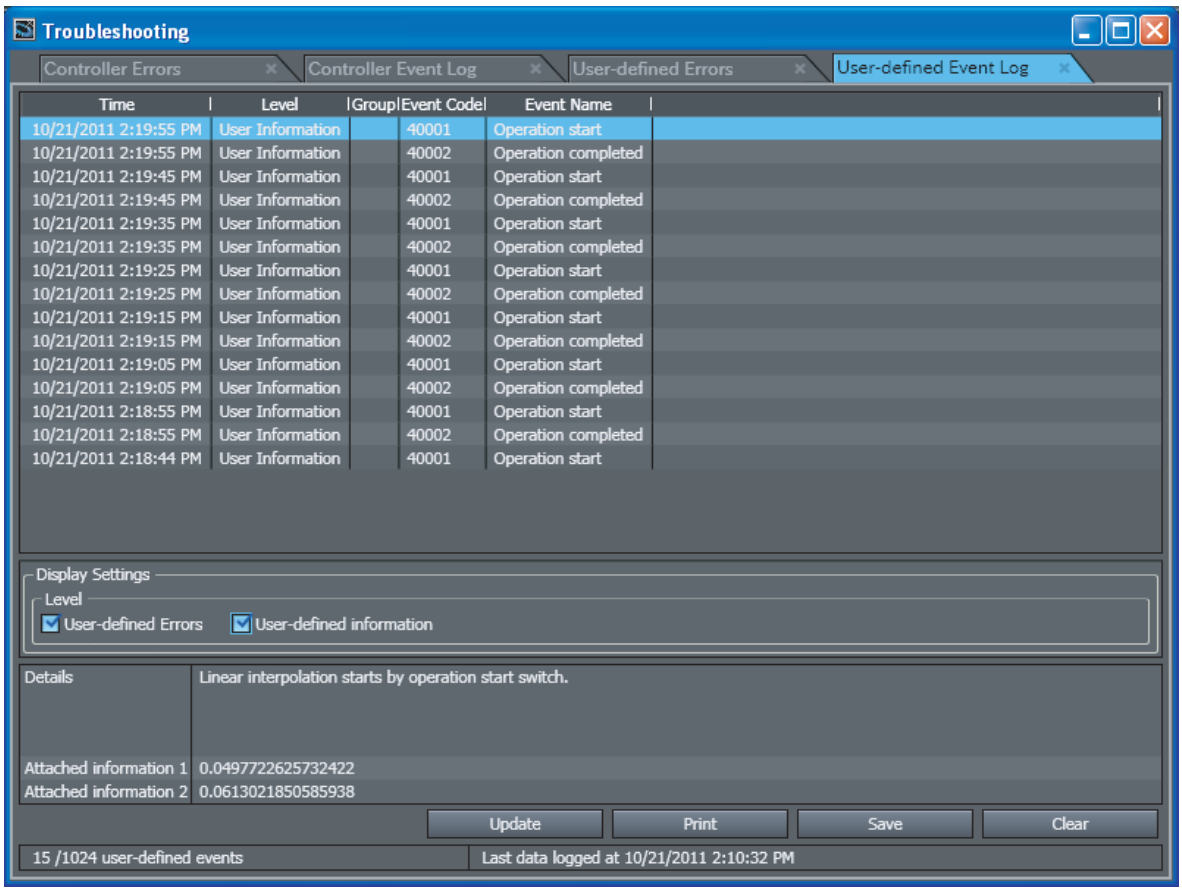
You must specify variables for *Info1* (Attached Information 1) and *Info2* (Attached Information 2) in the Create User-defined Error (SetAlarm) and Create User-defined Information (SetInfo) instructions. If you use a constant, a building error will occur. If you do not use the attached information, specify a dummy variable.

4 Transfer the Event Setting Table and user program to the CPU Unit.

● Checking for User-defined Events

1 Select ***Troubleshooting*** from the Tools Menu while online. Or, click the **Troubleshooting** Button in the toolbar.

You can view the status of user-defined events on the User-defined Errors and User-defined Event Log Tab Pages.



5-2 Protecting User Program Assets

The security functions are used to protect user program assets. The following table lists the security functions. This section provides an overview of these security functions.

Security measure	Application
Authentication of user program execution IDs	Prevention of the theft of assets
User program transfers with no restoration information	
Overall project file protection	
Operation authority verification	Prevention of incorrect operation
Write protection	
CPU Unit names	Prevention of incorrect connections

Additional Information

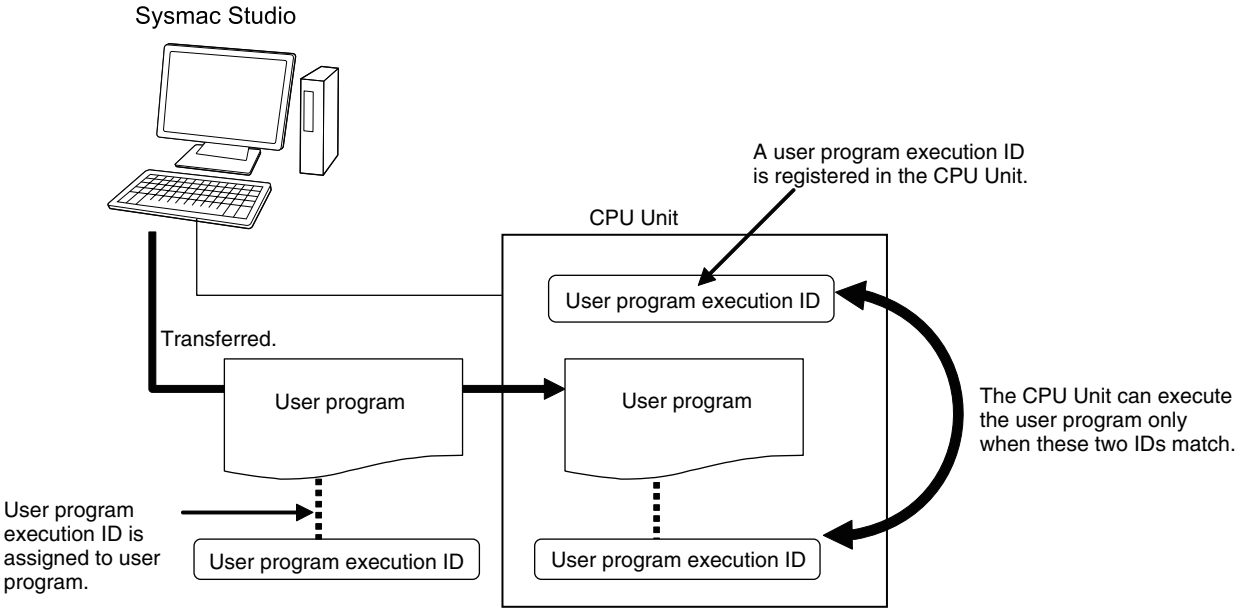
Refer to the *Sysmac Studio Version 1 Operation Manual* (Cat. No. W504) and the *NJ-series CPU Unit Software User's Manual* (Cat. No. W501) for details on the security functions.

5-2-1 Preventing Theft with Authentication of User Program Execution IDs

You can register a unique ID in the Controller in advance so that only the user program associated with that ID can be executed. This ID is called a user program execution ID.

You can use user program execution IDs to apply the following usage restrictions:

- Allow the execution of only a specific user program in a Controller.
- Prevent the use of the user program in a different Controller.



Precautions for Correct Use

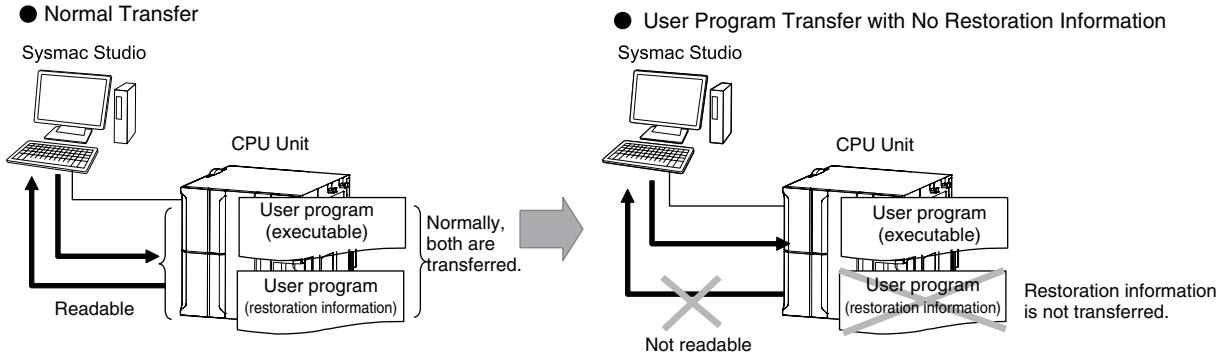
- A user program execution ID can be set only one time for a user program.
- Record the user program execution ID to ensure you do not lose it.
- We recommend that you backup the project file before you set the user program execution ID.

5-2-2 Preventing Theft By Transferring without User Program Restoration Information

Normally, when you transfer the user program from the Sysmac Studio to the CPU Unit, information is transferred to restore it back from the CPU Unit.

You can perform a transfer without the user program restoration information to prevent this restoration information from being transferred. This prevents the user program from being read from the CPU Unit.

This function is used to prevent theft of user program data when on-site maintenance of the user program is not required.



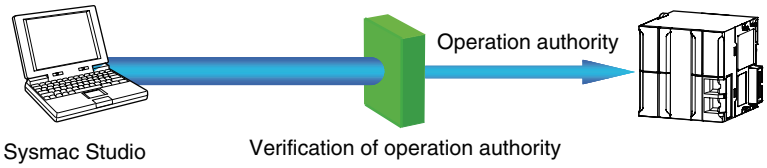
5-2-3 Protecting User Asset Information with Overall Project File Protection

You can apply a password to a project file when it is exported. This encrypts the project file and protects the user assets.

5-2-4 Preventing Incorrect Operation with Operation Authority Verification

Online operations are restricted by operation rights to prevent damage to equipment or injuries that may be caused by operating mistakes. You can register operation authority verification passwords in the CPU Unit in advance from the Sysmac Studio. When the Sysmac Studio goes online with the Controller and a password is entered, only the operations that match the operation authority category for that password are enabled.

The Administrator sets a password for each operation authority. Users are notified of the operation authority name and password according to their skills.



● Types of Operation Authorities

There are two types of operation authority for the Sysmac Studio, as given below.

- Administrator
- Maintainer

● Example of Operation Authority for Online Operations

(OK: Operation possible, VR: Verification required for each operation, NP: Operation not possible)

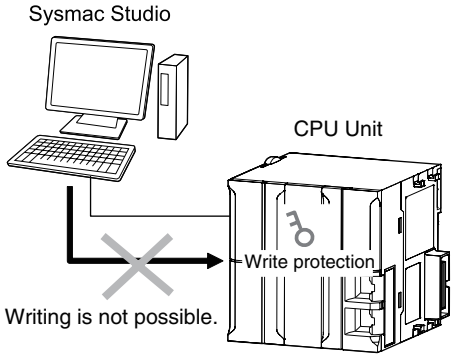
Controller operation	Administrator	Maintainer
Entering RUN mode	OK	VR
Entering PROGRAM mode	OK	VR
Online editing	OK	VR
Releasing access rights	OK	NP
Clearing memory (Memory All Clear)	OK	NP
Resetting the Controller	OK	NP
Resetting errors (troubleshooting)	OK	OK
Clearing event logs (error logs)	OK	OK
Controller clock operations	OK	NP
SD Memory Card operations	OK	OK

5-2-5 Preventing Write Operations from the Sysmac Studio with Write Protection

Write protection is used to prevent write operations from the Sysmac Studio.

Use one of the following two methods to enable write protection.

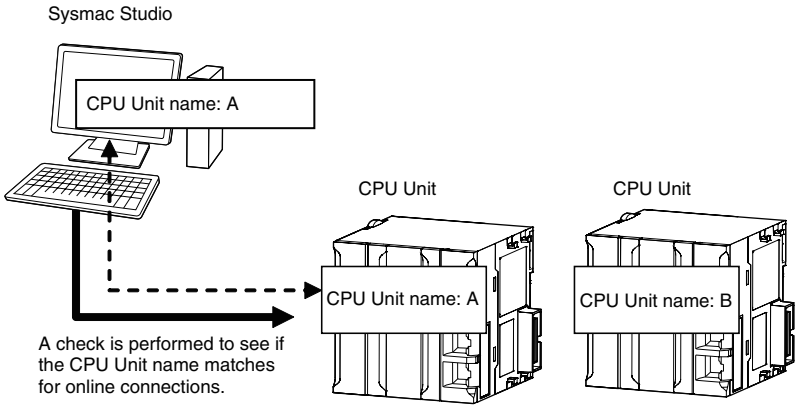
- You can automatically have write protection set when the power supply to the Controller is turned ON.
- In the Sysmac Studio, go online and select ***Security*** − ***Change Write Protect Switch*** from the Controller Menu to toggle write protection.



5-2-6 Using CPU Unit Names to Prevent Incorrect Connections from the Sysmac Studio

When going online to a CPU Unit from the Sysmac Studio, the CPU Unit name in the project is compared to the name of the CPU Unit being connected to.

This helps prevent incorrect connections to the CPU Unit from the Sysmac Studio. It is particularly effective when you connect over an EtherNet/IP network.



App

Appendices

The appendices describe common operating procedures for programming and debugging.

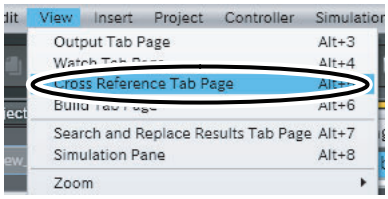
A-1 Using Cross References

You use cross references to display lists of the locations in a project where variables or I/O ports for EtherCAT slaves are used.

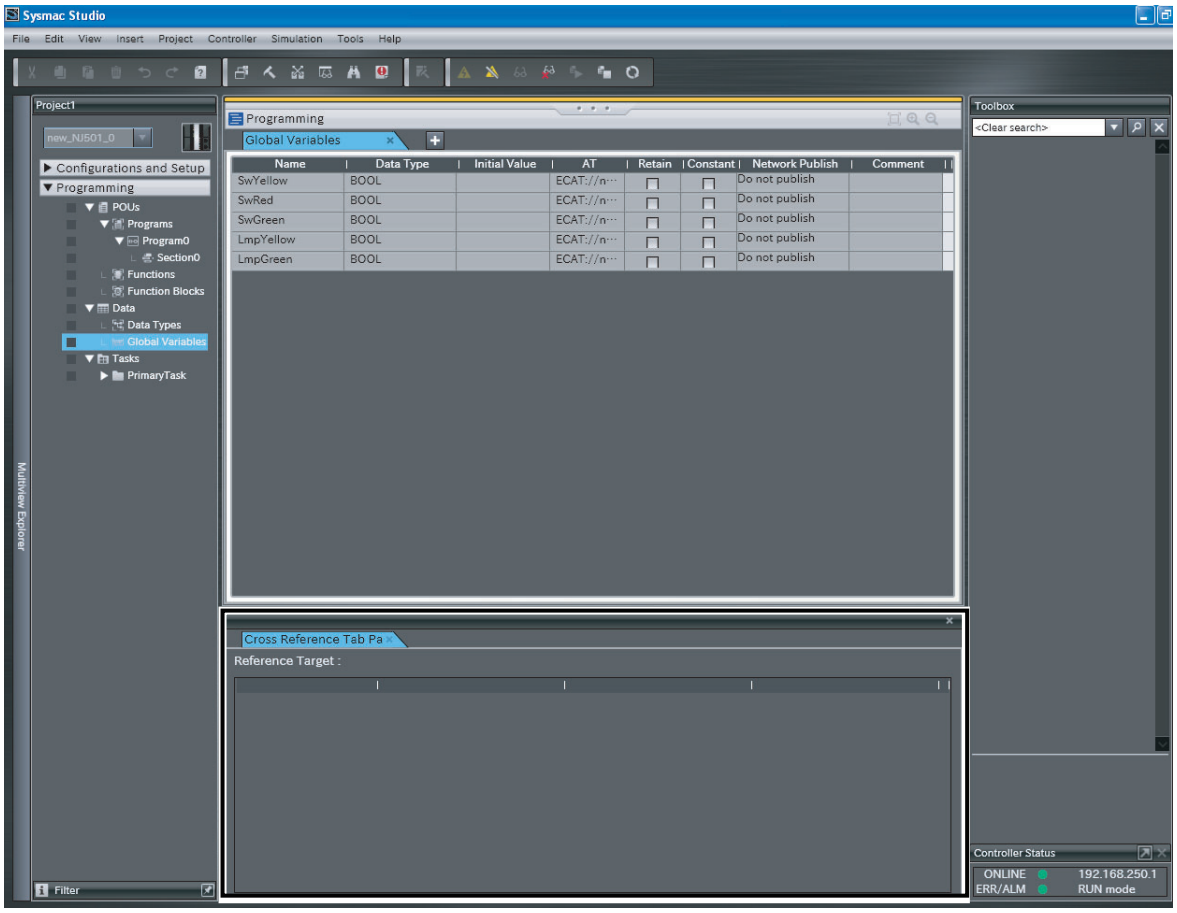
This appendix uses the project that was created in the startup procedures to show how to use cross references. The following procedure shows how to use cross references to see where the *SwRed* variable is used in the Ladder Editor. The *SwRed* variable is registered in the global variable table.

1 Open the global variable table.

2 Select ***Cross Reference Tab Page*** from the View Menu.

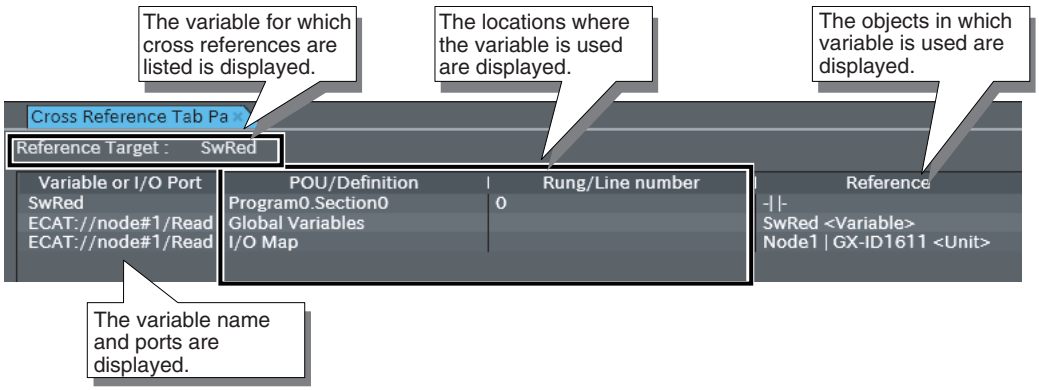


The Cross Reference Tab Page is displayed.

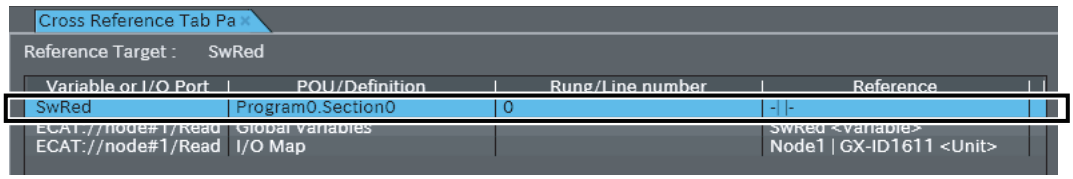


3 Click the *SwRed* variable in the global variable table.

The locations where the *SwRed* variable is used are displayed in the Cross Reference Tab Page. If the variable is set as a device variable, the I/O port for the variable is also displayed.

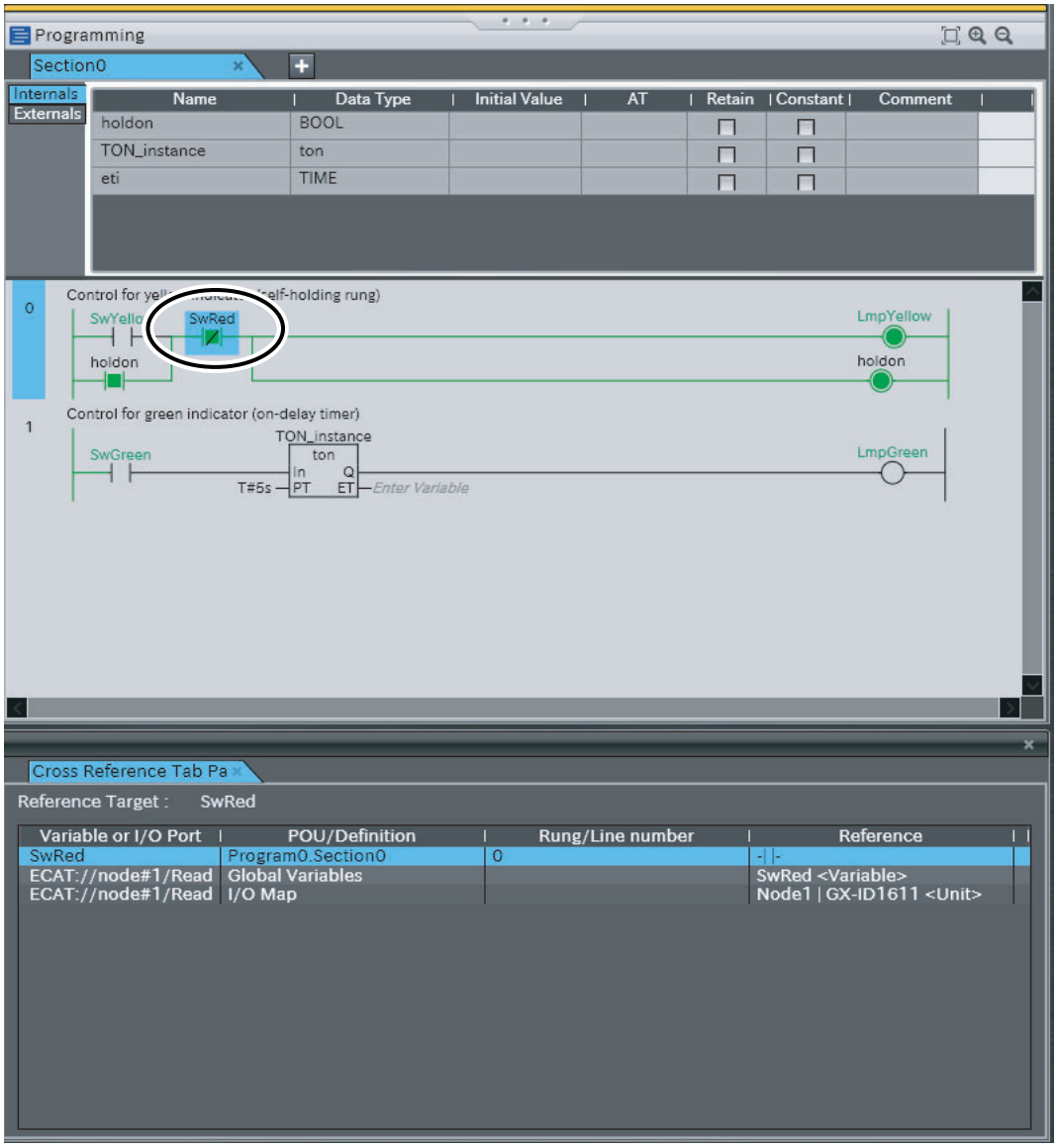


4 Click the line with the *SwRed* variable on the Cross Reference Tab Page.



App

The location where the *SwRed* variable is used is displayed in the Ladder Editor and the object in which it is used is selected.



Additional Information

In addition to the global variable table, you can also select variables in the Ladder Editor or I/O Map to find cross references for them.

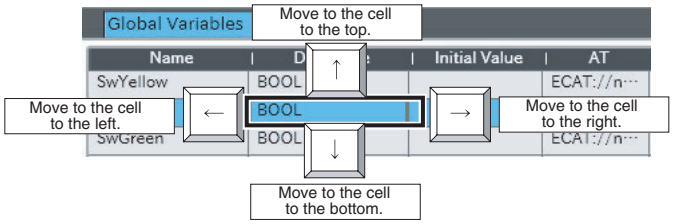
A-2 Useful Functions for Editing Variable Tables

These functions are useful when you edit variable tables: intuitive keyboard shortcuts for editing, variable name input assistance, and data type input assistance.

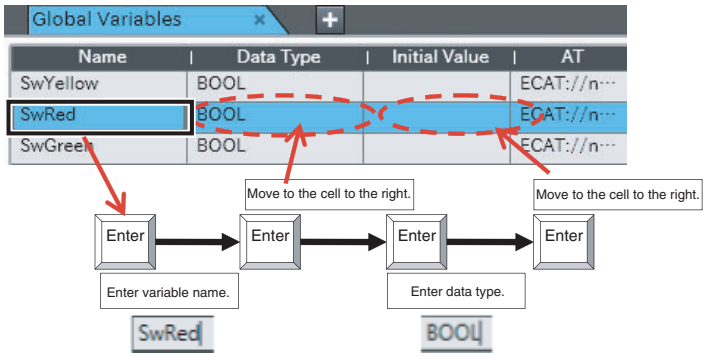
Intuitive Keyboard Shortcuts

You can intuitively use keyboard shortcuts to create new variables and edit existing variables quickly and easily.

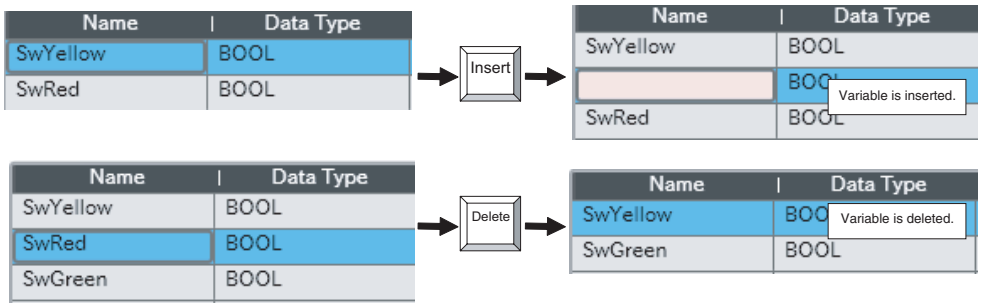
- Use the **Arrow** Keys to move between cells.



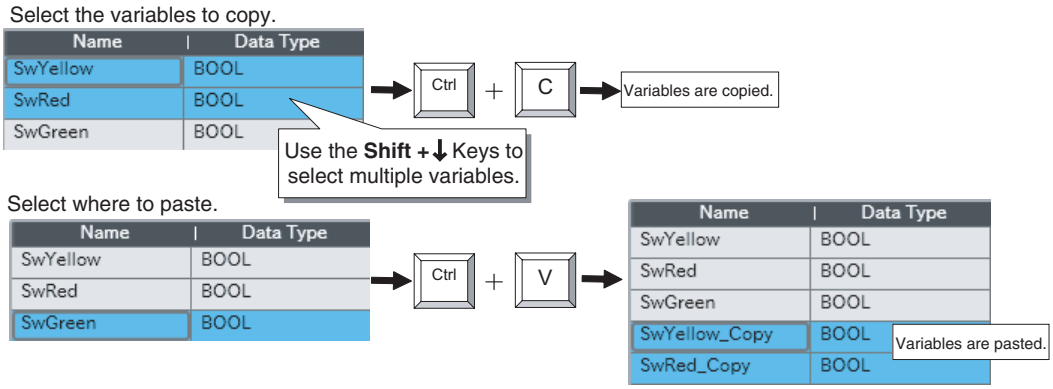
- Use the **Enter** Key to move to the cell to the right.



- Use the **Insert** Key to insert a variable and the **Delete** Key to delete a variable.



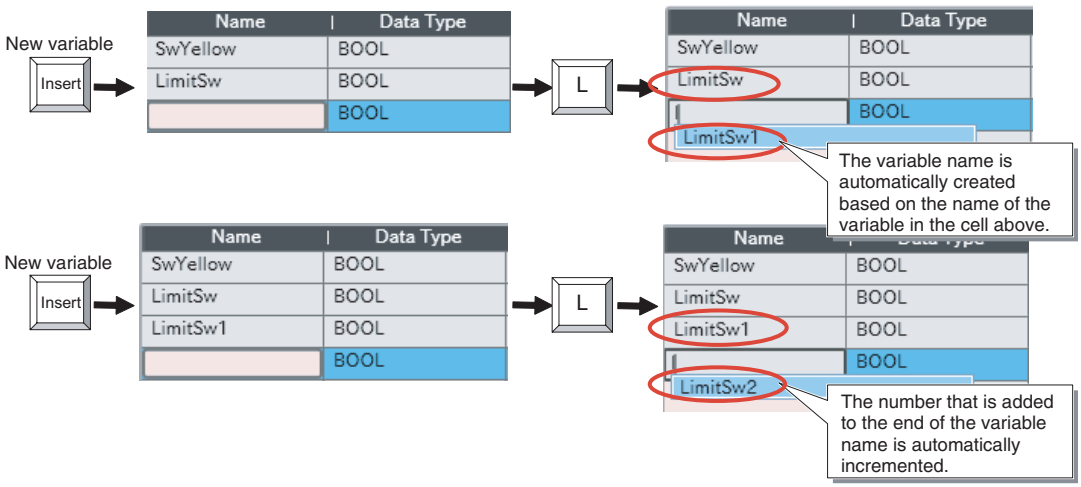
- Use the **Ctrl + C** Keys to copy a variable and **Ctrl + V** Keys to paste a variable.



Variable Name Input Assistance

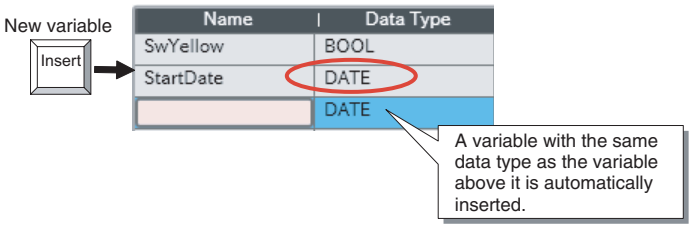
When a variable is created, a number is automatically added to the end of the variable name.

This is useful when creating similar variables.

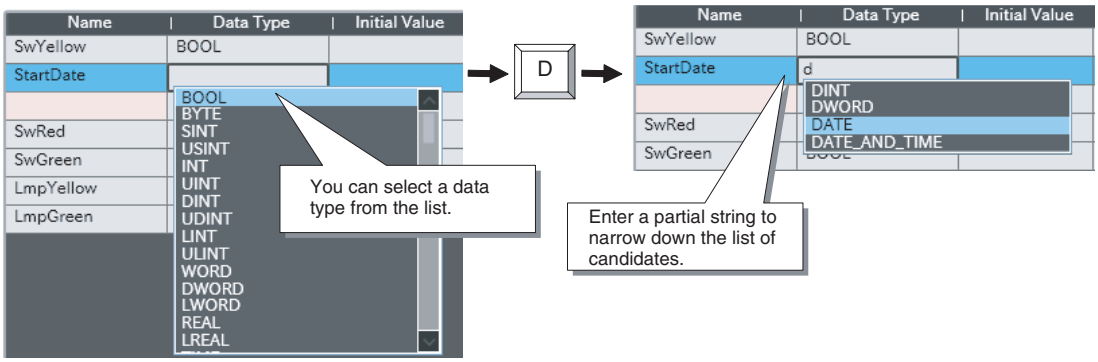


Data Type Input Assistance

- When a variable is created, a variable with the same data type as the variable above it is automatically inserted.



- You can select a data type from the list.

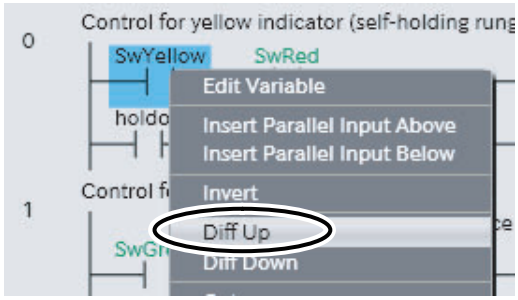


A-3 Frequently Used Programming Operations

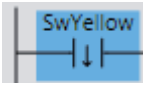
This section describes some frequently used programming operations.

Entering Differentiated Inputs

● Upward Differentiation

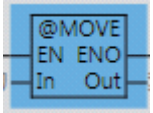
Right-click the input and select ***Diff Up*** from the menu.	 ▼ 

● Downward Differentiation

Right-click the input and select ***Diff Down*** from the menu.	

Entering Upward Differentiation Options for FUN Instructions

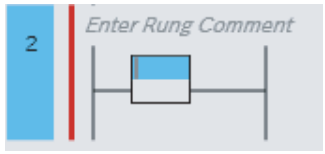
Enter @ at the beginning of the instruction name.

Click the connecting line and press the  **I** Key.	 A function is inserted, and you can enter the function name.
▼	
Enter *@MOVE*.	

Entering a One-second Clock Pulse Input

Click the connecting line and press the I Key.

I Key.

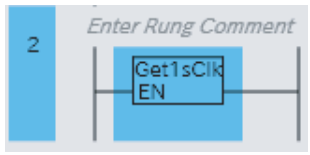


A function is inserted, and you can enter the function name.

▼

Enter *Get1sClk* and press the **Enter** Key twice.

Enter Enter



▼

Then, add a program output.

The clock pulse period is 100 us, 1 ms, 10 ms, 20 ms, 100 ms, 1 s, or 1 min.

Entering an Always TRUE Input

Click the connecting line and press the C Key.

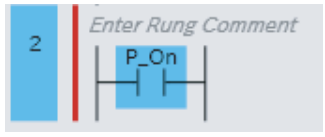
C Key.

An input is inserted.

▼

Enter *P_On*, and then press the **Enter** Key twice to confirm.

Enter Enter



Entering the First RUN Period Flag

Click the connecting line and press the C Key.

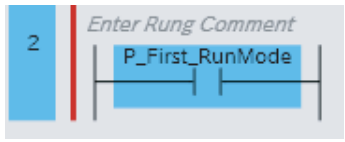
C Key.

An input is inserted.

▼

Enter *P_First_RunMode*, and then press the **Enter** Key twice to confirm.

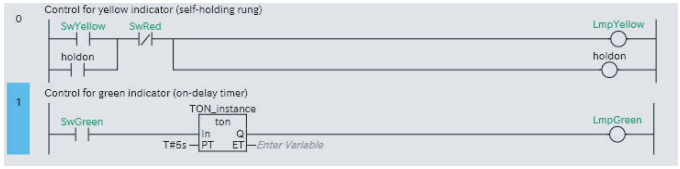
Enter Enter



App

Copying and Pasting Rungs

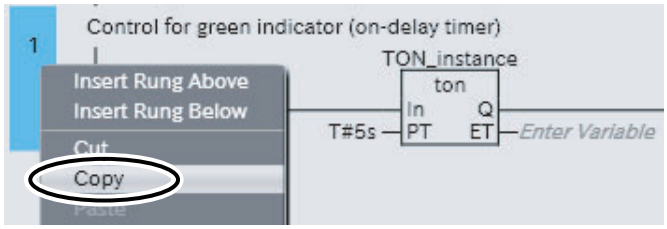
Select the rung to copy.*



▼

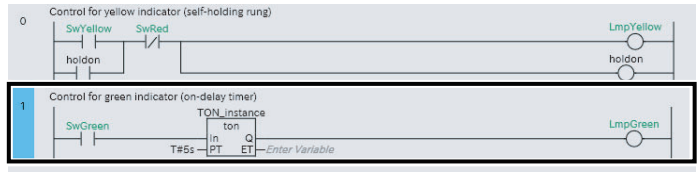
Right-click and select ***Copy*** from the menu.

Press the Ctrl **Ctrl** Key + the C **C** Key.



▼

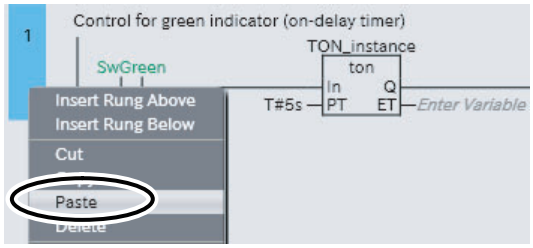
Select the location where you want to paste the rung.



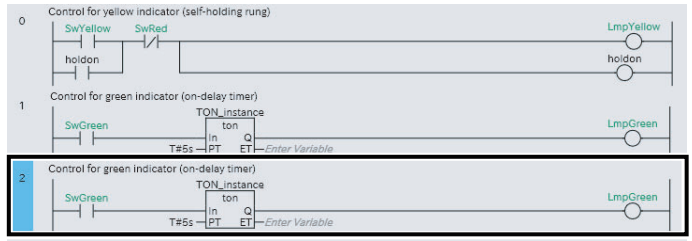
▼

Right-click and select ***Paste*** from the menu.

Press the Ctrl **Ctrl** Key + the V **V** Key.



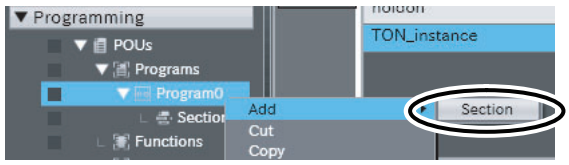
The copied rung is pasted below the selected rung.



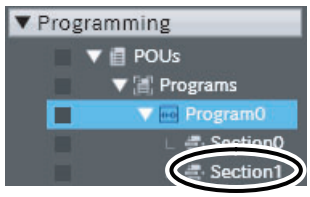
* To select multiple rungs, press and hold the **Shift** Key while you select additional rungs.

Adding Sections

Right-click the program you want to add a section to in the Multiview Explorer and select ***Add*** – ***Section*** from the menu.

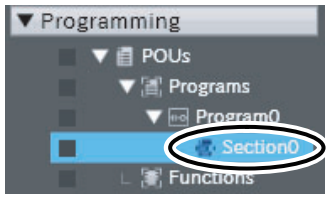


A new section is added.



Copying and Pasting Sections

Select the section to copy in the Multiview Explorer.



▼

Right-click and select ***Copy*** from the menu.

Press the Ctrl **Ctrl** Key + the C **C** Key.

▼

Right-click the program where you want to paste the section in the Multiview Explorer, and select ***Paste*** from the menu.

Press the Ctrl **Ctrl** Key + the V **V** Key.

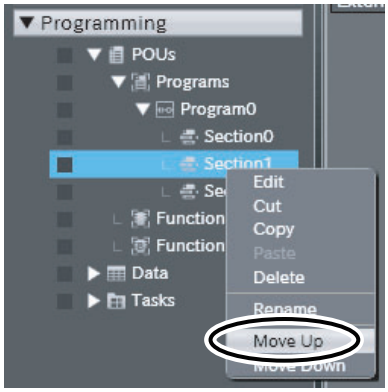
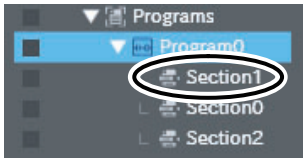
The section is pasted at the bottom of the selected program.

App

Changing the Order of Sections

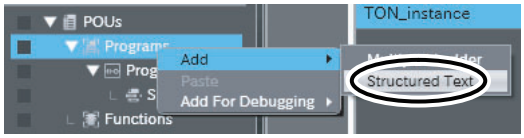
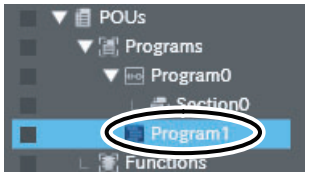
Sections are executed from top to bottom in the order that they are displayed in the Multiview Explorer. To change the order of execution, you must change the order of the sections.

This section gives the procedure for moving a section up in the order.

Select the section to move in the Multiview Explorer.	
▼	
Right-click and select ***Move Up*** from the menu. Or, drag the section to the location you want it.	 The section moves up. 

Adding POUs

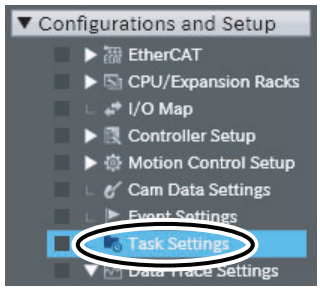
You use the following procedure to add an ST language POU.

Right-click **Programs** in the Multiview Explorer, and select ***Add*** – ***Structured Text*** from the menu.	 An ST language POU is added to the programs. 

Assigning Programs to Tasks

To execute the new POU, it must be assigned to a task. Use the following procedure to assign *Program1* (the program that was added) to the primary periodic task.

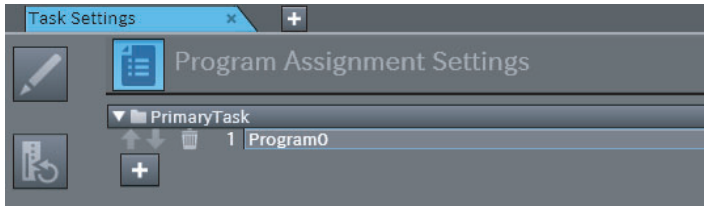
Double-click **Task Settings** under **Configurations and Setup** in the Multiview Explorer.



▼

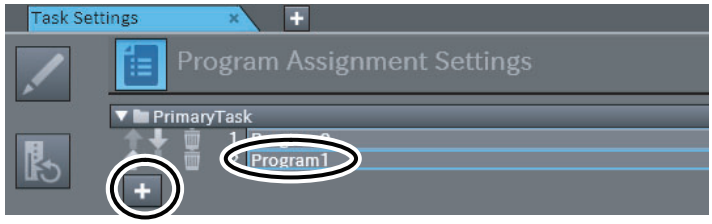
Click the  Button in the Edit Pane.

The Program Assignment Settings Display appears.



▼

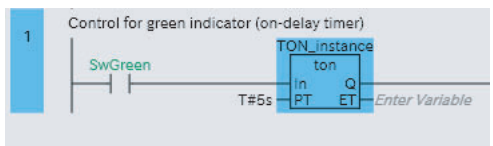
Click the  Button and select *Program1* from the list.



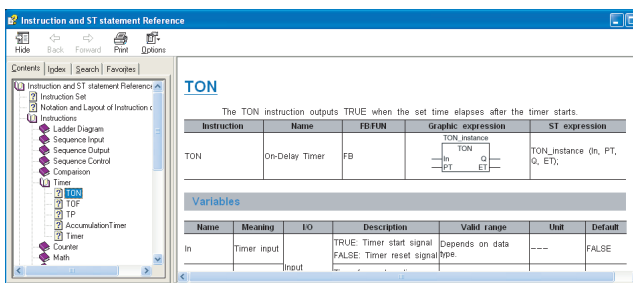
Referencing Detailed Instruction Information

● Displaying the Instructions Reference

Click the instruction, and then click the F1 Button.

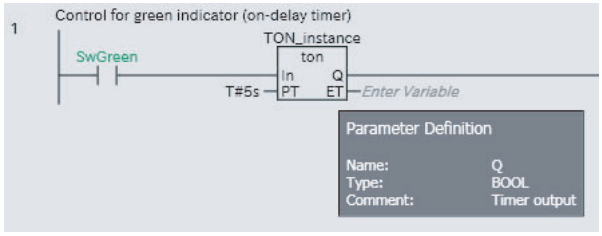


The Instruction Reference is displayed.



App

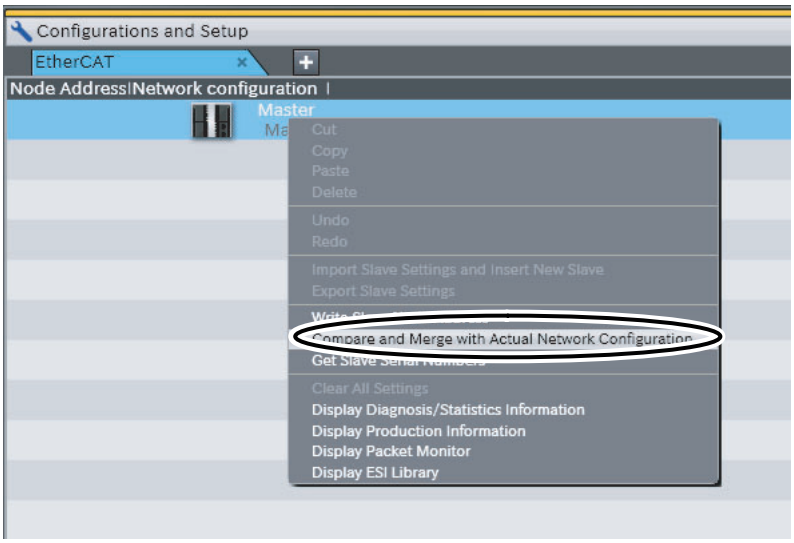
● Displaying Detailed Information on Input and Output Variables of Instructions

Move the mouse cursor over any input or output variable of an FB or FUN instruction.	Details are displayed on the Ladder Editor. 

A-4 Creating an Online Network Configuration

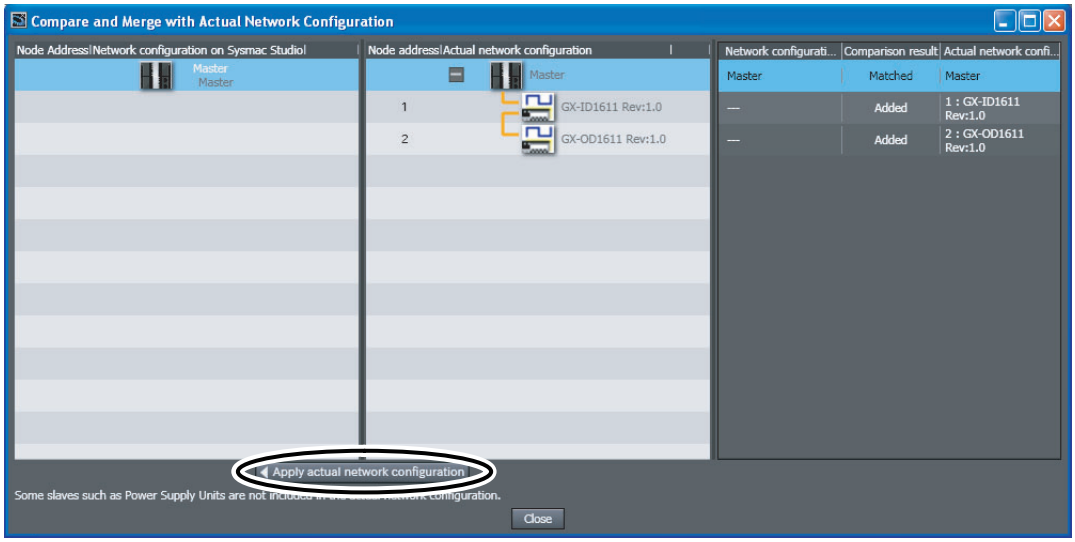
If the actual EtherCAT network configuration is already connected, you can automatically create the virtual network configuration in the Sysmac Studio based on the actual network configuration.

1 While online, right-click the master in the EtherCAT Tab Page and select ***Compare and Merge with Actual Network Configuration***.

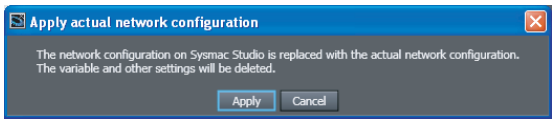


The Compare and Merge with Actual Network Configuration Window is displayed.

2 Click the **Apply actual network configuration** Button in the Compare and Merge with Actual Network Configuration Window.



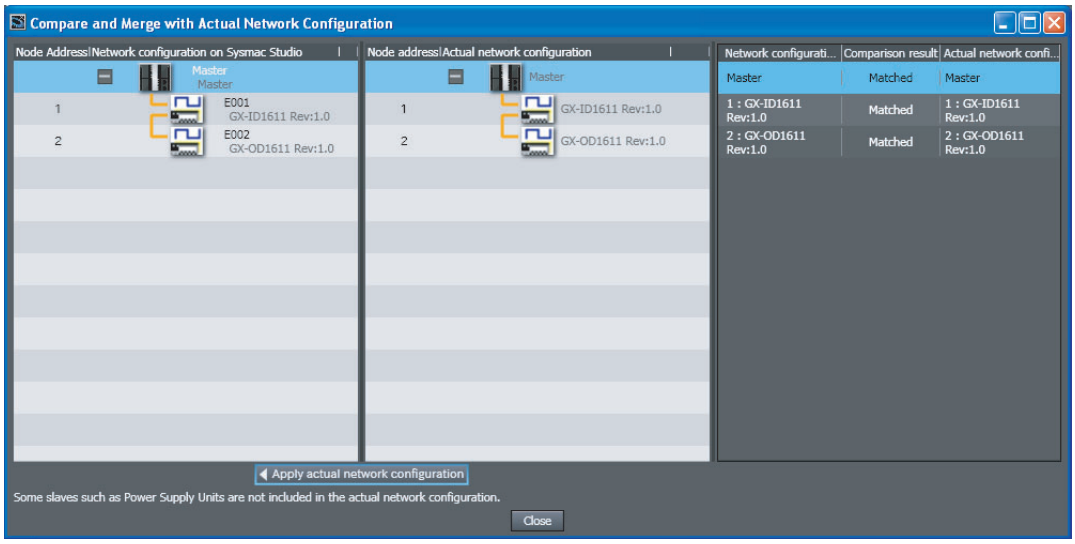
The following dialog box is displayed.



3 Click the **Apply** Button.

App

The actual network configuration is registered as the Sysmac Studio network configuration.



A-5 Differences between Online and Offline Debugging

There are different debugging functions available during online and offline debugging. The following table lists the differences between online and offline debugging.

Operation for debugging	Online debugging	Offline debugging
Monitoring	Supported.	Supported.
Monitoring on a Watch Tab Page	Supported.	Supported.
Monitoring in the I/O Map	Supported.	Supported.
Controlling BOOL variables (Set/Reset)	Supported.	Supported.
Forced refreshing (TRUE/FALSE/Cancel)	Supported.	Supported.
Changing present values of data	Supported.	Supported.
Clearing memory (Memory All Clear)	Supported.	Not supported.
Cross-reference pop-ups	Supported.	Supported.
Online editing	Supported.	Supported.
Monitoring Controller status	Supported.	Not supported.
Monitoring task execution status	Supported.	Supported.
Monitoring axis status (MC Monitor Table)	Supported.	Supported.
Changing the operating mode	Supported.	Not supported.
Resetting the Controller	Supported.	Not supported.
Data tracing	Supported.	Supported.
Setting triggers	Supported.	Supported.
Setting variables to sample	Supported.	Supported.
Starting and stopping tracing	Supported.	Supported.
Displaying trace results	Supported.	Supported.
Exporting trace results	Supported.	Supported.
Creating 3D device models	Supported.	Supported.
Displaying in a digital/analog chart	Supported.	Supported.
Displaying 3D axis paths	Supported.	Supported.
Monitoring task execution times	Supported.	Supported.
Estimating execution processing times	Not supported.	Supported.
Debugging with program simulations	Not supported.	Supported.
Setting what to simulate	Not supported.	Supported.
Changing the simulation speed	Not supported.	Supported.
Setting breakpoints	Not supported.	Supported.
Step execution	Not supported.	Supported.
Troubleshooting	Supported.	Not supported.
Monitoring error information	Supported.	Supported.
Displaying error logs	Supported.	Supported.
Event Setting Table	Supported.	Supported.
Monitoring user memory usage	Supported.	Supported.
Setting clock information	Supported.	Not supported.
Releasing access rights	Supported.	Not supported.

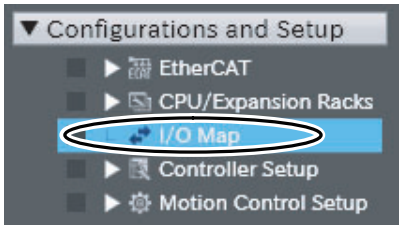
A-6 Forced Refreshing on the I/O Map

On the I/O Map, you can perform forced refreshing without defining variables or creating an algorithm. Forced refreshing on the I/O Map is therefore convenient to check the I/O wiring of EtherCAT slaves or CJ-series Units. You use the following procedure to perform forced refreshing on the I/O Map.

Forced Refreshing Procedure

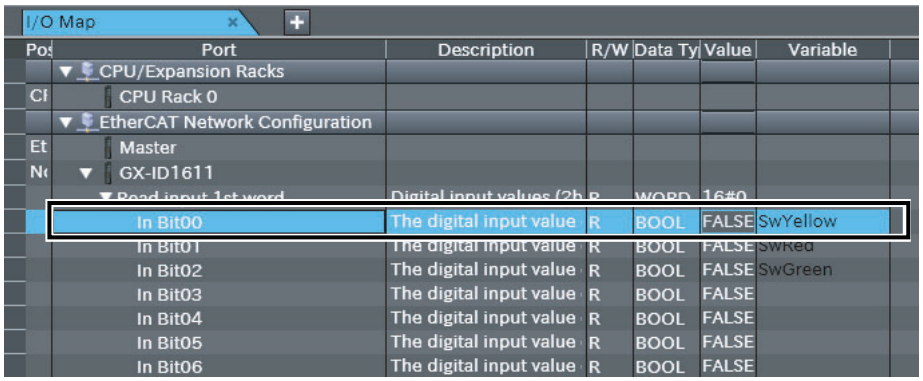
1 Go online with the Controller. Or, start the Simulator.

2 Double-click **I/O Map** under **Configurations and Setup** in the Multiview Explorer.

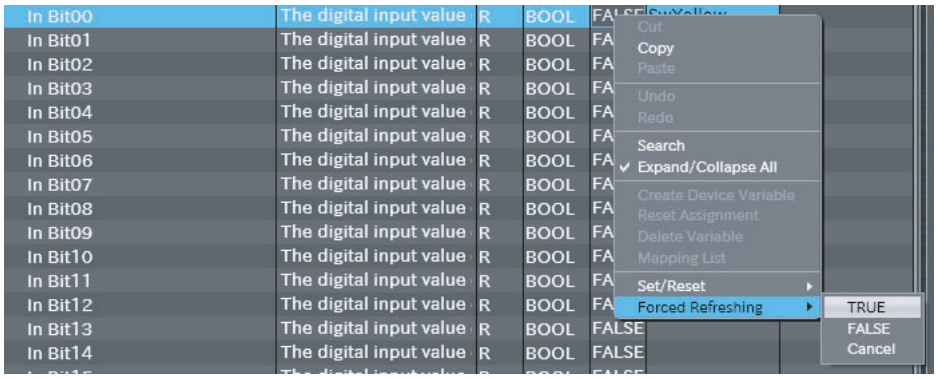


The I/O Map is displayed.

3 Select the I/O port or variable for forced refreshing on the I/O Map.



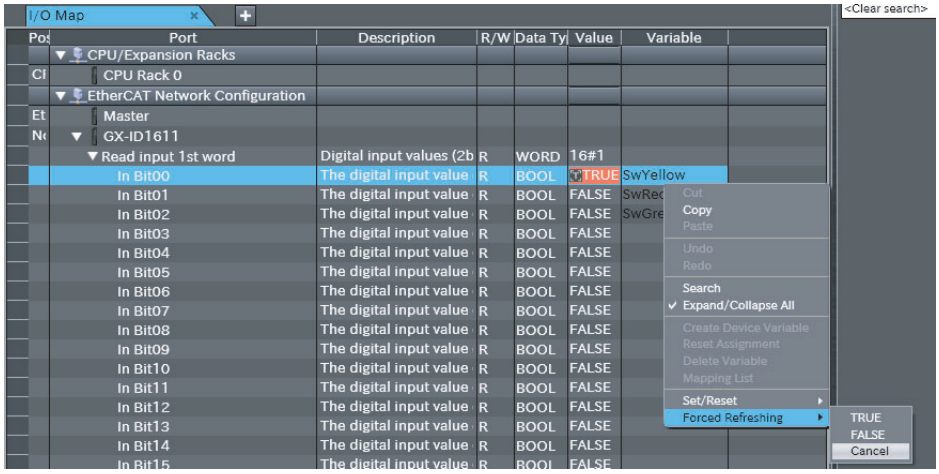
4 Right-click and select ***Forced Refreshing*** – ***TRUE*** or ***Forced Refreshing*** – ***FALSE*** from the menu.



If you select ***TRUE***, the variable or I/O port changes to TRUE. If you select ***FALSE***, the variable or I/O port changes to FALSE. **TRUE** or **FALSE** is displayed in the Value Column of the I/O port or variable. An icon also appears by the I/O port or variable that shows the forced status.

Canceling Forced Status

1 Right-click the I/O port or variable for which to cancel forced status and select ***Forced Refreshing*** – ***Cancel*** from the menu.



The forced value for the I/O port or variable is canceled and the forced status icon disappears. The TRUE/FALSE status of the I/O port or variable will change according to the user program or system configuration.

App

OMRON Corporation **Industrial Automation Company**
Tokyo, 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
One Commerce Drive Schaumburg,
IL 60173-5302 U.S.A.
Tel: (1) 847-843-7900/Fax: (1) 847-843-7787

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

OMRON (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. W513-E1-01 1111