OMRON

Automation Software

Sysmac Studio

Project Version Control Function Operation Manual

SYSMAC-SE2□□□
SYSMAC-TA4□□L

SYSMAC
always in control

W589-E1-04

NOTE

1. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, mechanical, electronic, photocopying, recording, or otherwise, without the prior written permission of OMRON.
2. No patent liability is assumed with respect to the use of the information contained herein. Moreover, because OMRON is constantly striving to improve its high-quality products, the information contained in this manual is subject to change without notice.
3. Every precaution has been taken in the preparation of this manual. Nevertheless, OMRON assumes no responsibility for errors or omissions.
   Neither is any liability assumed for damages resulting from the use of the information contained in this publication.

Trademarks

- Sysmac and SYSMAC are trademarks or registered trademarks of OMRON Corporation in Japan and other countries for OMRON factory automation products.
- Microsoft, Windows, Excel, Visual Basic, and Microsoft Edge are either registered trademarks or trademarks of Microsoft Corporation in the United States and other countries.
- EtherCAT® is registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.
- ODVA, CIP, CompoNet, DeviceNet, and EtherNet/IP are trademarks of ODVA.
- The SD and SDHC logos are trademarks of SD-3C, LLC.
- NVIDIA, the NVIDIA logo, GeForce, and the GeForce logo are the trademarks or registered trademarks of NVIDIA Corporation in the USA and other countries.
- ATI™ and Radeon™ are the trademarks of Advanced Micro Devices, Inc.
- Celeron, Intel and Intel Core are trademarks of Intel Corporation in the U.S. and / or other countries.
- Git and the Git logo are either registered trademarks or trademarks of Software Freedom Conservancy, Inc., corporate home of the Git Project, in the United States and/or other countries.
- JavaScript® is a trademark or registered trademark of Oracle Corporation and its subsidiaries and affiliates in the United States and other countries.
- EPLAN and EPLAN Electric P8 are registered trademarks of EPLAN GmbH & Co.KG.

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

Copyrights

Microsoft product screen shots used with permission from Microsoft.

Introduction

Thank you for purchasing a Sysmac Studio Team Development Option.
This manual contains information that is necessary to use the Sysmac Studio Team Development Option. Please read this manual and make sure you understand the functionality and performance of the Sysmac Studio Team Development Option before you attempt to use it in a control system.
Keep this manual in a safe place where it will be available for reference during operation.

Intended Audience

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

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

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

Applicable Products

This manual covers the following products.

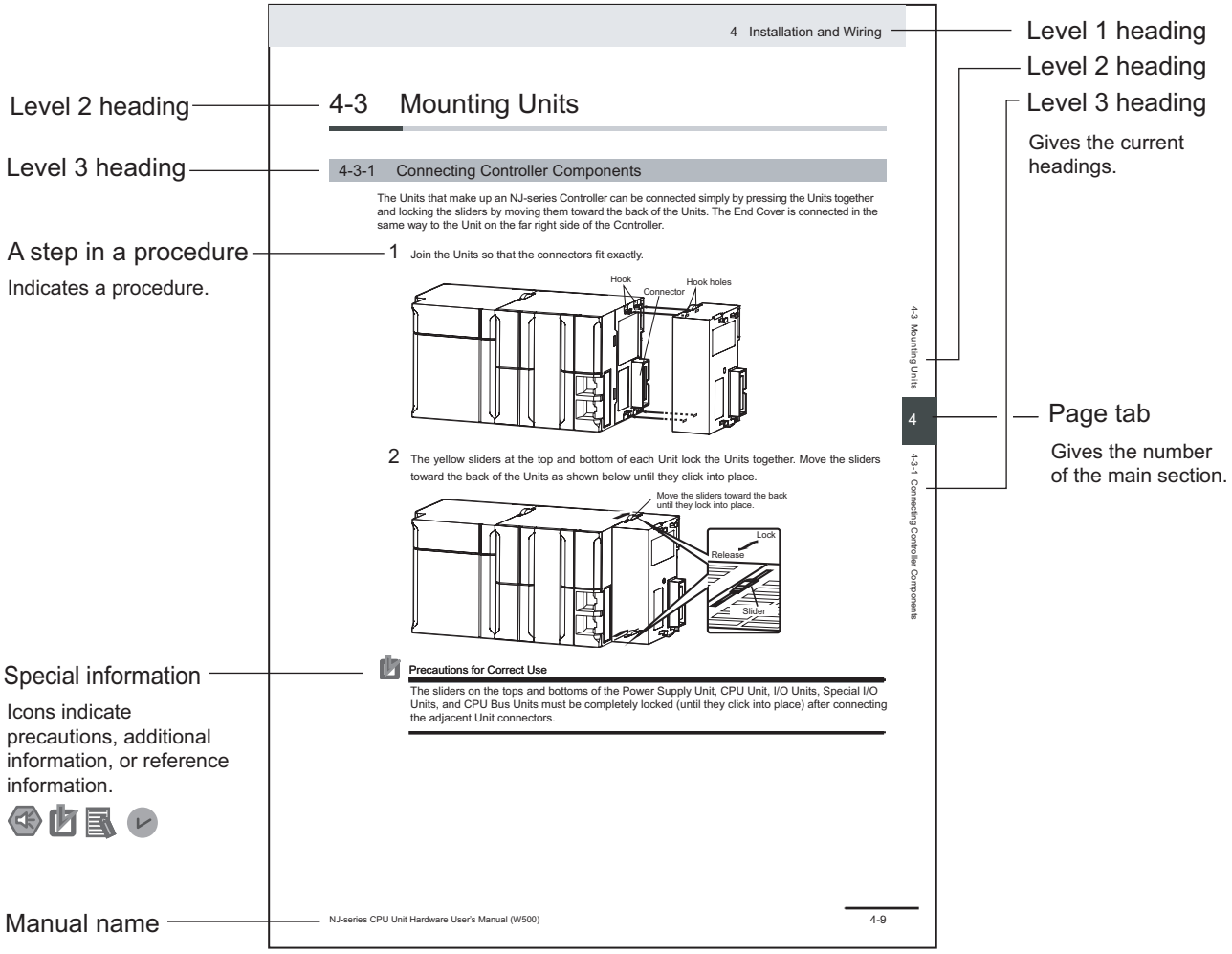
- Sysmac Studio Team Development Option

Part of the specifications and restrictions for the CPU Units are given in other manuals.
Refer to *Related Manuals* on page 14.

Manual Structure

Page Structure

The following page structure is used in this manual.



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

Special Information

Special information in this manual is classified as follows:

Precautions for Safe Use

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

Precautions for Correct Use

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

 **Additional Information**

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

 **Version Information**

Information on differences in specifications and functionality for Controllers and Units with different unit versions and for different versions of Support Software is given.

Precaution on Terminology

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

Sections in this Manual

CONTENTS

Section 1 Features and Specifications

Section 2 Software Setup and Basic Settings

Section 3 Basic Operations of Version Control

Section 4 Parts of the Window

Appendices

Index

Terms and Conditions Agreement

WARRANTY

- The warranty period for the Software is one year from the date of purchase, unless otherwise specifically agreed.
- If the User discovers defect of the Software (substantial non-conformity with the manual), and return it to OMRON within the above warranty period, OMRON will replace the Software without charge by offering media or download from OMRON's website. And if the User discovers defect of media which is attributable to OMRON and return it to OMRON within the above warranty period, OMRON will replace defective media without charge. If OMRON is unable to replace defective media or correct the Software, the liability of OMRON and the User's remedy shall be limited to the refund of the license fee paid to OMRON for the Software.

LIMITATION OF LIABILITY

- THE ABOVE WARRANTY SHALL CONSTITUTE THE USER'S SOLE AND EXCLUSIVE REMEDIES AGAINST OMRON AND THERE ARE NO OTHER WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO, WARRANTY OF MERCHANTABILITY OR FITNESS FOR PARTICULAR PURPOSE. IN NO EVENT, OMRON WILL BE LIABLE FOR ANY LOST PROFITS OR OTHER INDIRECT, INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF USE OF THE SOFTWARE.
- OMRON SHALL HAVE NO LIABILITY FOR DEFECT OF THE SOFTWARE BASED ON MODIFICATION OR ALTERNATION TO THE SOFTWARE BY THE USER OR ANY THIRD PARTY. OMRON SHALL NOT BE RESPONSIBLE AND/OR LIABLE FOR ANY LOSS, DAMAGE, OR EXPENSES DIRECTLY OR INDIRECTLY RESULTING FROM THE INFECTION OF OMRON PRODUCTS, ANY SOFTWARE INSTALLED THEREON OR ANY COMPUTER EQUIPMENT, COMPUTER PROGRAMS, NETWORKS, DATABASES OR OTHER PROPRIETARY MATERIAL CONNECTED THERETO BY DISTRIBUTED DENIAL OF SERVICE ATTACK, COMPUTER VIRUSES, OTHER TECHNOLOGICALLY HARMFUL MATERIAL AND/OR UNAUTHORIZED ACCESS.
- OMRON SHALL HAVE NO LIABILITY FOR SOFTWARE DEVELOPED BY THE USER OR ANY THIRD PARTY BASED ON THE SOFTWARE OR ANY CONSEQUENCE THEREOF.

APPLICABLE CONDITIONS

USER SHALL NOT USE THE SOFTWARE FOR THE PURPOSE THAT IS NOT PROVIDED IN THE ATTACHED USER MANUAL.

CHANGE IN SPECIFICATION

The software specifications and accessories may be changed at any time based on improvements and other reasons.

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.

Safety Precautions

Definition of Precautionary Information

The following notation is used in this manual to provide precautions required to ensure safe usage of the Sysmac Studio. The safety precautions that are provided are extremely important to safety. Always read and heed the information provided in all safety precautions.
The following notation is used.

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

Symbols

The ⃠ symbol indicates operations that you must not do.
The specific operation is shown in the ⃠ symbol and explained in text.
This example indicates prohibiting disassembly.

The △ symbol indicates precautions (including warnings).
The specific operation is shown in the △ symbol and explained in text.
This example indicates a precaution for electric shock.

The △ symbol indicates precautions (including warnings).
The specific operation is shown in the △ symbol and explained in text.
This example indicates a general precaution.

The ● symbol indicates operations that you must do.
The specific operation is shown in the ● symbol and explained in text.
This example shows a general precaution for something that you must do.

Warnings

⚠ WARNING

To prevent computer viruses, install antivirus software on a computer where you use this software. Make sure to keep the antivirus software updated.

Keep your computer's OS updated to avoid security risks caused by a vulnerability in the OS.

Always use the highest version of this software to add new features, increase operability, and enhance security.

Manage usernames and passwords for this software carefully to protect them from unauthorized uses.

Set up a firewall (E.g., disabling unused communication ports, limiting communication hosts, etc.) on a network for a control system and devices to separate them from other IT networks. Make sure to connect to the control system inside the firewall.

Use a virtual private network (VPN) for remote access to a control system and devices from this software.

During Use of the Version Control System

When you manage a project using the version control system, do not use the version control system's functions directly from Windows Explorer on files that compose the project. Doing so may cause the loss of consistency among files that compose the project, and the control system may perform unexpected operation.

If you use the version control system to develop programs with multiple developers, check them for proper execution before you use them for actual operation.

Regulations and Standards

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/.

Versions

Hardware revisions and unit versions are used to manage the hardware and software in NJ/NX-series Units, NY-series Industrial PCs, and EtherCAT slaves.
Refer to *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for details on versions.

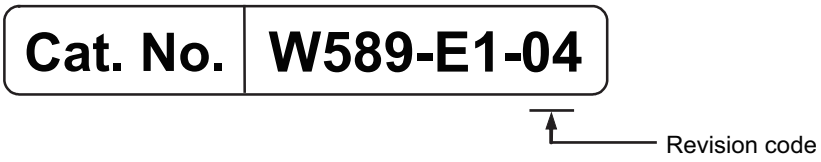
Related Manuals

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

Manual name	Cat. No.	Model numbers	Application	Description
Sysmac Studio Project Version Control Function Operation Manual	W589	SYSMAC-SE2□□□ SYSMAC-TA4□□□	Learning about the Sysmac Studio project version control function and its operating procedures.	Provides an introduction to the Sysmac Studio project version control function along with its installation method, basic operations, execution method for the main functions, and other information.
Sysmac Studio Version 1 Operation Manual	W504	SYSMAC-SE2□□□	Learning about the operating procedures and functions of the Sysmac Studio.	Describes the operating procedures of the Sysmac Studio.
Sysmac Studio Drive Functions Operation Manual	I589	SYSMAC-SE2□□□ SYSMAC-DE□□□L	Learning about the Servo Drive related functions of the Sysmac Studio.	Describes the Servo Drive related operating procedures and functions among those of the Sysmac Studio.
NJ/NX-series CPU Unit Software User's Manual	W501	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning how to program and set up an NJ/NX-series CPU Unit. Mainly software information is provided.	The following information is provided on a Controller built with an NJ/NX-series CPU Unit. • CPU Unit operation • CPU Unit features • Initial settings • Programming based on IEC 61131-3 language specifications
NA-series Programmable Terminal Software User's Manual	V118	NA5-□W□□□□	Learning about NA-series PT pages and object functions.	Describes the pages and object functions of the NA-series Programmable Terminals.

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	October 2017	Original production
02	March 2019	Corrected mistakes.
03	January 2020	Revised for the support of Sysmac Studio (64 bit)
04	August 2022	Revisions for adding safety precautions regarding security.

1

Features and Specifications

This section describes the features and specifications of the Sysmac Studio version control function that is supported by the *Sysmac Studio Team Development Option*.

1-1 Introduction

The Sysmac Studio project version control function (which is called version control function hereafter) controls the change record of a Sysmac Studio project by recording "who changed what and when." This function realizes various control capabilities by combining the Sysmac Studio with an open source software version control system that is commonly used in software development.
The Sysmac Studio version control function effectively solves problems such as the following, which you may encounter in large-scale development of production machines.

- The complexity of source code management due to increase in the scale of program development
- The increase in the workload of change management due to the increased variation of production machines
- The increase in the complexity of source code management due to the increased opportunities of development by multiple developers

Project Record Control

The version control function enables you to leave a change record of a project at any timing.
You can return to the desired project by tracing back the change record and comparing with past projects.

Project Development by Multiple Developers

The version control function provides the capability to check the difference and merge the changes when you apply them to the master project.
When you use the version control function to control projects, you can use this capability to develop programs by multiple developers.

Derived Development of Machines

The version control function allows you to use the record control function to derive, from a project, projects with partially different configurations and programs.
This facilitates the management of developing derived machines.

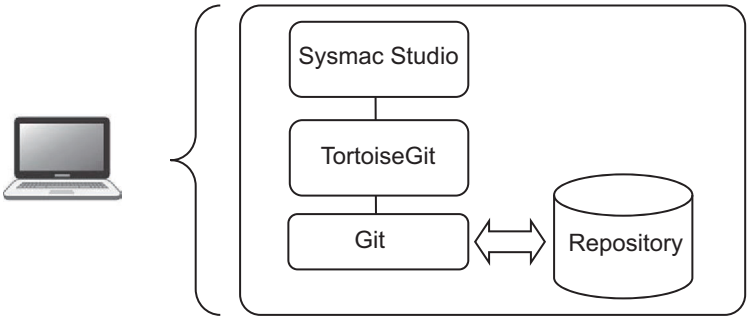
1-1-1 Overview of the System Configurations

Basic Configuration

The Sysmac Studio version control function operates in an environment that consists of the Sysmac Studio, "Git™" (version control system), "TortoiseGit"*1 (client software for Git), and "repositories" (folders managed by Git).

*1. “TortoiseGit” is open source and downloaded from (https://tortoisegit.org/download/).

The following figure illustrates the minimum configuration in which a single user has access to Sysmac Studio projects.



Basic configuration of the Sysmac Studio version control function environment (in a single computer)

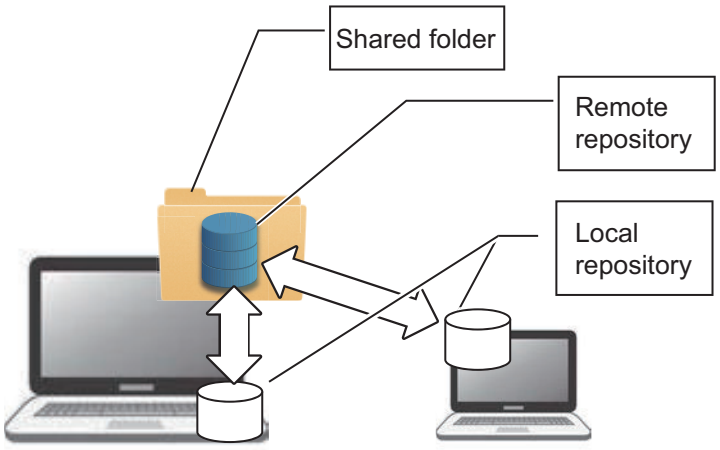
Configurations to Share the Repository with Multiple Users

Sysmac Studio version control function works with Git which has a feature of distributed version control system and they offer a mechanism to share the repository with multiple users.
The configuration consists of local repositories registered in the computers of each user and the remote repository shared by multiple users. At a certain timing of each user, the local repository and remote repository can be synchronized.
To share changes in the local repository with other users, perform a push operation to the remote repository. To apply changes made by other users to the local repository, perform a pull operation from the remote repository.
There are the following three practical configurations depending on the difference in how the remote repository is shared.

1. Using a shared folder on the computer to share it as the remote repository
2. Building a dedicated Git server to share it as the remote repository
3. Utilizing a Git server service on the Internet to share the remote repository

● Using a Shared Folder on the Computer to Share It as the Remote Repository

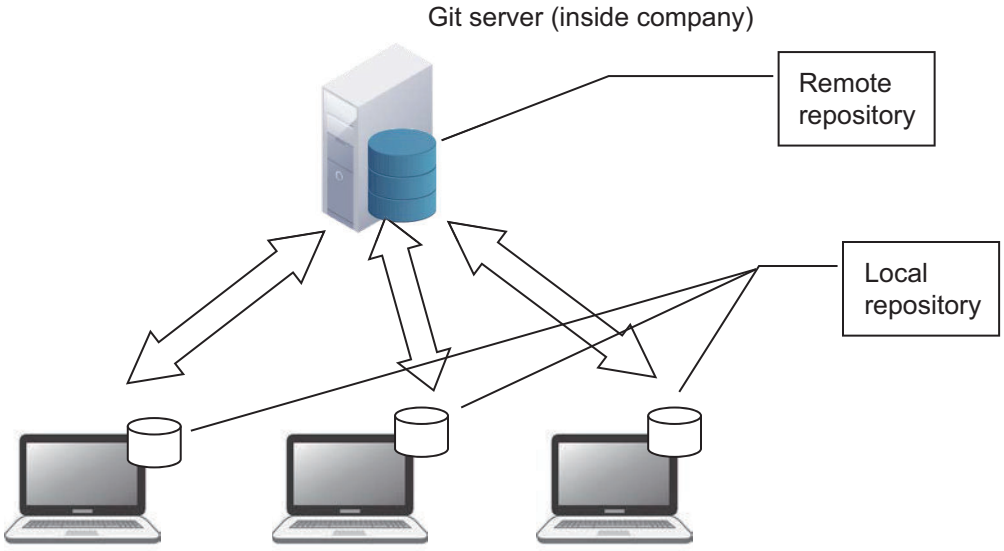
This is the easiest way to build a remote repository.
In this method, you use a Windows shared folder to publish a remote repository that is synchronized with the local repository to the local network so that it can be accessed from other computers.
In the following description, we use a remote repository that is built in this way.



Using a shared folder on the computer to share it as the remote repository

● Building a Dedicated Git Server to Share It as the Remote Repository

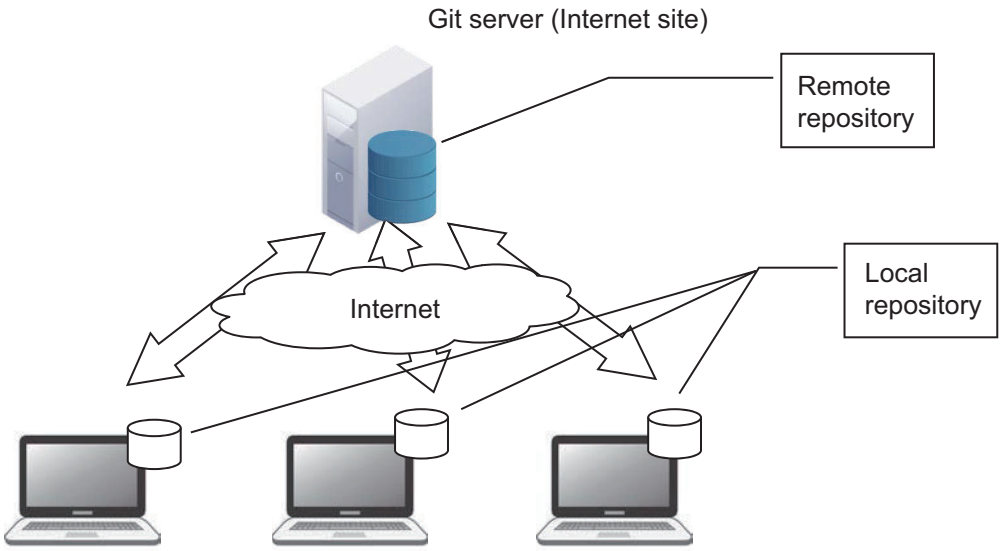
You can build a dedicated Git server to share the remote repository.
As Git server software, Gitbucket (for Windows) or GitLab, Gitbucket, Gitblit, Gogs, and so on (for Linux) are available. Although this configuration incurs costs for building and maintaining the server, it has an advantage in reducing the risk of data leakage because the system is closed within the company.



Sysmac Studio version control system on each user's computer

● Utilizing a Git Server Service on the Internet to Share the Remote Repository

On the Internet, there are Git server services such as GitHub. Although these are commercial paid services that incur a cost, there are advantages that they require no server maintenance and allow development in parallel with external developers.



Sysmac Studio version control system on each user's computer

1-2 Specifications

Product Model Numbers

To use the Sysmac Studio version control function, the following products are needed.

● Licenses

Product name	Number of licenses	Model number
Sysmac Studio Standard Edition Ver.1.□□	1 license	SYSMAC-SE201L
	3 licenses	SYSMAC-SE203L
	10 licenses	SYSMAC-SE210L
	30 licenses	SYSMAC-SE230L
	50 licenses	SYSMAC-SE250L

● Sysmac Studio Options

Product name	Number of licenses	Model number
Sysmac Studio Team Development Option	1 license	SYSMAC-TA401L
	3 licenses	SYSMAC-TA403L
	10 licenses	SYSMAC-TA410L
	30 licenses	SYSMAC-TA430L
	50 licenses	SYSMAC-TA450L

Supported Languages

The supported languages conform to the specifications of the Sysmac Studio. Refer to *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for details.

Applicable Models

You can use the version control function on all models that are supported by the Sysmac Studio. Refer to Sysmac Studio Version 1 Operation Manual (Cat. No. W504) for details.
However, for Controllers, the function is applicable to unit version 1.16 or later.

Applicable Computers

The applicable computers conform to the specifications of the Sysmac Studio. Refer to *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for details.

Necessary Software

To use the Sysmac Studio version control function, the following software is needed besides the Sysmac Studio.

Download the latest edition of software from their official Web sites.

- "Git" (32-bit or 64-bit edition)
  https://git-scm.com/downloads
- "TortoiseGit" (32-bit or 64-bit edition)
  https://tortoisegit.org/download/

1-3 Scope of the Version Control Function

The version control function is applicable to devices that are registered in the project, as well as the following data of each device.

- Data in *Configurations and Setup* and lower-level folders in the Multiview Explorer
- Data in *Programming* and lower-level folders (Applicable to devices that have data in *Programming* and lower-level folders only) or in *HMI* and lower-level folders in the Multiview Explorer

Depending on the device, however, there is other version-controlled data in addition to the above, or some of the above data is not version-controlled.
For devices with such data, refer to the following description.

Note that display settings for windows, such as the layout of each pane in the main window, are not version-controlled.

Additional Information

In addition to project data, you can include the following data in the scope of version control.

- Document files created by a user to provide a project description
- Library files for Controllers

Store these files in the *Document* and *Lib* folders, which are version-controlled. Refer to the description of *4-1-1 Version Control Project Folder List* on page 4-2 for details.

Controllers

In addition to the data configured under *Configurations and Setup* or *Programming* in the Multiview Explorer, the version control function is applicable to the following data.

Data type	Version control
Referenced libraries	○
Tag Data Link Setting	○

Among the data configured under *Configurations and Setup* or *Programming* in the Multiview Explorer, the version control function is not applicable to the following data.

Category	Item
Window display settings	Color code settings for Multiview Explorer
	Display settings for Cam Editor toolbar
	Bookmark settings
Data to monitor	Registered data in Watch tab page
	Registered data for differential monitoring
Data trace	Trace results
Communications settings	Communications settings
Simulation settings	Integrated NS-series PT simulation settings
	Variable snapshot settings
	Execution Time Estimation Mode settings
	Breakpoints
Build data	All build data

HMIs

Among the data configured under *Configurations and Setup* or *HMI* in the Multiview Explorer, the version control function is not applicable to the following data.

Category	Item
Window display settings	Color code settings for Multiview Explorer
	Window display states
	Bookmark settings
Data to monitor	Registered data in Watch tab page
Simulation settings	Breakpoints

2

Software Setup and Basic Settings

This section describes the procedures to set up software that is necessary to use the Sysmac Studio version control function and configure the basic software settings.

2-1 Installing the Sysmac Studio

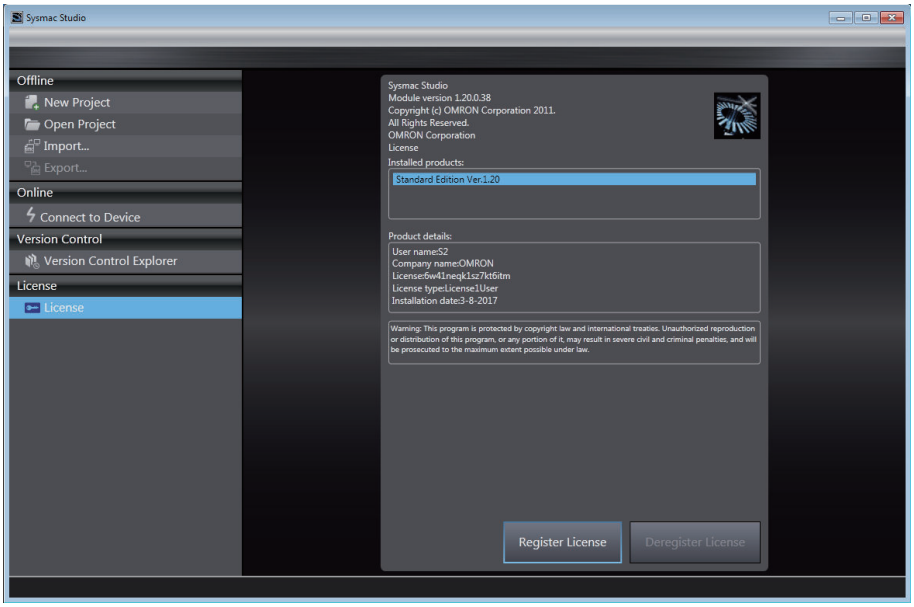
Install the Sysmac Studio from the DVD. For details of the installation procedure, refer to *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)*.

2-2 Registering a Sysmac Studio Option License

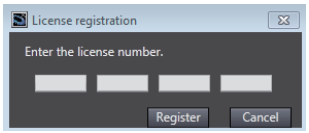
To use the Sysmac Studio version control function, you must register a *Sysmac Studio Team Development Option* license in the Sysmac Studio Standard Edition.
Use the following procedure to register an option license.

1 Select **All Programs** - **OMRON** - **Sysmac Studio** - **Sysmac Studio** from the Windows Start menu.
The Sysmac Studio starts and the start page is displayed.

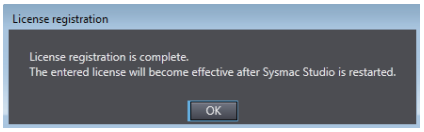
2 Click **License** on the Start page.
The licenses that are currently registered are displayed.



3 Click the **Register License** button.
A **License registration** dialog box is displayed.



4 Enter the license number for the Sysmac Studio Team Development Option, and then click the **Register** button.
If the license is registered normally, a message appears asking you to restart the software.



Restart the Sysmac Studio to complete registration.

2-3 Installing Git

Download the latest installer from the Git download site and install it as a user with administrator rights.

Git (32-bit or 64-bit edition)

https://git-scm.com/downloads

Depending on the operating system installed on the computer, download the 32-bit or 64-bit edition of the installer.

Follow the instructions in the Git installer wizard to complete the installation. Although wizard displays several pages during the installation, the description below covers only the pages on which you must select a specific item. You can leave other pages as default.

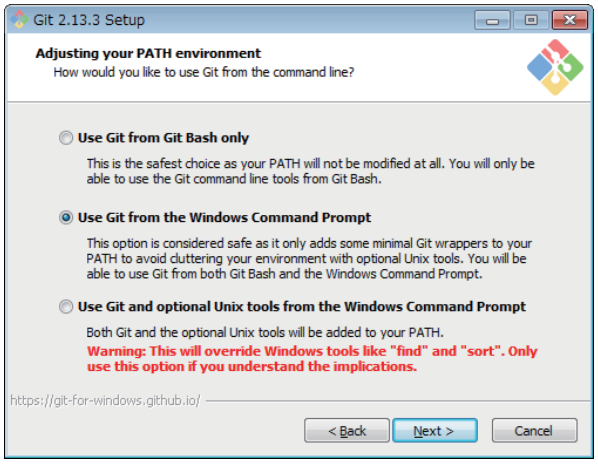
Precautions for Correct Use

To install Git, be sure to log onto Windows as the administrator or as a user with administrator rights.

From here, we explain the procedure using the window of Git version 2.13.3 as an example.

● Adjusting your PATH environment Page

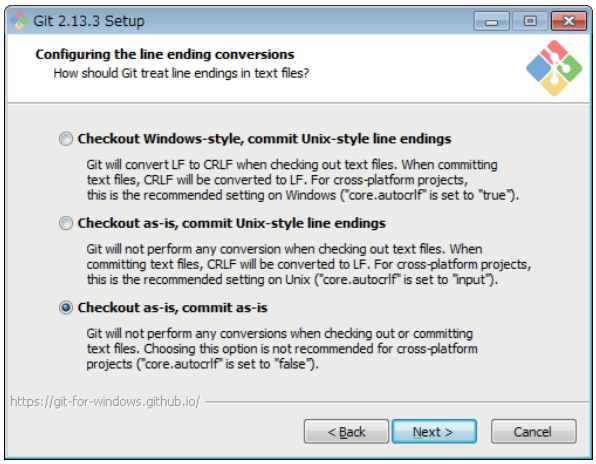
On the **Adjusting your PATH environment** page, be sure to select **User Git from the Windows Command Prompt** (the default).



● Configuring the line ending conversions Page

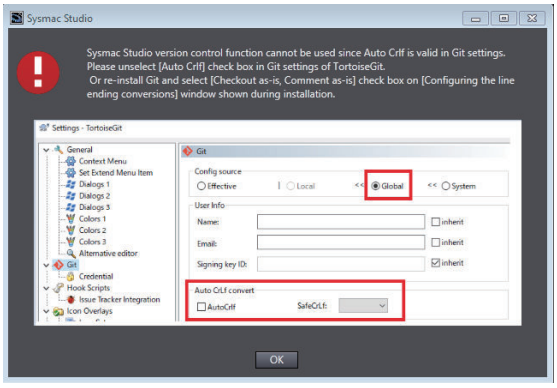
On the **Configuring the line ending conversions** page, be sure to select **Checkout as-is, commit as-is**.

When you do not select **Checkout as-is, commit as-is**, you cannot use the Sysmac Studio version control function.

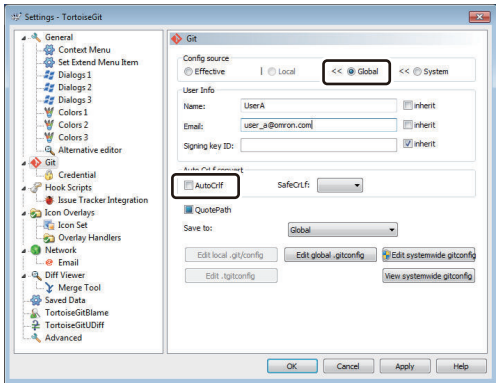


Additional Information

If you select an option other than **Checkout as-is, commit as-is** and try to use the Sysmac Studio version control function, an error message is displayed.



To clear the selection of **AutoCrlf** in the Git settings in **TortoiseGit**, select **Global** in **Config source** and then clear the selection.



2-4 Installing "TortoiseGit"

Download the latest installer from the "TortoiseGit" download site and install it as a user with administrator rights.

"TortoiseGit" (32-bit or 64-bit edition)

https://tortoisegit.org/download/

Depending on the operating system installed on the computer, download the 32-bit or 64-bit edition of the installer.

Follow the instructions in the "TortoiseGit" installer wizard to complete the installation. You can leave all the items that you encounter during the installation process as default.

In addition, download and install a language pack as necessary.

Precautions for Correct Use

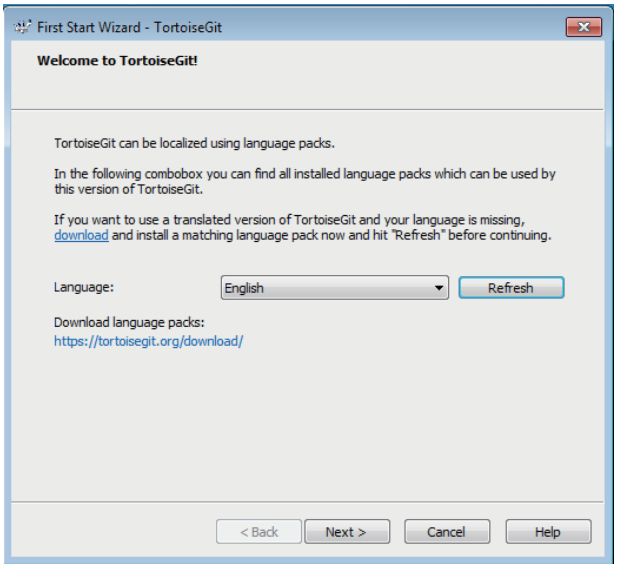
To install "TortoiseGit", be sure to log onto Windows as the administrator or as a user with administrator rights.

From here, we explain the procedure using the window of "TortoiseGit 2.4.0.2" as an example.

2-4-1 Initial Setting for "TortoiseGit"

This section describes the initial settings for using "TortoiseGit".

1 In the Installation **Completed** dialog box, select the **Run first start wizard** check box and click the **Finish** button. Or, select **All Programs** - **TortoiseGit** - **Settings** from the Windows Start menu and, in the **Settings** dialog box, select **Re-run First Start Wizard** in the **General** pane. The initial settings dialog box is displayed.

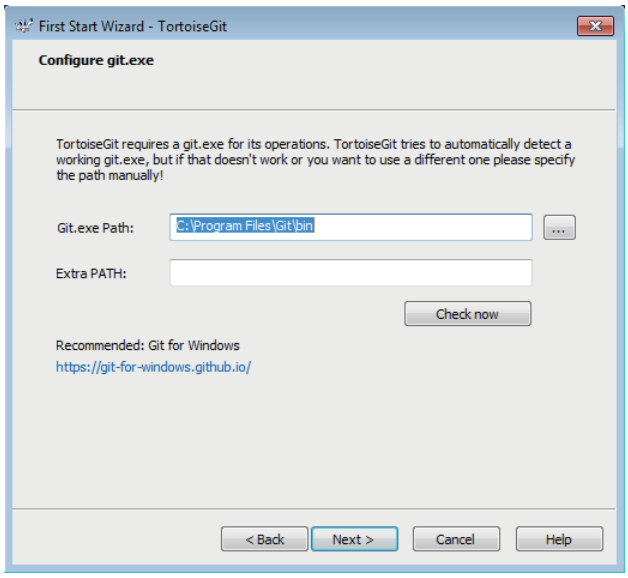


2 Click the **Next** button.

An overview of "TortoiseGit" is displayed.

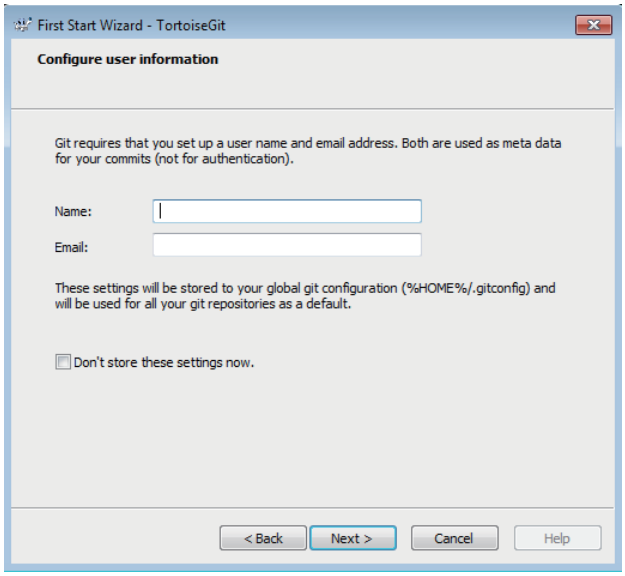
3 Click the **Next** button.

A dialog box is displayed to configure the path to Git.exe.



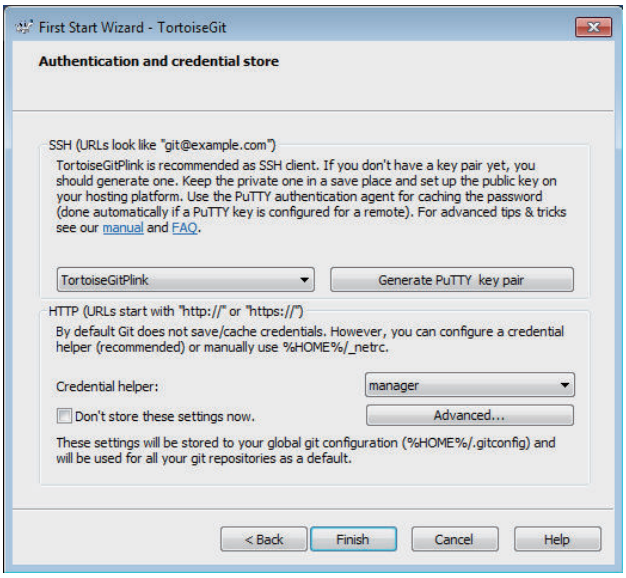
Set the path to Git execution module “Git.exe”. If you did not change the install path when installing Git, leave it as default.

4 Click the **Next** button.
The Configure user information dialog box is displayed.



The user name and email address that you enter here will be used as change record information.

5 Enter the user name and email address, and then click the **Next** button.
The Authentication and credential store dialog box is displayed.



To use the Sysmac Studio version control function, do not change the option selected by default.

6 Click the **Finish** button.
This completes the initial setting for "TortoiseGit".

2-5 Creating a Shared Folder and a Remote Repository

Use the following setting procedures to share the remote repository using a shared folder. For the procedures to build a dedicated Git server and use a Git server service on the Internet, refer to the service provider's information.
Create a shared folder in which to store a remote repository, and then create a folder that serves as the remote repository in the shared folder.

2-5-1 Creating a Shared Folder

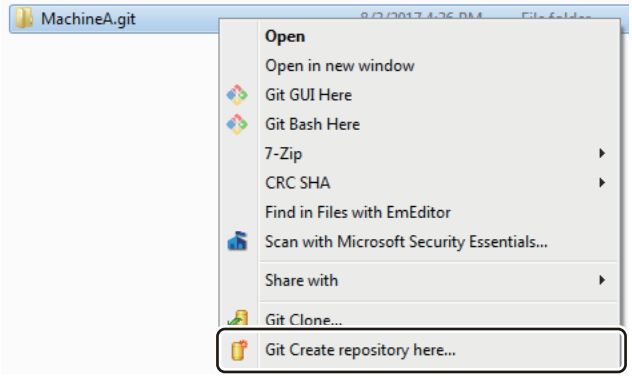
1 Creating a folder
In Windows Explorer, create a new folder.
You can create this folder in any location with any name. Let's assume that it is C:\Git.

2 Setting the folder to a shared folder
Right-click the folder that you created and select **Properties** from the pop-up menu. Then, in the dialog box that is displayed, click the **Share** tab to perform the sharing settings.
Here, you configure the folder to allow full access from other users' computers on which the Sysmac Studio version control function used.

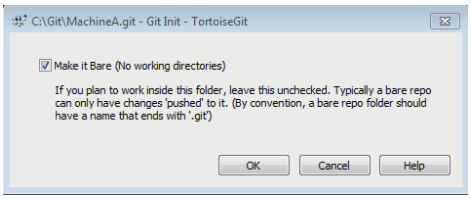
2-5-2 Creating a Remote Repository

1 Under the shared folder, create a new folder.
According to the Git conventions, it is required that you name the folder to use as a remote repository to the repository name followed by ".git". For example, if the repository name is "MachineA," then name the folder "MachineA.git".

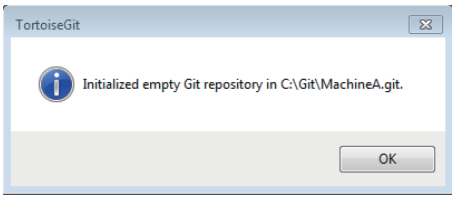
2 Right-click the created folder and select **Git Create repository here** from the pop-up menu.



A repository creation option dialog box is displayed.



3 Select the check box in the dialog box and click the **OK** button.
The repository is created and the following dialog box is displayed.



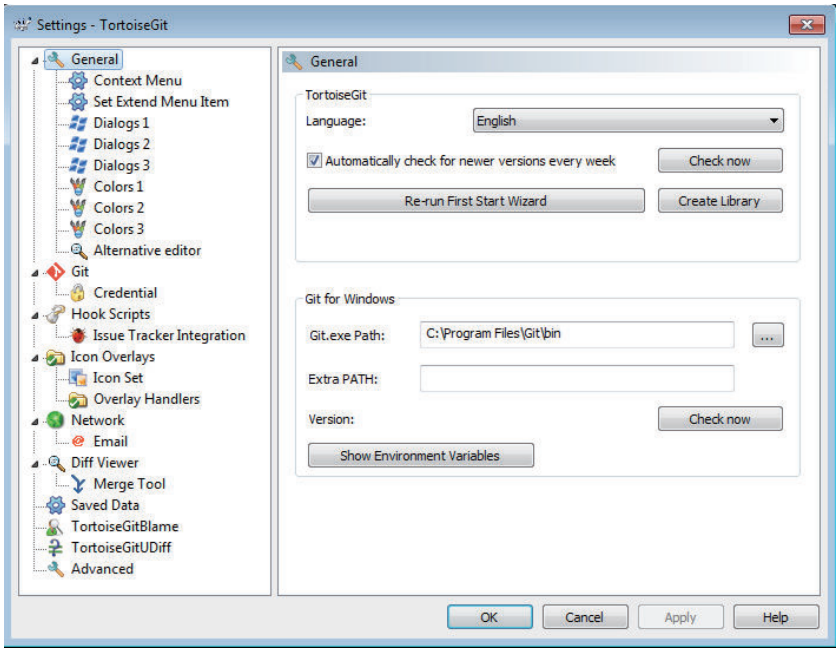
4 Click the **OK** button.
This completes the creation of a remote repository.

Note that you can control only one Sysmac Studio project per repository. Create a directory for each project to use version control function.

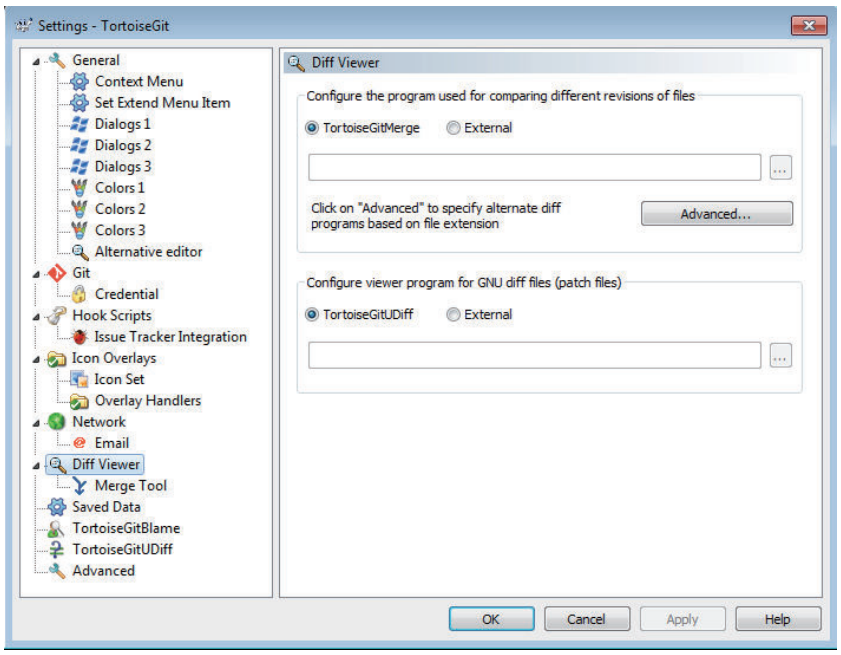
2-6 Additional Setting for "TortoiseGit"

In "TortoiseGit", add settings to enable graphical comparison of Sysmac Studio projects from "TortoiseGit".
By adding these settings, you can check the difference between projects in the Sysmac Diff dialog box. Refer to *4-3 Sysmac Diff Dialog Box* on page 4-8 for information on the Sysmac Diff dialog box.

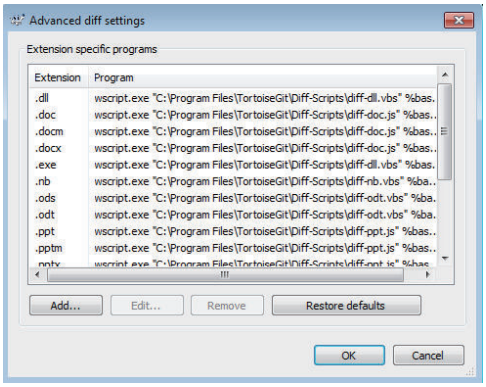
1 Select **All Programs** - **TortoiseGit** - **Settings** from the Windows Start menu.
The **Settings** dialog box is displayed.



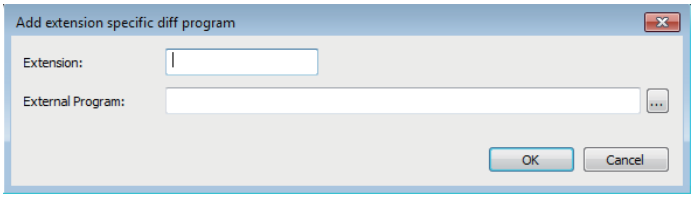
2 Select **Diff Viewer** from the tree.
The **Diff Viewer** pane is displayed.



3 In the **Diff Viewer** pane, click the **Advanced** button.
The **Advanced diff settings** dialog box is displayed.



4 Click the **Add** button.
The **Add extension specific diff program** dialog box is displayed.



5 Enter the following text string, and then click the **OK** button.

Item	Text string to enter
Extension	.oem
External program	(Sysmac Studio installation folder*1)\SysmacDiff.exe -gitdiff %base %mine %bpath %brev %yrev

*1. The Sysmac Studio installation folder is by default as follows.

- **Sysmac Studio (32 bit)**
  For 32-bit OS: C:\Program Files\OMRON\Sysmac Studio
  For 64-bit OS: C:\Program Files (x86)\OMRON\Sysmac Studio
- **Sysmac Studio (64 bit)**
  For 64-bit OS: C:\Program Files\OMRON\Sysmac Studio

6 Click the **OK** button in the **Advanced diff settings** dialog box.
This completes the additional setting for "TortoiseGit".

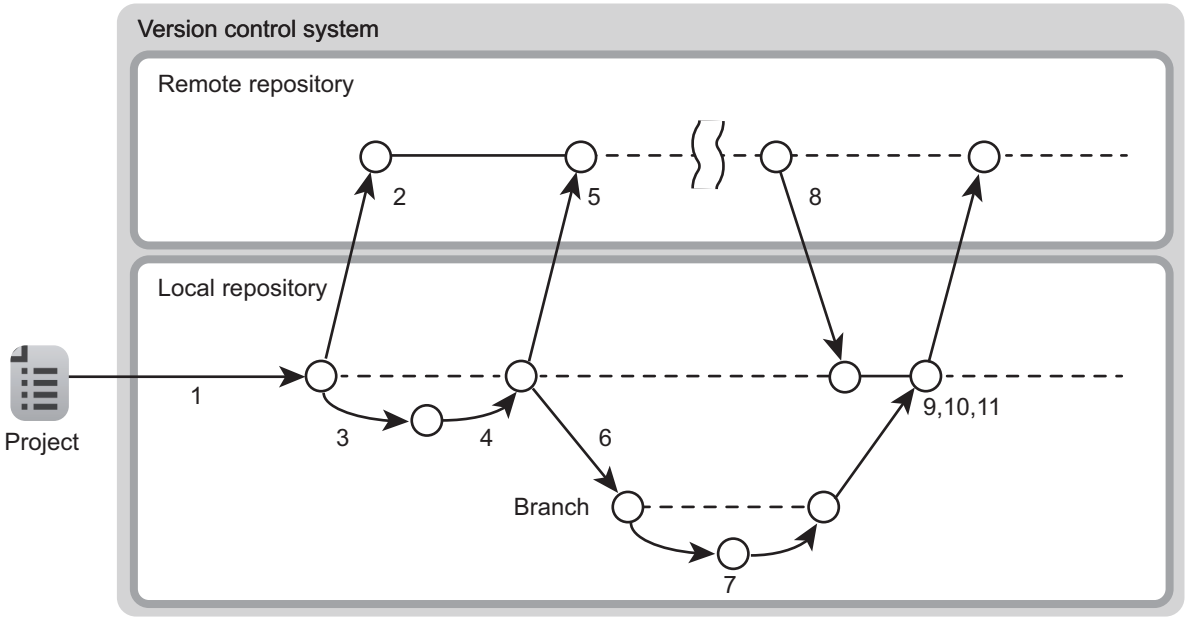
3

Basic Operations of Version Control

This section describes the basic operations of the Sysmac Studio version control function, as well as the precautions for use of the function.

3-1 Basic Development Flow Using the Version Control Function

The following describes the basic development flow while you control the versions of a Sysmac Studio project.
From here, we explain the operation procedure for each task according to this work flow.



1 Importing a project

Add a project to the version control system.

Work	Description	Reference
Importing a project	Import a project that is not yet version controlled into the version control system.	*3-2-1 Importing a Project* on page 3-8
Committing the project	Commit the imported project and register it in the local repository.	*3-2-2 Committing the Changes* on page 3-10

2 Pushing the project data to the remote repository

Save the data to the remote repository, which is the place to manage the deliverables.

Work	Description	Reference
Creating a remote repository	Create a remote repository in which to manage the imported project.	*2-5-2 Creating a Remote Repository* on page 2-10
Setting the path to the remote repository	For the imported project, set the pass to the remote repository.	*3-2-3 Setting the Path to the Remote Repository* on page 3-12
Pushing the project data to the remote repository	Register the imported project in the remote repository.	*3-2-4 Pushing the Project Data to the Remote Repository* on page 3-13

3 Editing the project in the local repository

Edit the project.

Work	Description	Reference
Editing the project	Edit the imported project.	*3-2-5 Editing the Project* on page 3-14
Checking the changes to the project while editing	Check the changes made to the project while editing.	*3-2-6 Checking the Changes to the Project While Editing* on page 3-15

4 Committing the changes

Commit the changes.

Work	Description	Reference
Committing the changes	Apply the changes made to the project to the local repository.	*3-2-2 Committing the Changes* on page 3-10

5 Pushing the changes to the remote repository

Apply the results of edits made to the project to the data in the remote repository.

Work	Description	Reference
Pushing the changes to the remote repository	Apply the changes made to the project in the local directory to the remote repository.	*3-2-4 Pushing the Project Data to the Remote Repository* on page 3-13

6 Creating and switching the branch

Create working branch data.

Work	Description	Reference
Creating a branch	Create a branch in the local repository.	*3-2-7 Creating a Branch* on page 3-17
Switching the working branch	Switch the work place to the created branch.	*3-2-8 Switching to the Branch* on page 3-18

7 Making changes in the branch

Edit the project in the working branch.

Work	Description	Reference
Editing the project	Edit the project in the branch.	*3-2-5 Editing the Project* on page 3-14
Committing the changes	Apply the changes made to the project to the branch in the local repository.	*3-2-2 Committing the Changes* on page 3-10

8 Pulling the project data from the remote repository to the local repository

Acquire the latest deliverable data from the remote repository.

Work	Description	Reference
Pulling the project data from the remote repository to the local repository	Acquire the latest project changed by another operator into the local repository.	*3-2-9 Pulling the Project Data from the Remote Repository to the Local Repository* on page 3-20

9 Merging the changes

Apply the current changes made to the project in the branch to the latest deliverable data.

Work	Description	Reference
Merging the changes to the branch into the local repository	Merge the changes made to the branch into the local repository.	*3-2-10 Merging the Changes* on page 3-21

10 Resolving a conflict

If different changes have been made to the same portion when you merge the project data, determine which changes to apply.

Work	Description	Reference
Resolving a conflict	If the same portion has been changed by another operator, a conflict occurs when you merge the changes made in the branch into the local repository. Determine which changes to apply before you merge the project data.	*3-2-11 Resolving a Conflict* on page 3-23

11 Pushing the data to the remote repository after merging

Save the merged project data in the remote repository.

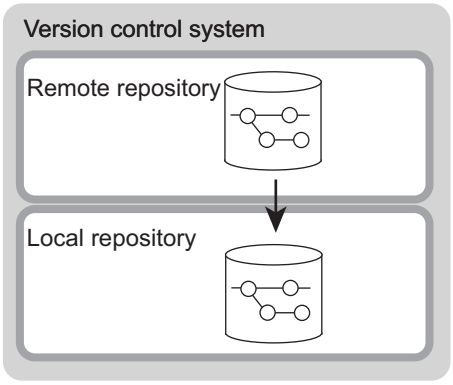
Work	Description	Reference
Committing the merged data	Apply the merged data to the local repository.	*3-2-2 Committing the Changes* on page 3-10
Adding a tag	Add tag information to the data to resister in the remote repository.	*3-2-12 Adding a Tag* on page 3-24
Pushing the data to the remote repository after merging	After completion of merging, register the completed data in the remote repository.	*3-2-4 Pushing the Project Data to the Remote Repository* on page 3-13
Checking the version control log	Display the version control log and check that changes made by each operator are applied correctly.	*3-2-13 Displaying the Version Control Log* on page 3-25

Other Operations

The operations of the Sysmac Studio version control function that you perform as necessary are described below.

● Cloning Project Data

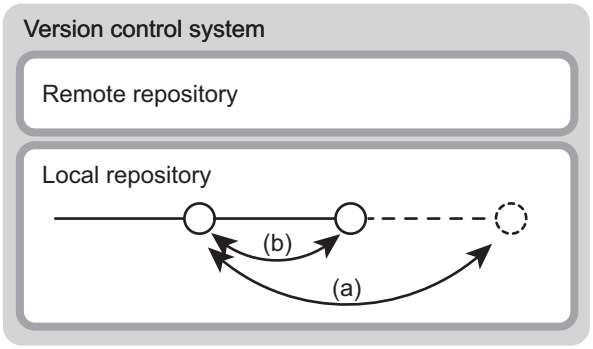
When you create a version-controlled project in the local repository, clone the existing data from a remote repository.



Work	Description	Reference
Cloning project data	Create a folder in which to store cloned data on the version control system, and then copy the data from a remote repository.	*3-2-14 Cloning Project Data* on page 3-26

● Comparing Project Data

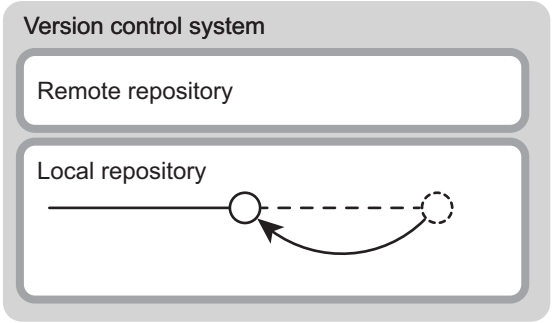
Compare the contents of the current project or a specific revision of the project with the contents of another revision of the project.



	Work	Description	Reference
(a)	Comparing the current project with a specific revision of the project	Compare the contents of the current project with the contents of a specific revision of the project.	*Comparing the Current Project with a Specific Revision of the Project* on page 3-28
(b)	Comparing two revisions	Compare any two revisions from those listed in the change record of the project.	*Comparing Two Revisions* on page 3-30

● Canceling Changes

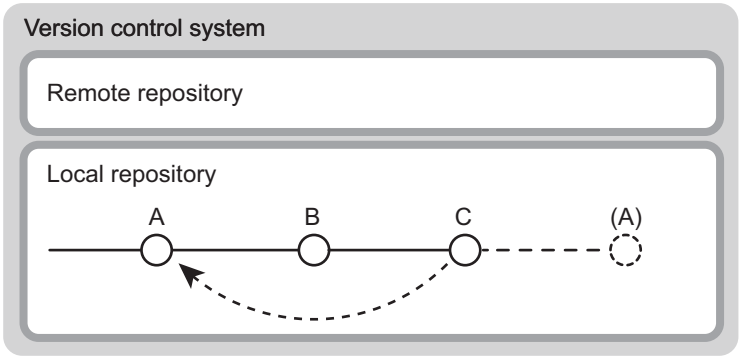
Revert the contents of the project to the point when they were last committed.



Work	Description	Reference
Canceling changes	Revert the contents of the project to the point when they were last committed by canceling the changes made after that point.	*3-2-16 Canceling Changes* on page 3-31

● Restoring a Project

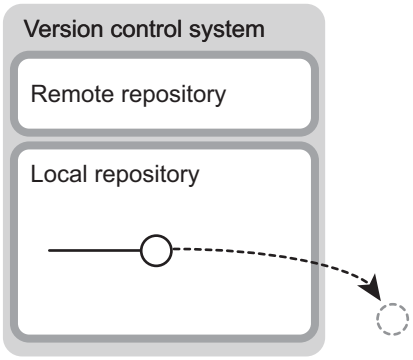
Revert the contents of the project to an arbitrary older revision.



Work	Description	Reference
Restoring a Project	Revert the contents of the project to an arbitrary older revision by canceling the changes made after the revision.	*3-2-17 Restoring a Project* on page 3-32

● Deleting a Project

Delete a version-controlled project from the local repository.



Work	Description	Reference
Deleting a project	Delete a version-controlled project from the local repository.	*3-2-18 Deleting a Project* on page 3-34

3-2 Operation Procedures in the Basic Development Flow

The following describes the specific operation procedures in the development flow when you control the versions of a Sysmac Studio project.

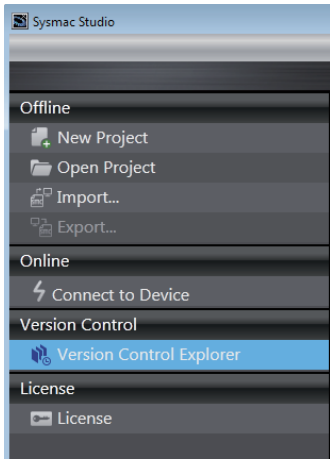
⚠ WARNING

When you manage a project using the version control system, do not use the version control system's functions directly from Windows Explorer on files that compose the project. Doing so may cause the loss of consistency among files that compose the project, and the control system may perform unexpected operation.
If you use the version control system to develop programs with multiple developers, check them for proper execution before you use them for actual operation.

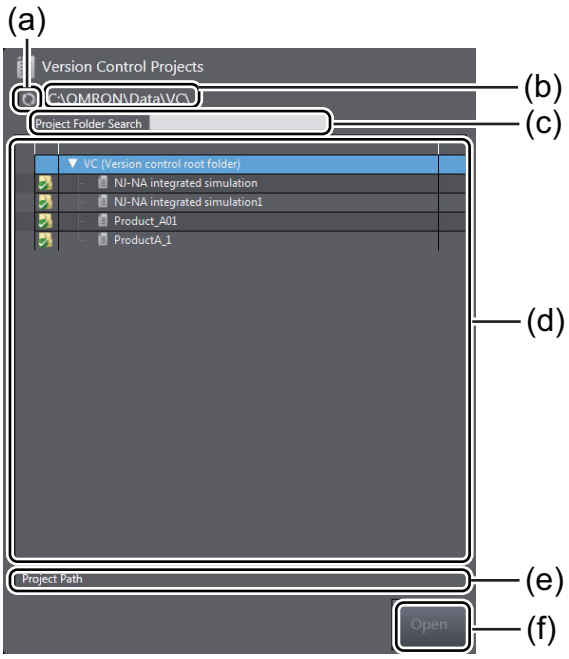
3-2-1 Importing a Project

Use the following procedures to import a project that is not under control of the Sysmac Studio version control function into the version control system (local repository).

1 On the start page of Sysmac Studio, select **Version Control** - **Version Control Explorer**.



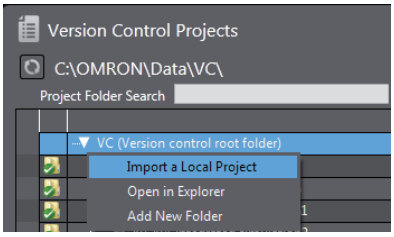
The **Version Control Projects** window is displayed.



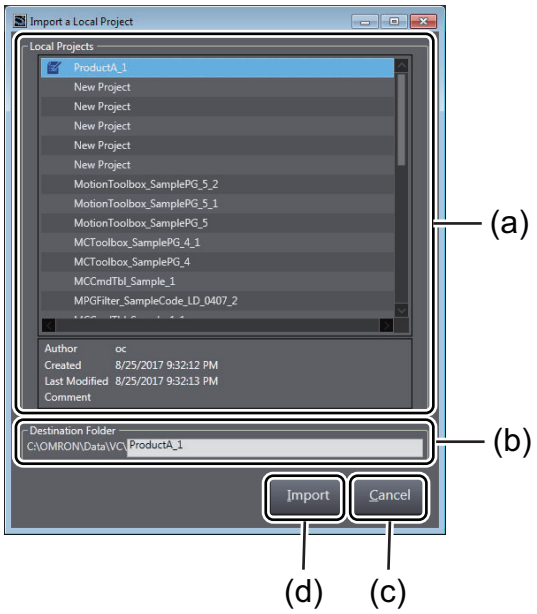
	Item	Description
(a)	**Update** button	Use this button to update the version-controlled project list.
(b)	Version control root folder path	The path to the *VC (Version control root folder)* is displayed. You cannot edit this path because it is fixed.
(c)	Project folder search box	Enter a text string here to narrow the project folders listed in the version-controlled project list to those that contain the entered text in the folder name.
(d)	Version-controlled project folder list	Folders that contain version-controlled projects are listed. Refer to *4-1 Version Control Projects Window* on page 4-2 for details on the icons and menu commands.
(e)	Project location	The folder path of the selected project is displayed.
(f)	**Open** button	Use this button to open the selected project.

Note that all version-controlled projects are controlled in folders under the *VC (Version control root folder)*.

2 Right-click **VC (Version control root folder)** and select **Import a Local Project**.



The **Import a Local Project** dialog box is displayed.



	Item	Description
(a)	Project list	Among the projects that currently exist on the computer, those that are not version controlled are listed. Select the project to import.
(b)	Import destination folder	Set the name of the import destination version control target folder. The folder name must consist of up to 80 single-byte characters, including the folder path. By default, this is set to the name of the project that you selected in the **Project list**.
(c)	**Cancel** button	Use this button to cancel the import and close the dialog box.
(d)	**Import** button	Use this button to import the selected project to the version control system.

3 Select the project to import and click the **Import** button.
A project folder is created under the **VC (Version control root folder)** and the project is imported in it.

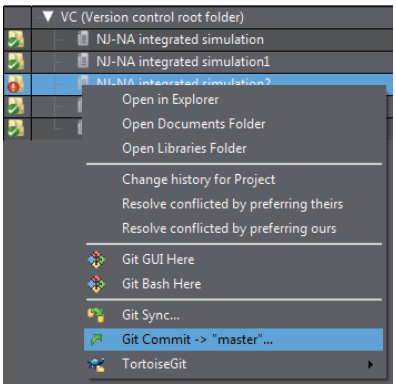
Precautions for Correct Use

- You cannot import password-protected projects to the version control system. Disable the password protection before you import the project.
- You cannot import the following projects to the version control system.
  - Projects that include Controllers with version 1.15 or earlier
  - Projects that include Controllers with protected data

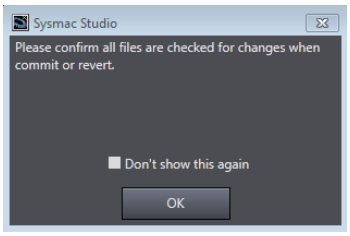
3-2-2 Committing the Changes

"Commit" means to apply changes to the local repository after you make changes to a project or newly import a project into the version control system. Use the following procedures to commit changes.

1 Right-click the target project in the **Version Control Projects** window and select **Git Commit -> "master"** from the pop-up menu. Or, right-click the folder icon in the Multiview Explorer and select **Git Commit -> "master"** from the pop-up menu.

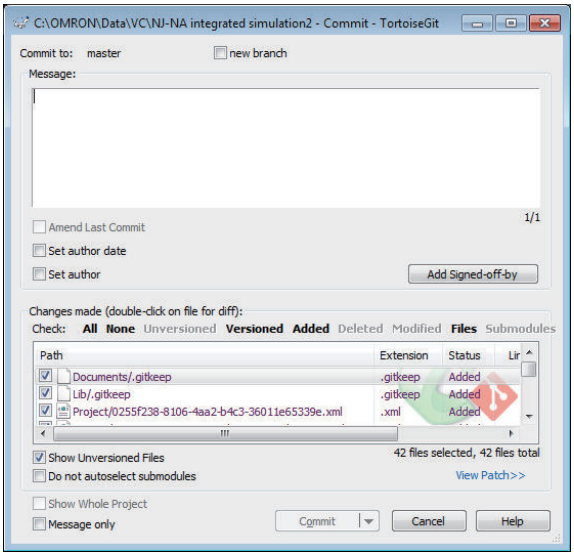


When the project is not yet saved, a confirmation dialog box is displayed to ask if you need to save the project. If you click the **OK** button, the following dialog box is displayed to alert you.



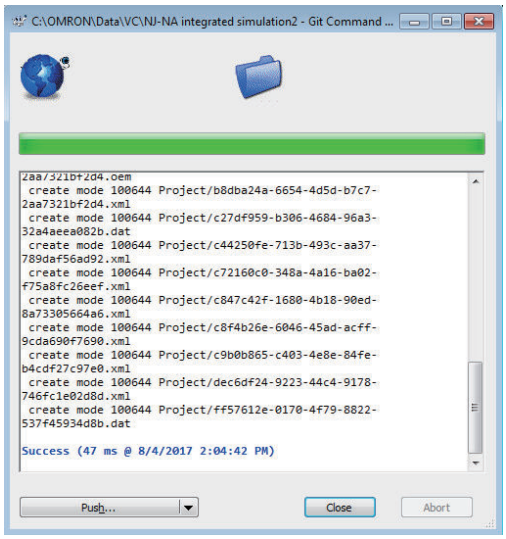
If you do not want to show this message from the next time, select the check box.

2 Check the message and then click the **OK** button.
The "TortoiseGit" **Commit** dialog box is displayed.



Confirm that the check boxes for all files listed under **Changes made** are selected.

3 Enter a message that you want to leave as a record in the message area and click the **Commit** button.
A dialog box is displayed to indicate the completion of the commit processing.



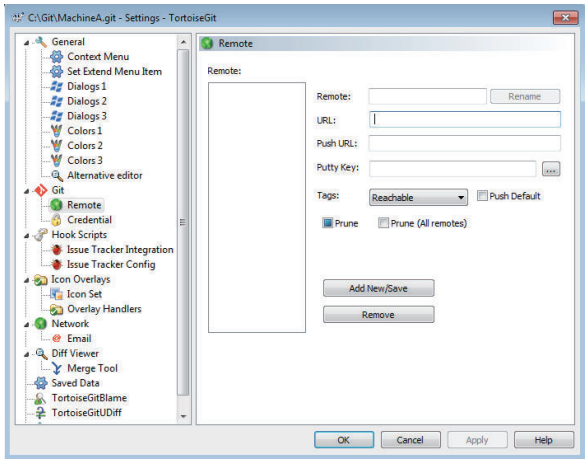
4 Click the **Close** button.
The dialog box is closed.
This completes the commit procedure.

3-2-3 Setting the Path to the Remote Repository

For the imported project, set the pass to the remote repository.
Here, we set *C:\Git\MachineA.git* as the path to the remote repository, as an example.
Refer to *2-5-2 Creating a Remote Repository* on page 2-10 for the procedure to create a remote repository.

1 Right-click the target project folder in the **Version Control Projects** window and select **TortoiseGit** - **Settings** from the pop-up menu.
The **Settings** dialog box is displayed.

2 In the **Settings** dialog box, select **Git** - **Remote**.
The **Remote** pane is displayed.



3 Configure the **Remote** and **URL** settings.

Item	Description	Entry example
Remote	The connection name to connect to the remote repository. Enter any connection name. Normally, enter *origin*.	origin
URL	Enter the path to the remote repository. This must be a folder path if the remote repository is in a shared folder or a URL if the remote repository is on the server.	C:\Git\MachineA.git

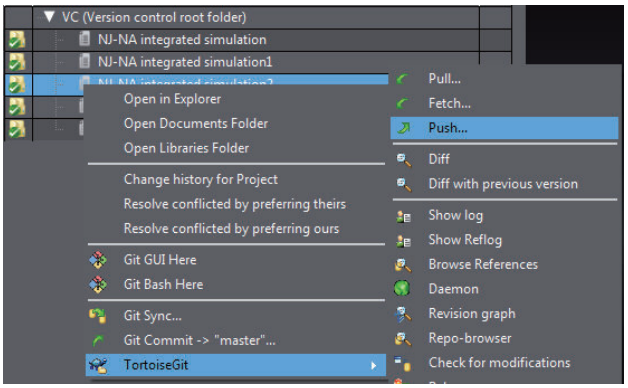
4 Click the **Add New/Save** button.
The remote connection that you created is added to the **Remote** list.

5 Click the **OK** button.
This completes the setting of the path to the remote repository.

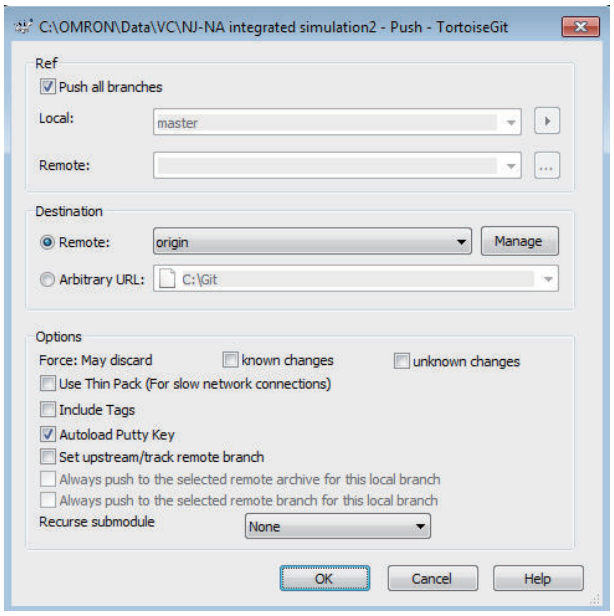
3-2-4 Pushing the Project Data to the Remote Repository

After you edit the project in the local repository on your computer, apply the changes to the master project in the remote repository. Or, after you newly import a project, register the target project as the master project in the remote repository.
Use the following procedure to push the project data from the local repository to the remote repository.

1 Right-click the target project in the **Version Control Projects** window and select **TortoiseGit** - **Push** from the pop-up menu. Or, right-click the folder icon in the Multiview Explorer and select **TortoiseGit** - **Push** from the pop-up menu.



The "TortoiseGit" **Push** dialog box is displayed.



Select the local repository in **Local** under **Ref** and the remote connection name in **Remote** under **Destination**. To register a newly imported project in the remote repository, select the **Push all branches** check box.
Normally it is not necessary to change the option item settings. If you must change the option item settings, refer to the "TortoiseGit" Help to become familiar with details on each setting.

Additional Information

If the path to the remote repository is not set, the remote connection name is not displayed. Click the **Manage** button and set this in the "TortoiseGit" **Settings** dialog box. Refer to *3-2-3 Setting the Path to the Remote Repository* on page 3-12 for details.

2 Click the **OK** button.
A dialog box is displayed to indicate that the push processing is finished

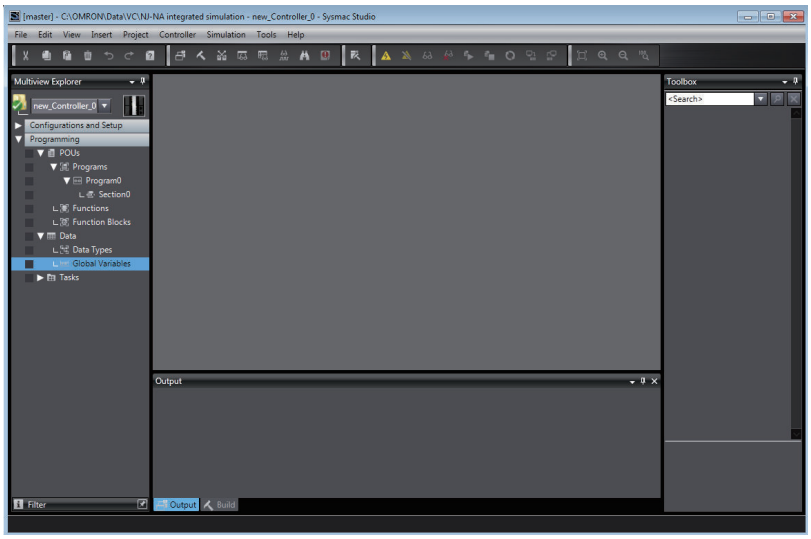
3 Click the **Close** button.
The dialog box is closed.
This completes the procedure to push the project data to the remote repository.

3-2-5 Editing the Project

Use the following procedure to edit the project in the local repository.

1 On the start page of Sysmac Studio, select **Version Control** - **Version Control Explorer**.
The **Version Control Projects** window is displayed.

2 Select the project to edit and click the **Open** button.
The project is opened.



3 Edit the project.

Note that there are device-specific precautions that you must follow when you edit version controlled projects. Refer to *3-3 Precautions on Use of Project Version Control* on page 3-35 for details.

For the procedures to edit the configurations, setups, or programs, refer to *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)*.

4 When you finish editing the project, select **Save** from the **File** menu.

The results of edits are saved.

After you complete all of the necessary edits, you normally proceed to committing changes to the local repository. Refer to *3-2-2 Committing the Changes* on page 3-10 for details on the commit procedure.

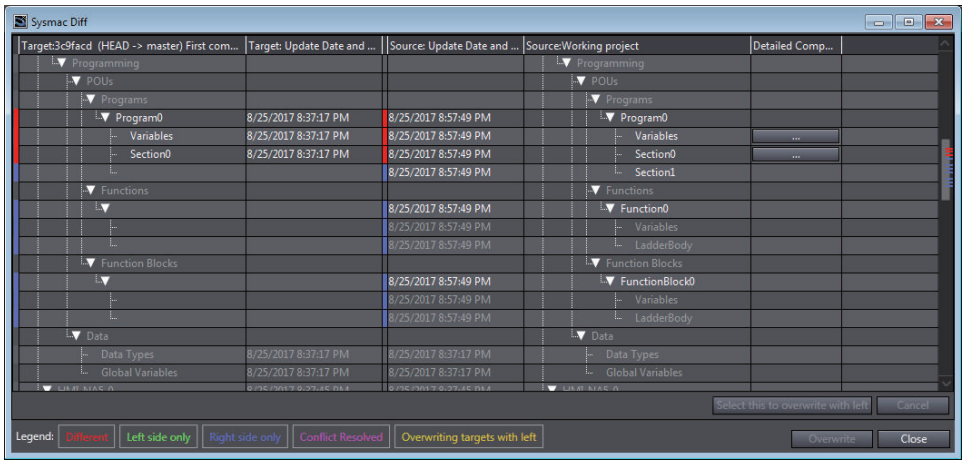
3-2-6 Checking the Changes to the Project While Editing

You can check while editing where in the project you have made changes, by comparing the current project with the source project.

Use the following procedures to check the changes.

1 Right-click the folder icon in the Multiview Explorer and select **Show Pending Changes for Project** from the pop-up menu.

When the project is not yet saved, a confirmation dialog box is displayed to ask if you need to save it. Click the **Yes** button to display the **Sysmac Diff** dialog box.



2 Check where you have made changes. Refer to *4-3 Sysmac Diff Dialog Box* on page 4-8 for details on the **Sysmac Diff** dialog box.

Checking Changes for Each Data Item

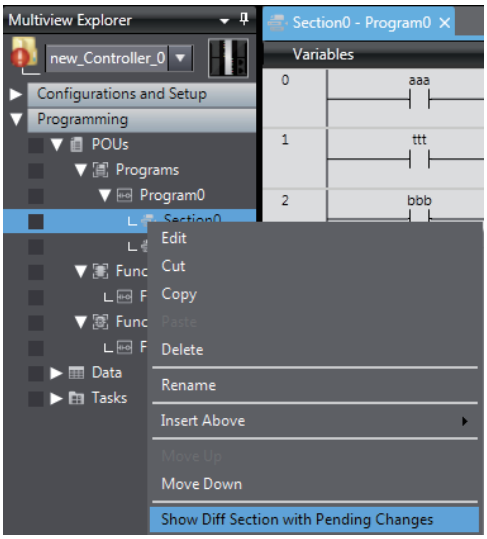
You can also check changes to the project in the following units of data.

- Controllers
  - Ladder programs
  - ST programs
  - Local variables
  - Data types
  - Global variables
  - NC programs
- HMIs
  - Pages
  - Page subroutines
  - Recipe fields
  - Recipes
  - Data types
  - Global variables
  - Global events
  - Global subroutines
  - Resources

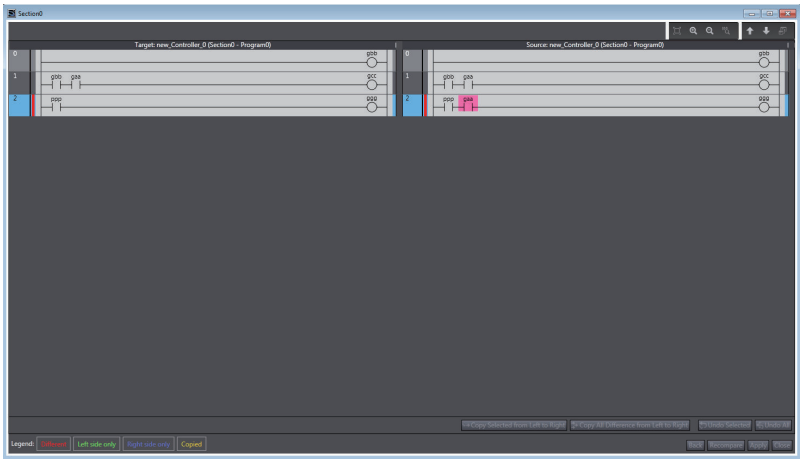
Right-click each item in the Multiview Explorer and execute the command from the pop-up menu. Refer to *4-2-2 Menu Commands for Checking Changes for Each Data Item* on page 4-7 for information on the pop-up menu for each data item.

The following operation procedure uses a ladder program for a section as an example.

1 In the Multiview Explorer, right-click the target section and select **Show Diff Section with Pending Changes** from the pop-up menu.



A detailed comparison dialog box is displayed.



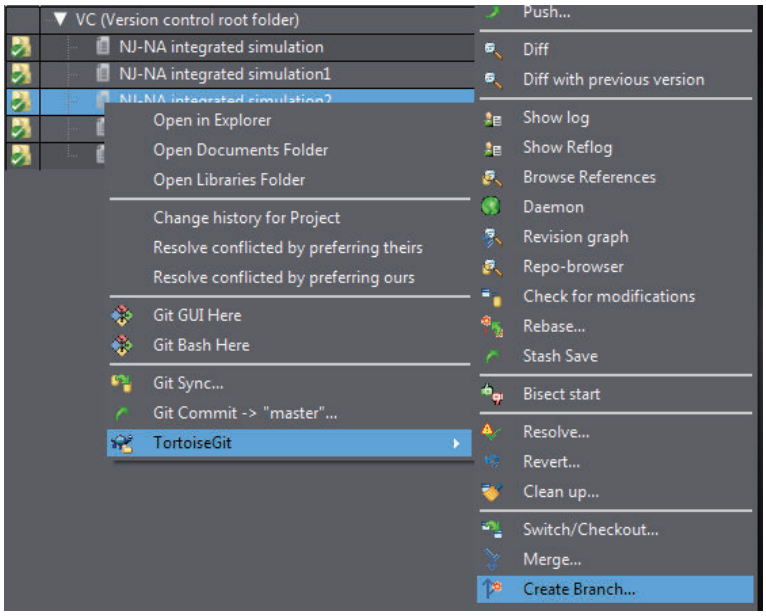
In the detailed comparison dialog box, you can check changes and undo them if necessary. Refer to *Detailed Comparison Window* on page 4-9 for details on the detailed comparison dialog box.

2 After checking changes, click the **Close** button.

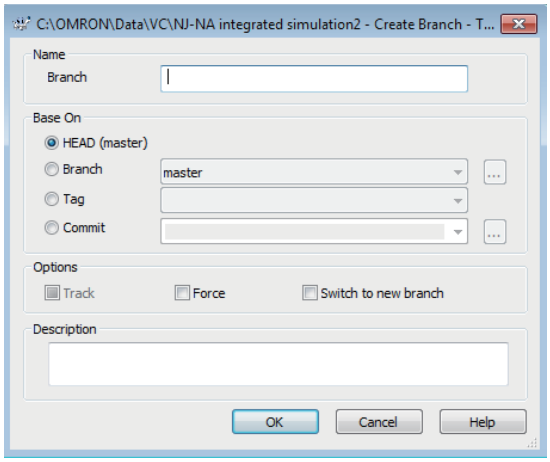
3-2-7 Creating a Branch

In project development by multiple developers or derived development of machines, you create data that is branched from the master project and edit the data. Here, you create a branch to edit branch data.
Use the following procedure to create a branch.

1 Right-click the target project in the **Version Control Projects** window and select **TortoiseGit** - **Create Branch** from the pop-up menu. Or, right-click the folder icon in the Multiview Explorer and select **TortoiseGit** - **Create Branch** from the pop-up menu.



The "TortoiseGit" **Create Branch** dialog box is displayed.



2 Enter the name of the branch, select an option in **Base On** of the branch, and click the **OK** button. To create a branch from the master, select the **HEAD (master)** check box.
The branch is successfully created.

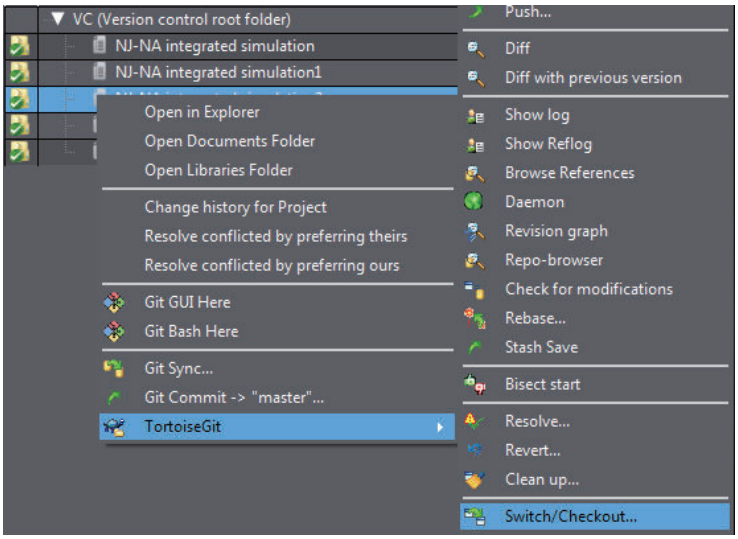
Precautions for Correct Use

If you use branches to develop a project with multiple developers, depending on the data, you may not be able to merge changes made by the developers.
Refer to *3-3 Precautions on Use of Project Version Control* on page 3-35 for details.

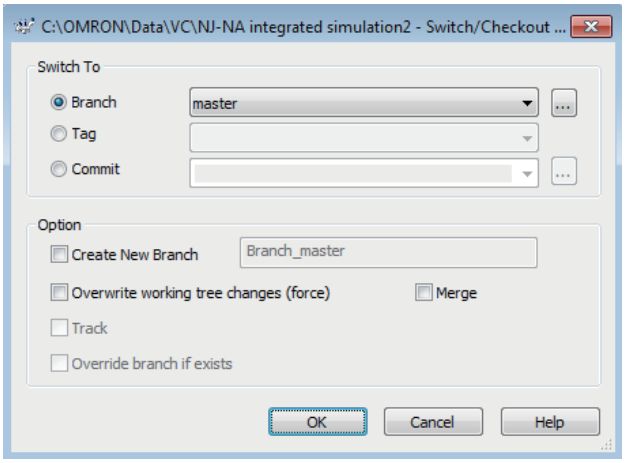
3-2-8 Switching to the Branch

Switch to the branch in which to edit the project.
Use the following procedure to switch to the working branch.

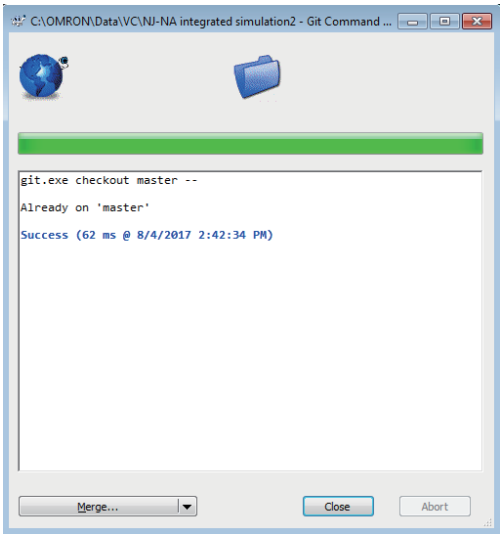
1 Right-click the target project in the **Version Control Projects** window and select **TortoiseGit - Switch/Checkout** from the pop-up menu. Or, right-click the folder icon in the Multiview Explorer and select **TortoiseGit - Switch/Checkout** from the pop-up menu.



The "TortoiseGit" **Switch/Checkout** dialog box is displayed.



2 Select the **Branch** option under **Switch To**, select the branch, and then click the **OK** button.
A dialog box is displayed to show the switching results. The switching to branch is successfully completed.



3 Click the **Close** button.

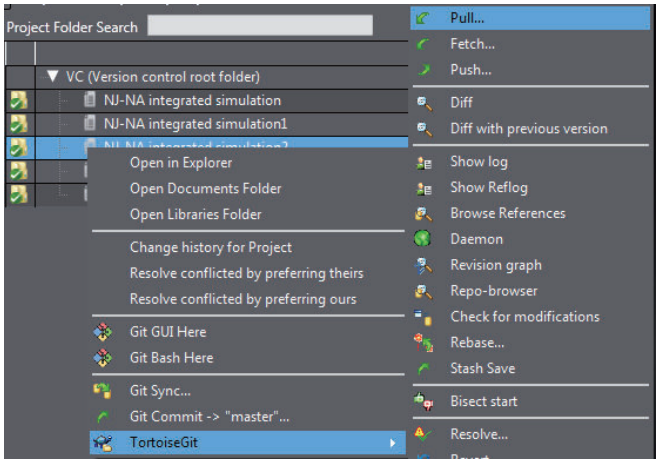
3-2-9 Pulling the Project Data from the Remote Repository to the Local Repository

To acquire the latest project data from the remote repository, pull the project data from the remote repository to the local repository. Use the following procedures to pull the project data.

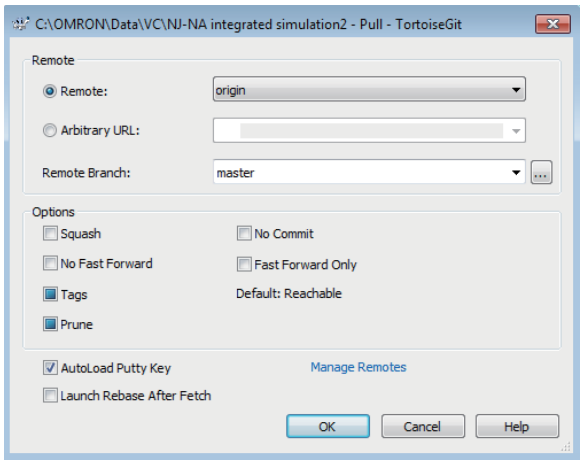
Precautions for Correct Use

Confirm that you committed the target project to the local repository before you pull the project data from the remote repository.

1 Right-click the target project in the **Version Control Projects** window and select **TortoiseGit** - **Pull** from the pop-up menu. Or, right-click the folder icon in the Multiview Explorer and select **TortoiseGit** - **Pull** from the pop-up menu.



The "TortoiseGit" **Pull** dialog box is displayed.



2 Select the **Remote** option, select the target remote repository, and then click the **OK** button. A dialog box is displayed to indicate the completion of the pulling.

3 Click the **Close** button.

3-2-10 Merging the Changes

Apply the changes that you made in a branch to project data in another branch where the master is located.

Use this procedure, for example, to apply changes in a branch to the latest project data in a remote repository.

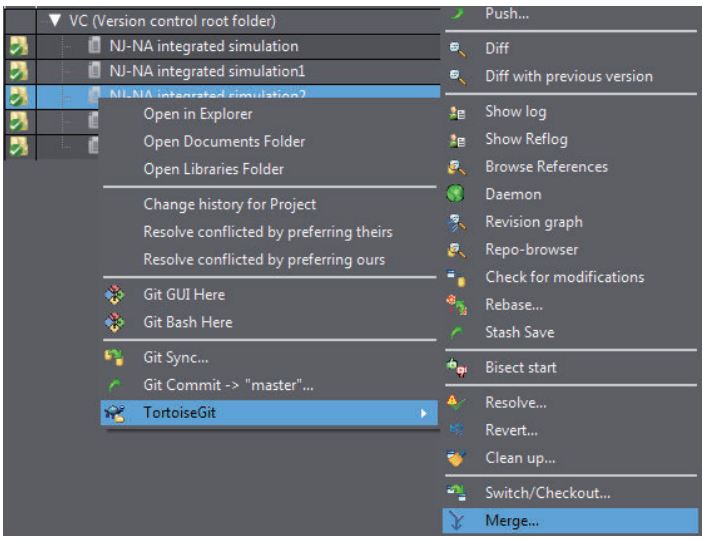
Here, we explain the procedure to merge the changes that you made in a branch into the project data in the local repository as an example, after pulling the latest project data from a remote repository to the local repository.

Before you start the following procedure, switch to the branch in which to edit the master. For the procedure to switch the branch, refer to *3-2-8 Switching to the Branch* on page 3-18.

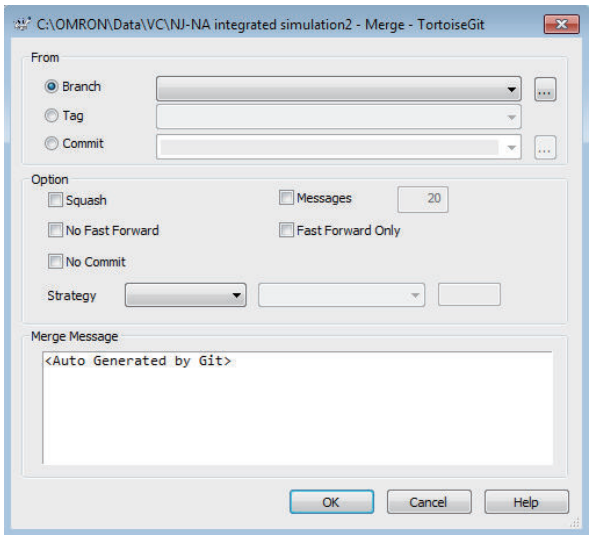
Precautions for Correct Use

Confirm that you committed the target project to the local repository before you merge the changes.

1 Right-click the target project in the **Version Control Projects** window and select **TortoiseGit** - **Merge** from the pop-up menu. Or, right-click the folder icon in the Multiview Explorer and select **TortoiseGit** - **Merge** from the pop-up menu.

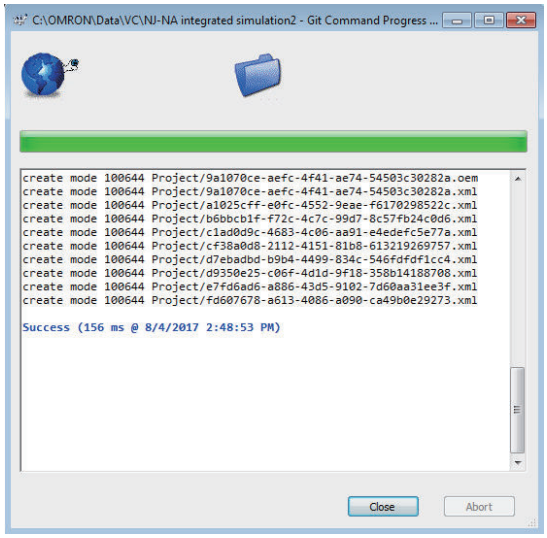


The "TortoiseGit" **Merge** dialog box is displayed.



2 Select the **Branch** option under **From**, select the branch to merge, and then click the **OK** button.

A dialog box is displayed to indicate the completion of the merging.



3 Click the **Close** button.

Additional Information

If the same portion has been changed by another operator after you made changes in the branch, a conflict occurs and the merging fails.

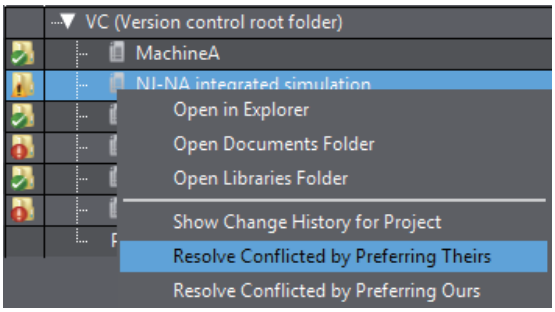
```
git.exe merge remotes/origin/master

Updating c3bff19..ecf209b
error: Your local changes to the following files would be overwritten by merge:
Project/4a81a0b9-b8f7-492f-ad79-6e85f4bceaa0.xml
Project/61dfd2be-48b8-49d0-8f86-3a5212f3980a.dat
Project/9a1070ce-aefc-4f41-ae74-54503c30282a.oem
Project/c1ad0d9c-4683-4c06-aa91-e4edefc5e77a.xml
Please commit your changes or stash them before you merge.
Aborting


git did not exit cleanly (exit code 1) (47 ms @ 8/7/2017 10:34:33 AM)
```

In this case, follow the procedure for *Resolving a conflict* after completion of the merging procedure. Refer to *3-2-11 Resolving a Conflict* on page 3-23 for this procedure.
If a conflict occurs, be sure to follow the procedure for **Resolving a conflict** in the Sysmac Studio. Do not resolve the conflict directly from Windows Explorer, by using the "TortoiseGit"'s **Resolve** menu command on Sysmac Studio files.

3-2-11 Resolving a Conflict

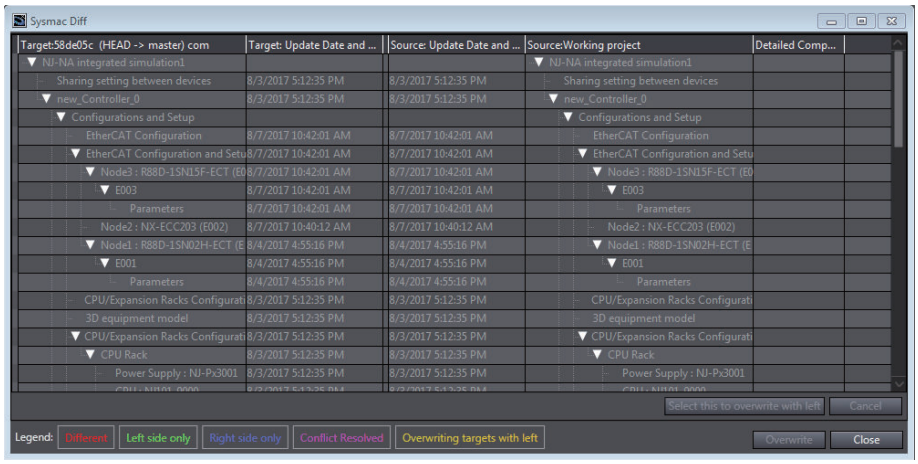
If the same portion of data has been changed in the merge source and in the merge target, a conflict occurs at the time of merging and the merging of the portion fails. At this time, you cannot open the project until you resolve the conflict.
Follow the procedure for *Resolving a conflict* to preferentially apply whichever changes you made in the source or target of merge.
Use the following procedure to merge the project data.

1 After you close the dialog box that displays the results of merge processing, right-click the target project in the **Version Control Projects** window and select **Resolve Conflicted by Preferring Ours** or **Resolve Conflicted by Preferring Theirs** from the pop-up menu.



To give priority to project data in the current working branch, select **Resolve Conflicted by Preferring Ours**. To give priority to project data in the branch to merge, select **Resolve Conflicted by Preferring Theirs**.

The selected changes are applied preferentially and the **Sysmac Diff** dialog box is displayed.



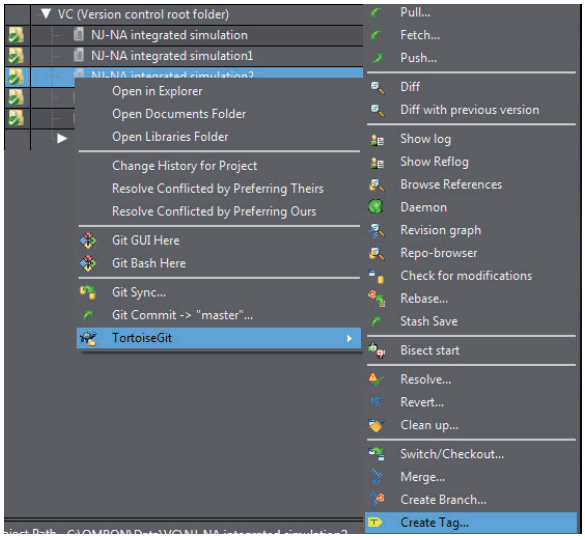
In the **Sysmac Diff** dialog box, check the location of the conflicts that occurred and their resolution result.
Refer to *4-3 Sysmac Diff Dialog Box* on page 4-8 for details on the **Sysmac Diff** dialog box.

2 After checking the result, click the **Close** button.
The **Sysmac Diff** dialog box is closed.
After you apply changes, you need to commit them to the local repository. Refer to *3-2-2 Committing the Changes* on page 3-10 for the commit procedure.

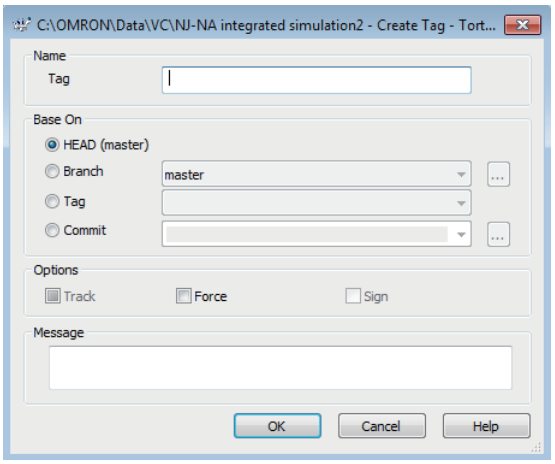
3-2-12 Adding a Tag

"TortoiseGit" provides a function to add a tag as a mark to indicate that the data has been changed.
Adding a tag as a mark is useful when you release a project as a complete edition after you finished editing programs.
Normally, you add a tag to data before you push it to the remote repository.
Use the following procedure to add a tag to the project data.

1 Right-click the target project in the **Version Control Projects** window and select **TortoiseGit** - **Create Tag** from the pop-up menu.



The **Create Tag** dialog box is displayed.



2 Enter the name of the tag in **Tag**, and click the **OK** button.
The tag is added.

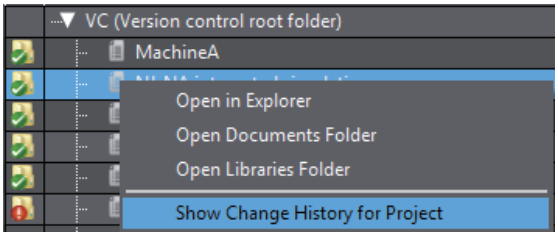
Additional Information

To register data added with a tag in the remote repository, select **Include Tags** check box under **Options** in the "TortoiseGit" **Push** dialog box.

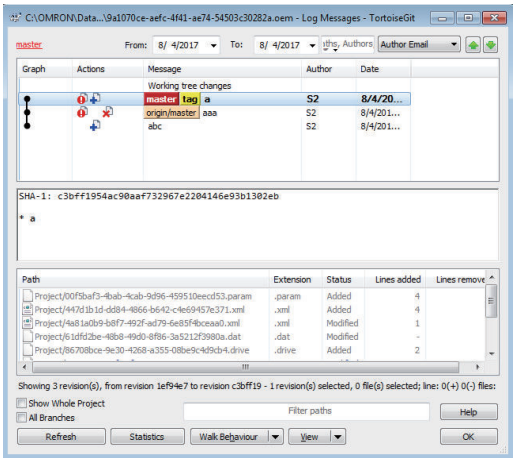
3-2-13 Displaying the Version Control Log

You can use the change record of the project as control log information to check the states of project data in branches, in the local repository, and in remote repositories.

1 Right-click the target project in the **Version Control Projects** window and select **Show Change History for Project** from the pop-up menu. Or, right-click the folder icon in the Multi-view Explorer and select **Show Change History for Project** from the pop-up menu.



The "TortoiseGit" **Log Messages** dialog box is displayed.



The **Log Messages** dialog box displays the change record of the project.
In the message area, the message that you entered at the time of committing changes is displayed, which allows you to check the change record of the project. The dialog box also displays the status of the master and branches in the local repository and the remote repository.
Refer to the "TortoiseGit" Help for details on the "Log Messages" dialog box.

Precautions for Correct Use

Be sure to display the **Log Messages** dialog box from the Sysmac Studio. Do not display the **Log Messages** dialog box by directly selecting the project folder in Windows Explorer.

Additional Information

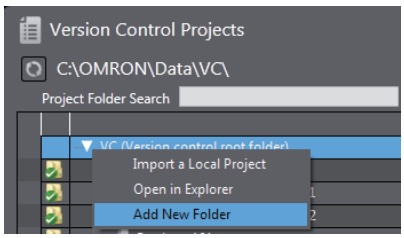
You can also select **TortoiseGit** - **Show log** in the pop-up menu to display the **Log Messages** dialog box. The **Log Messages** dialog box that is displayed by selecting **TortoiseGit** - **Show log** also displays the change record of the files in the *Document* and *Lib* folders.

3-2-14 Cloning Project Data

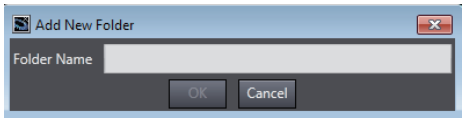
Create a version-controlled project in the local repository by cloning the existing data from a remote repository.
Here, we explain the procedure to clone project data by creating a new folder *ProductA* directly under the VC (Version control root folder) and then cloning the project *ProductA* from a remote repository to the created folder, as an example.

1 On the start page of Sysmac Studio, select **Version Control** - **Version Control Explorer**.
The **Version Control Projects** window is displayed.

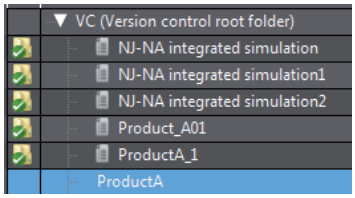
2 Right-click **VC (Version control root folder)** and select **Add New Folder**.



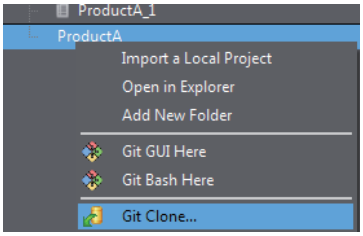
The **Add New Folder** dialog box is displayed.



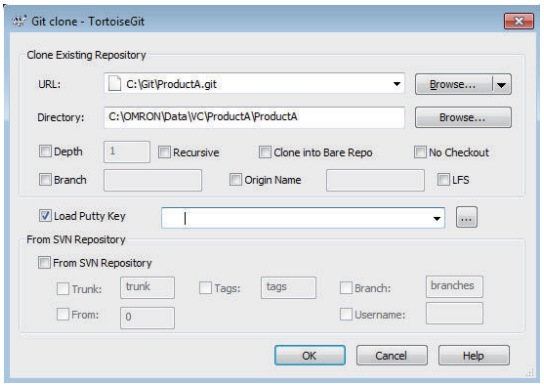
3 Enter the folder name, and click the *OK* button. Here, we enter *ProductA* as the folder name, as an example.
The **ProductA** folder is created.



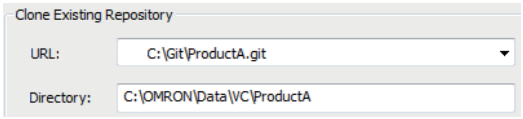
4 Right-click the **ProductA** folder and select **Git Clone** from the pop-up menu.



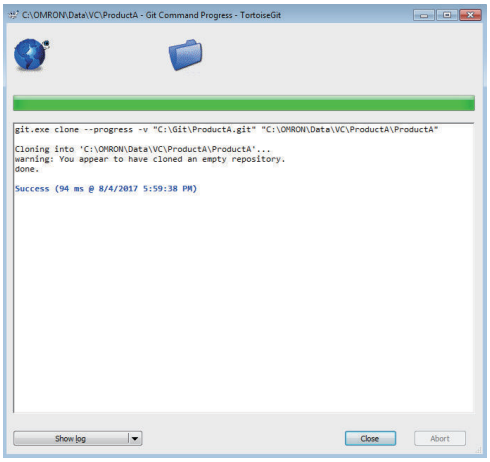
The "TortoiseGit" **Git Clone** dialog box is displayed.



5 Specify the *URL* of the remote repository from which to clone the target project data and the folder (*directory*) to which to clone the project data. Then, click the *OK* button. Here, you enter *ProductA.git*, which is located in the remote repository in a shared folder, in the *URL*, and *ProductA*, which is located directly under the *VC (Version control root folder)*, in *Directory*.

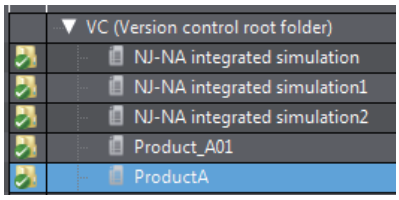


A dialog box is displayed to indicate the completion of the cloning.



6 Click the **Close** button.

The project data is cloned to the *ProductA* folder. You are now ready to use the version control function.



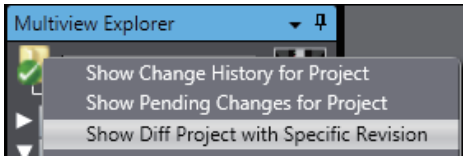
3-2-15 Comparing Project Data

Compare the contents of the current project or a specific revision of the project with the contents of another revision of the project.

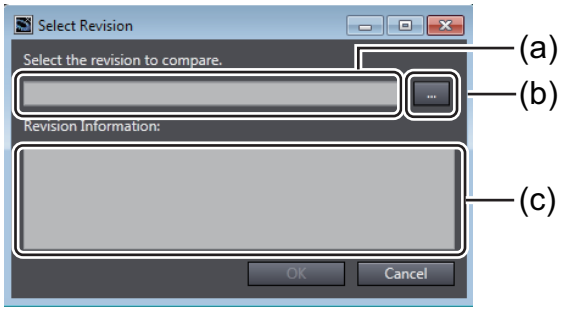
Comparing the Current Project with a Specific Revision of the Project

Compare the current project with a specific revision of the project.

1 Right-click the folder icon in the Multiview Explorer and select **Show Diff Project with Specific Revision** from the pop-up menu.

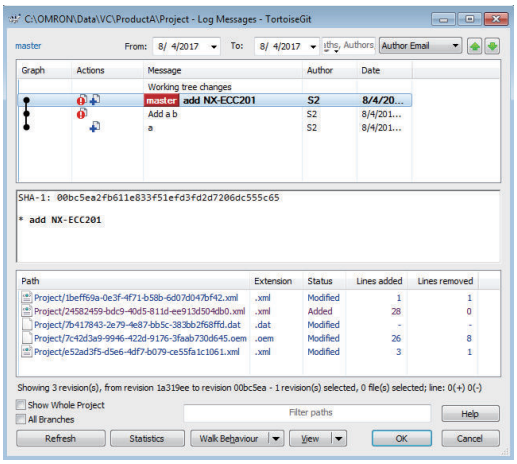


The **Select Revision** dialog box is displayed.

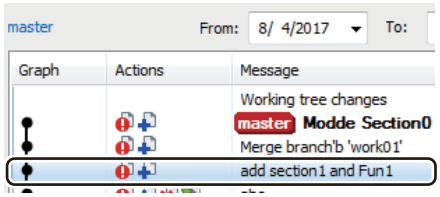


	Item	Description
(a)	Revision to compare	The hash ID (SHA-1) of the selected revision to compare is displayed.
(b)	Selection button	Use this button to display a dialog box in which you can select the revision to compare.
(c)	Revision Information	The message about the selected revision entered when changes were committed is displayed.

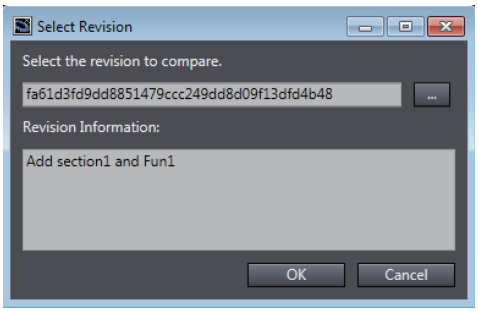
2 Click the selection button.
The "TortoiseGit" **Log Messages** dialog box is displayed.



3 Select the revision to compare and then click the **OK** button.

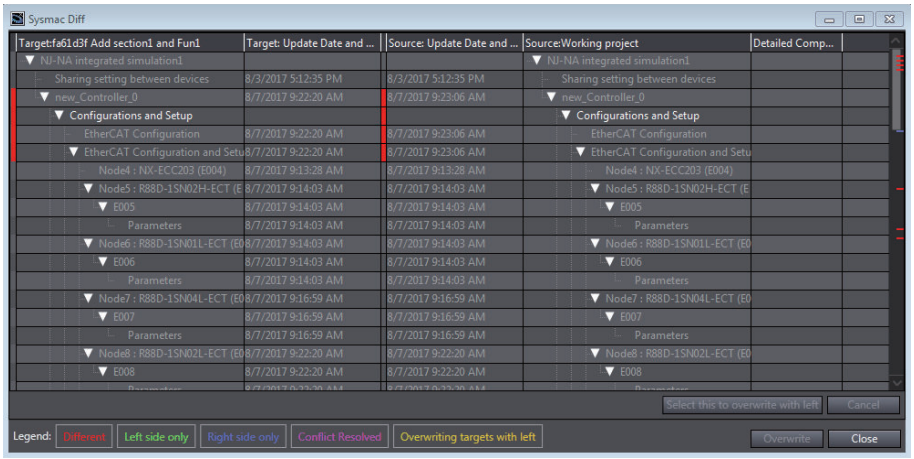


In the **Select Revision** dialog box, the hash ID (SHA-1) of the selected revision and the message about the revision are displayed.



4 Click the **OK** button.

The **Sysmac Diff** dialog box is displayed.



The dialog box displays the contents of the selected project revision on the left side and the contents of the current project on the right side.

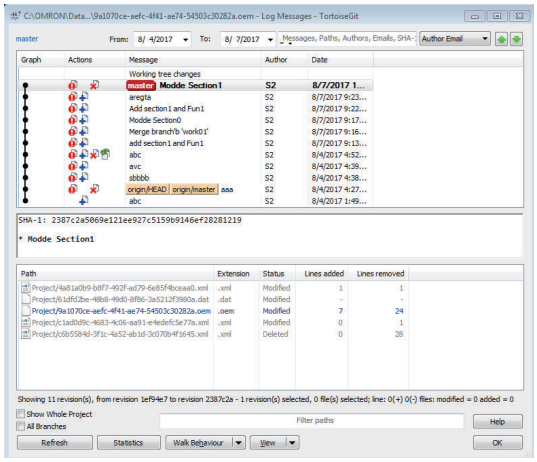
Refer to *4-3 Sysmac Diff Dialog Box* on page 4-8 for details on the contents of the **Sysmac Diff** dialog box.

5 Click the **Close** button.

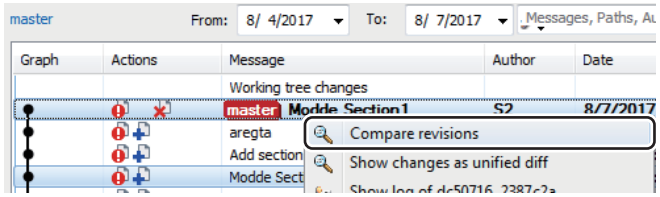
Comparing Two Revisions

Compare any two revisions from those listed in the change record of the project.

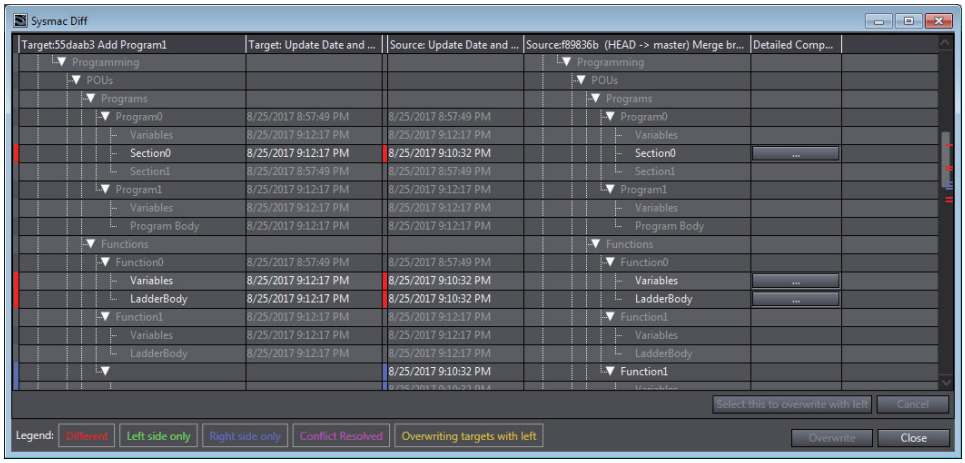
1 Right-click the target project in the **Version Control Projects** window and select **Show Change History for Project** from the pop-up menu. Or, right-click the folder icon in the Multi-view Explorer and select **Show Change History for Project** from the pop-up menu.
The "TortoiseGit" **Log Messages** dialog box is displayed.



2 Select two revisions to compare while you press and hold the Ctrl key, right-click, and select **Compare revisions** from the pop-up menu.



The **Sysmac Diff** dialog box is displayed.



3 Check the difference between the revisions and click the **Close** button.

4 Click the **OK** button in the **Log Messages** dialog box.

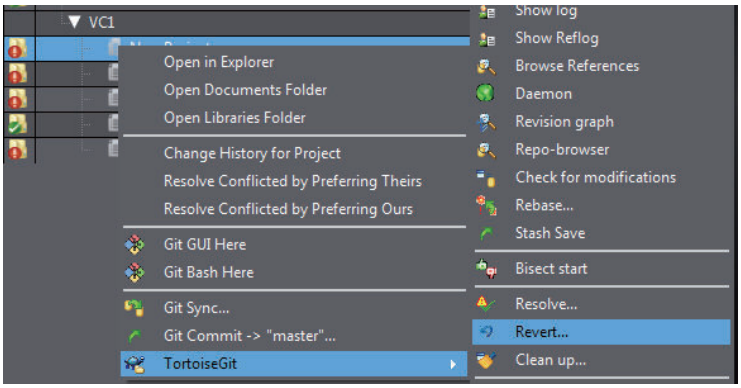
Precautions for Correct Use

Be sure to compare two revisions in the **Log Messages** dialog box that can be displayed by selecting **Show Change History for Project** from the pop-up menu. If you display the **Log Messages** dialog box by selecting **TortoiseGit** - **Show log**, the **Sysmac Diff** dialog box is not displayed.

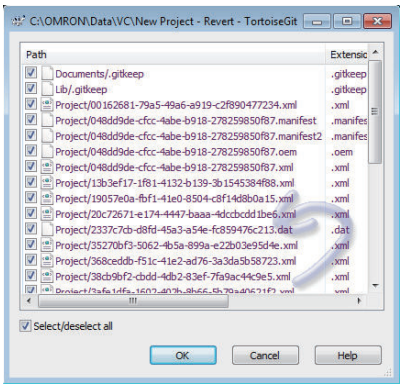
3-2-16 Canceling Changes

Revert the contents of the project to the point when they were last committed by canceling the changes made after that point.

1 Right-click the folder to cancel changes in the **Version Control Projects** window and select **TortoiseGit** - **Revert** from the pop-up menu.

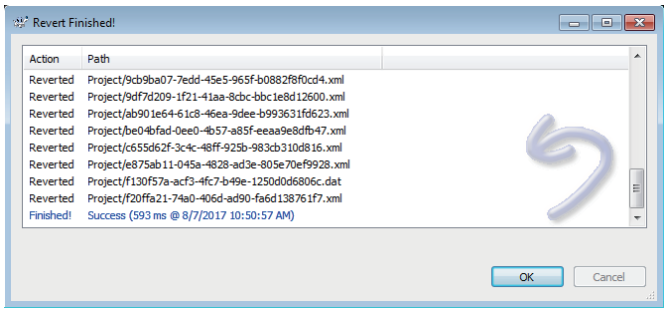


The "TortoiseGit" **Revert** dialog box is displayed.

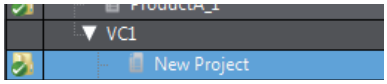


If not all files are selected, be sure to select the **Select/deselect all** check box to select all files.

2 Make sure that all files are selected, and then click the **OK** button.
The "TortoiseGit" **Revert Finished** dialog box is displayed.



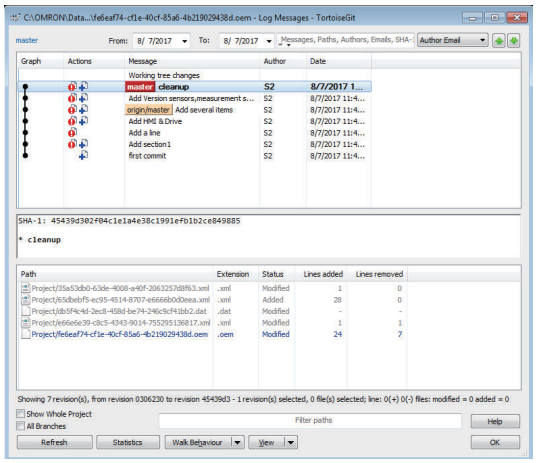
3 Click the **OK** button.
The changes are canceled and the project is reverted to the point when they were last committed.



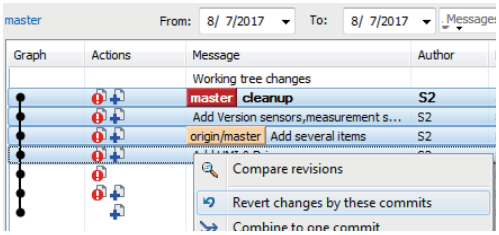
3-2-17 Restoring a Project

For version-controlled projects, you can restore the contents of version controlled projects to an arbitrary older revision. You can restore projects in the "TortoiseGit" **Log Messages** dialog box.
The following describes the procedure to restore the contents of the third latest project revision as an example, by canceling the changes made up to second latest revision.

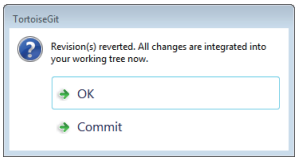
1 Right-click the target project in the **Version Control Projects** window and select **TortoiseGit** - **Show log** from the pop-up menu. Or, right-click the folder icon in the Multiview Explorer and select **TortoiseGit** - **Show log** from the pop-up menu.
The **Log Messages** dialog box is displayed.



2 Select one row after another up to the revision whose contents you want to restore while holding down the Shift key, right-click it, and select **Revert changes by these commits** from the pop-up menu.

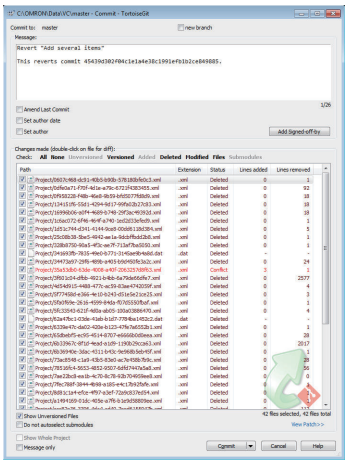


The result dialog box of "TortoiseGit" is displayed.



3 Click the **Commit** button.

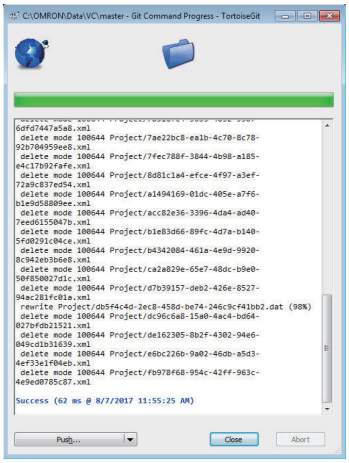
The "TortoiseGit" **Commit** dialog box is displayed.



In the **Message** area, comments on the revert processing are automatically displayed.

4 Confirm that the check boxes for all files listed under **Changes made** are selected, and click the **Commit** button.

A dialog box is displayed to indicate the completion of the commit processing.



5 Click the **Close** button.

This completes the procedure to revert a project.

Precautions for Correct Use

Be sure to restore projects in the **Log Messages** dialog box that can be displayed by selecting **TortoiseGit** - **Show log**. If you display the **Log Messages** dialog box by selecting **Show Change History for Project** from the pop-up menu, you may not be able to restore projects.

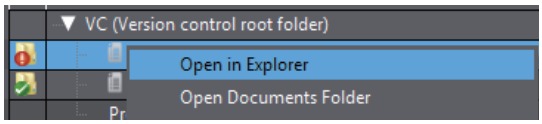
Additional Information

If the commits to revert include changes made to files in the *Document* or *Lib* folder, these changes will also be canceled.

3-2-18 Deleting a Project

Delete a version-controlled project from the local repository.

1 Right-click the project folder to delete in the **Version Control Projects** window and select **Open in Explorer** from the pop-up menu.



The target project folder is displayed.

2 Move one level above the displayed folder and delete the project folder in which the project to delete is stored.

The project is deleted from the local repository. You can now check that the target project folder no longer exists in the **Version Control Projects** window.

Additional Information

If you have the same data as that of the project you removed from the local repository, you can clone the project data to the local repository.
In this case, however, you cannot redo the changes that you made in the local repository after pulling the project data from the remote repository.
Refer to *3-2-14 Cloning Project Data* on page 3-26 for the procedure to clone project data.

3-3 Precautions on Use of Project Version Control

This section describes the precautions common to all devices and precautions for devices that require attention for safe use of project version control.
For devices for which no precautions are given here, you can edit projects in the same way as you do for projects that are not version controlled.

3-3-1 Precautions Common to All Devices

Precautions that are common to all devices are given below.

- You cannot import password-protected projects to the version control system. Disable the password protection before you import the project.
- If you develop a project with multiple developers, all of the people involved should use the Sysmac Studio with the same language settings.
- You cannot merge changes to data in the Multiview Explorer, except for the following data. This data will be always overwritten by the contents of either the source or target of merge.
  - Controller's data in **Programming** and lower-level folders
  - HMI's data in Page and lower-level folders

Additional Information

To enable protection on a version-controlled project, you must export and remove the project from the version-controlled project folder. Then, import the exported project to the Sysmac Studio as a project that is not version controlled and enable protection.
Refer to *A-2 Exporting a Version-controlled Project* on page A-5 for the export procedure. Refer to *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for the procedure to import a project as a project that is not version controlled.

3-3-2 Controllers

Observe the following precautions when you control the versions of projects that include Controllers.

Use

You cannot merge changes to the Controller's *Configurations and Setup*. This data will be always overwritten by the contents of either the source or target of merge.

- If you develop Controller programs with multiple developers, allow a person who supervises the project development to edit the *Configurations and Setup*.
  You may not be able to merge the changes made to the *Configurations and Setup* as intended if the *Configurations and Setup* is edited by multiple developers.
- We recommend that you remove data (such as data trace settings) configured in the *Configurations and Setup* that you added for the purpose of program debugging before you commit changes to the project.

Unit Version

- You can use only Controllers with unit version 1.16 or later in version-controlled projects.
  You cannot add Controllers with unit version 1.15 or earlier to version-controlled projects, or change the Controllers to unit version 1.15 or earlier.
- You cannot transfer data from Controllers with unit version 1.15 or earlier to the computer.

Controller Functions

For version-controlled projects, you cannot set data protection on POUs for Controllers.
Moreover, if a Controller in a version-controlled project has a data-protected POU, you cannot save the project. Release the data protection before you save the project.

3-3-3 HMIs

Observe the following precautions when you control the versions of projects that include HMIs.

Use

You cannot merge changes to the HMI's **Configurations and Setup**, and data in folders except for Page. This data will be always overwritten by the contents of either the source or target of merge.

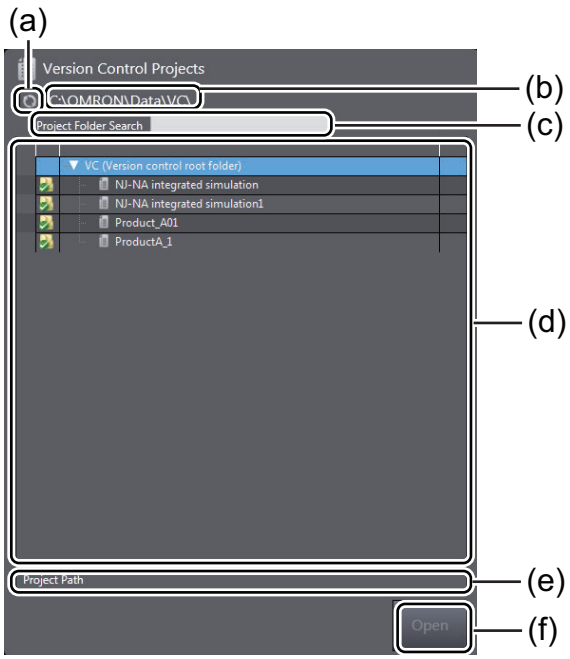
4

Parts of the Window

This section gives the names and functions of window parts related to the Sysmac Studio version control function.

4-1 Version Control Projects Window

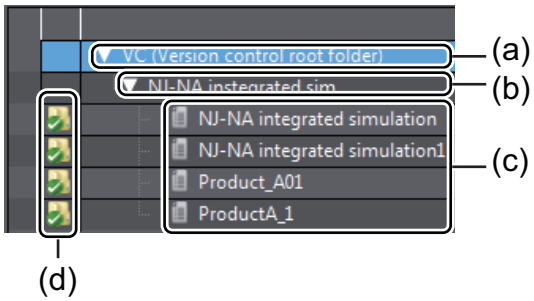
The Version Control Projects window is the entrance from where you add or open version-controlled projects.



	Item	Description
(a)	**Update** button	Use this button to update the version-controlled project list.
(b)	Version control root folder path	The path to the *VC (Version control root folder)* is displayed. You cannot edit this path because it is fixed.
(c)	Project folder search box	The version-controlled project folder list is narrowed to show only those project folders that contain the entered text in the folder name. The text in the folder name is shown in blue.
(d)	Version-controlled project folder list	Folders that contain version-controlled projects are listed.
(e)	Project location	The folder path of the selected project is displayed.
(f)	**Open** button	Use this button to open the selected project.

4-1-1 Version Control Project Folder List

This list displays folders that contain version-controlled projects.



	Item	Description	Pop-up menu command
(a)	VC (Version control root folder)	This displays the root folder under which you save version-controlled projects. You cannot edit this path because it is fixed. Under this folder, create folders in which to save projects.	• Import a Local Project • Open in Explorer • Add New Folder
(b)	User folder	This is a user-created folder.	• Import a Local Project • Open in Explorer • Add New Folder
(c)	Version-controlled project folders	These are folders that contain version-controlled projects. An error icon is displayed at the left end of the project if it is in an invalid state.	• Open in Explorer • Open Documents Folder • Open Libraries Folder • Show Change History for Project • Resolve Conflicted by Preferring Theirs • Resolve Conflicted by Preferring Ours • Git GUI Here • Git Bash Here • Git Sync • Git Commit -> "master" • TortoiseGit
(d)	Icons to show version control status	These icons show the control status of version-controlled projects. Refer to *Status Icons for Version-controlled Projects* on page 4-5 for details on these icons.	---

Precautions for Correct Use

In Windows Explorer, each version-controlled project is displayed as a folder, under which the *Project*, *Document*, and *Lib* folders are stored. Do not change the names and structure of these folders. Note also that you must not create new folders in the same hierarchical level as the *Project*, *Document*, and *Lib* folders. Otherwise, the version control function may not operate normally.

Pop-up Menu Commands

The table below describes the pop-up menu commands that appear when you right-click an item in the version-controlled project list.

You can right-click an item in the version-controlled project list to display the corresponding pop-up menu, as follows.

Menu command	Description
Import a Local Project	The **Import a Local Project** dialog box is displayed. You can import projects that are currently not version controlled on the computer into the selected folder.
Open in Explorer	The selected folder is displayed in Windows Explorer.
Add New Folder	A new folder is created under the selected folder.
Open Documents Folder	The *Document* folder that is located under the selected folder is displayed in Windows Explorer. Files in the *Document* folder are controlled as a part of the version-controlled project. To control the versions of design data or other externally created documents together with the project, add them to the *Document* folder.

Menu command	Description
Open Libraries Folder	The *Lib* folder that is located under the selected folder is displayed in Windows Explorer. Files in the *Lib* folder are controlled as a part of the version-controlled project. If Controllers in the project reference libraries, add the applicable library files to the *Lib* folder.
Show Change History for Project	The "TortoiseGit" **Log Messages** dialog box displays the change record of the project.
Resolve Conflicted by Preferring Theirs	If a conflict occurred at the time of merging, select this command to display the *Sysmac Diff* dialog box and overwrite the data in the merge target with the changes made to the merge source. If no conflict occurred, selecting this displays a message box that indicates that no collision has occurred.
Resolve Conflicted by Preferring Ours	If a conflict occurred at the time of merging, select this command to display the *Sysmac Diff* dialog box and overwrite the data in the merge source with the changes made to the merge target. If no conflict occurred, selecting this displays a message box that indicates that no collision has occurred.
Git GUI Here	The standard operation window of Git is displayed. This command is not used for the Sysmac Studio version control function.
Git Bash Here	The standard command-line window of Git is displayed. This command is not used for the Sysmac Studio version control function.
Git Sync	The "TortoiseGit" **Git Sync** dialog box is displayed.
Git Commit -> "master"	The "TortoiseGit" **Commit** dialog box is displayed. The string enclosed in double quotes (*""*) represents the name of the current working branch.
TortoiseGit	The submenu commands of "TortoiseGit" are displayed. Refer to *"TortoiseGit" Menu Commands* on page 4-4 for details.

● "TortoiseGit" Menu Commands

The table below describes the pop-up menu commands of "TortoiseGit".
Among the submenu commands included in **TortoiseGit** pop-up menus, these commands are mainly used for the Sysmac Studio version control function.
For the functions of other "TortoiseGit" menu commands, refer to the "TortoiseGit" Help.

Menu command	Description	Reference
Pull	The "TortoiseGit" **Pull** dialog box is displayed.	*3-2-9 Pulling the Project Data from the Remote Repository to the Local Repository* on page 3-20
Push	The "TortoiseGit" **Push** dialog box is displayed.	*3-2-4 Pushing the Project Data to the Remote Repository* on page 3-13
Show Log	The "TortoiseGit" **Log Messages** dialog box is displayed.	*3-2-13 Displaying the Version Control Log* on page 3-25
Revert	The "TortoiseGit" **Revert** dialog box is displayed.	*3-2-16 Canceling Changes* on page 3-31
Switch/ Checkout	The "TortoiseGit" **Switch/Checkout** dialog box is displayed.	*3-2-8 Switching to the Branch* on page 3-18
Merge	The "TortoiseGit" **Merge** dialog box is displayed.	*3-2-10 Merging the Changes* on page 3-21
Create Branch	The "TortoiseGit" **Create Branch** dialog box is displayed.	*3-2-7 Creating a Branch* on page 3-17

Menu command	Description	Reference
Create Tag	The "TortoiseGit" **Create Tag** dialog box is displayed.	*3-2-12 Adding a Tag* on page 3-24

Additional Information

You can change the menu commands displayed in "TortoiseGit" submenus as desired by configuring "TortoiseGit". Refer to the "TortoiseGit" Help for details.

Status Icons for Version-controlled Projects

The table below describes icons that represent the status of a version-controlled project.

Icon	Description
	The project remains unchanged.
	The project has been changed, but not committed.
	A conflict occurred between the changes to the merge source project and the changes to the merge target project at the time of merging.
	The newly added project has not been committed.

Additional Information

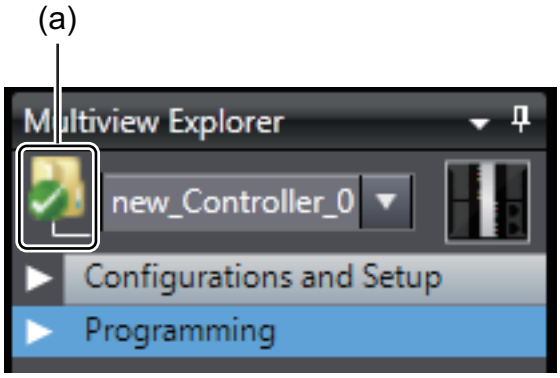
You can change the icons by configuring "TortoiseGit". Refer to the "TortoiseGit" Help for details.

4-2 Multiview Explorer

The table below describes the window parts of the Multiview Explorer related to the version control function.

4-2-1 Version Control Icon

A version control icon is displayed if it is a version-controlled project.



	Item	Description
(a)	Status icon for version-controlled project	This icon represents the status of the version-controlled project. Refer to *Status Icons for Version-controlled Projects* on page 4-5 for details on these icons.

Pop-up Menu Commands

The table below describes the pop-up menu commands that appear when you right-click the status icon for version-controlled project in the Multiview Explorer.
You can right-click the version control project icon to display the corresponding pop-up menu, as follows.

Menu command	Description
Show Change History for Project	The "TortoiseGit" **Log Messages** dialog box is displayed. You can check the change record of the project. Refer to *3-2-13 Displaying the Version Control Log* on page 3-25 for details.
Show Pending Changes for Project	The results of comparison between before and after you edit the project is displayed in the **Sysmac Diff** dialog box. *Pending* means the state in which *the commit of changes is on hold*.
Show Diff Project with Specific Revision	The **Select Revision** dialog box is displayed. You can select the revision to compare and check the results of comparison with the current project. Refer to *Comparing the Current Project with a Specific Revision of the Project* on page 3-28 for details.
Git GUI Here	The standard operation window of Git is displayed. This command is not used for the Sysmac Studio version control function.
Git Bash Here	The standard command-line window of Git is displayed. This command is not used for the Sysmac Studio version control function.
Git Sync	The "TortoiseGit" **Git Sync** dialog box is displayed.
Git Commit -> "master"	The "TortoiseGit" **Commit** dialog box is displayed. The string enclosed in double quotes ("") represents the name of the current working branch.

Menu command	Description
TortoiseGit	The submenu commands of "TortoiseGit" are displayed. Refer to *"TortoiseGit" Menu Commands* on page 4-4 for details.

4-2-2 Menu Commands for Checking Changes for Each Data Item

The following tables show the menu commands for checking changes for each data item.
You can display the menu commands for checking changes by right-clicking a data item in the Multiview Explorer.

Controllers

The following table shows the menu commands for each data item available with Controllers.

Data item	Where to select	Context menu command
Ladder programs	**Programs** - **Program** - **Section**	Show Diff Section with Pending Changes
	Functions - **Function** (Ladder) **Function Blocks** - **FunctionBlock** (Ladder)	Show Diff Ladder with Pending Changes
ST programs	**Programs** - **Program** (ST) **Functions** - **Function** (ST) **Function Blocks** - **FunctionBlock** (ST)	Show Diff ST with Pending Changes
Local variables	**Programs** - **Program** **Functions** - **Function** **Function Blocks** - **FunctionBlock**	Show Diff Variables with Pending Changes
Data types	**Data types** data types	Show Diff Data Types with Pending Changes
Global variables	**Global Variables**	Show Diff Global Variables with Pending Changes

Refer to *Checking Changes for Each Data Item* on page 3-16 for the procedure.

HMIs

The following table shows the menu commands for each data item available with HMIs.

Data item	Where to select	Context menu command
Pages	**Pages** - **Page**	Show Diff Page with Pending Changes
Page subroutines	**Pages** - **Page**	Show Diff Code with Pending Changes
Recipe fields	**Recipes** - **Recipe**	Show Diff Ingredients with Pending Changes
Recipes	**Recipes** - **Recipe**	Show Diff Recipes with Pending Changes
Data types	**Data Types**	Show Diff Data Types with Pending Changes
Global variables	**Global Variables**	Show Diff Global Variables with Pending Changes
Global events	**Global Events**	Show Diff Global Event with Pending Changes
Global subroutines	**Global Subroutines** - **SubroutineGroup**	Show Diff Global Subroutine with Pending Changes
Resources	**Resources** - **Group**	Show Diff Resources with Pending Changes

Refer to *Checking Changes for Each Data Item* on page 3-16 for the procedure.

4-3 Sysmac Diff Dialog Box

The **Sysmac Diff** dialog box displays the difference between projects in terms of the devices registered in the projects.
The **Sysmac Diff** dialog box consists of the project comparison window that displays the comparison results for the entire project and the **Detailed Comparison** window that displays the detailed comparison results for selected items for each device.
Here, we explain the **Sysmac Diff** dialog box windows common to all devices and then those specific to each device.
The devices listed below have specific display items.

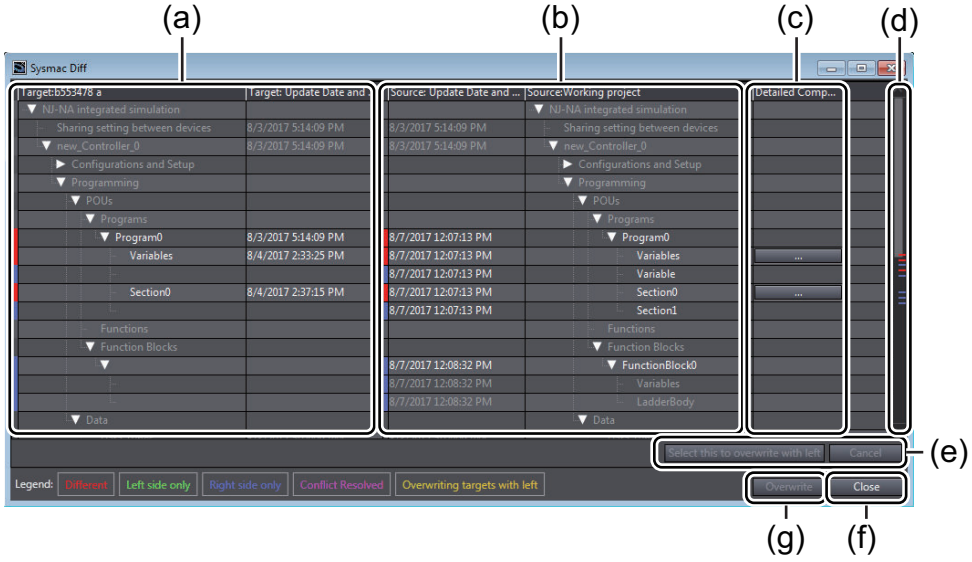
- Controllers
- HMIs
- Drives (Servo Drives/Servomotors)

4-3-1 Common Windows

Information on common Sysmac Diff windows for all devices is given below.

Project Comparison Window

The table below describes the project comparison window of the **Sysmac Diff** dialog box.
The project comparison window of the **Sysmac Diff** dialog box displays the comparison results for all the devices registered in the projects.



	Item	Description	Remarks
(a)	Target project (Target)	The contents of the target project are listed. In the title row, *Target:* is followed by the revision and the comment entered at the time of committing the project.	The leftmost column is colored as shown in the legend, if there are any differences.

	Item	Description	Remarks
(b)	Source project (Source) (Working project)	The contents of the source project are listed. In the title row, *Source:* is followed by the revision and the comment entered at the time of committing the project. For the current project, *Source: Working project* is displayed	The leftmost column is colored as shown in the legend, if there are any differences.
(c)	**Detailed Comparison** button	This button is displayed for each item that has detailed comparison results. Click this button to display the detailed comparison window.	The contents of the detailed comparison window vary depending on the device. For details, refer to the description of the detailed comparison window for each device.
(d)	Scroll bar	If there are differences between the projects, the relevant location is colored based on the legend.	
(e)	**Select this to overwrite with left** button, **Cancel** button	If you overwrite the contents of the source project with the contents of the target project, click the Select Item to Overwrite Data on Right Side by Data on Left Side button to select the item to overwrite. To overwrite the data, click the **Overwrite** button. To cancel the overwriting the data, click the **Cancel** button.	• The selectable items are displayed in white text. • You cannot select items when you *compare two revisions*.
(f)	**Close** button	Use this button to close the **Sysmac Diff** dialog box.	
(g)	**Overwrite** button	Use this button to overwrite data for the item that you selected using the **Select this to overwrite with left** button.	

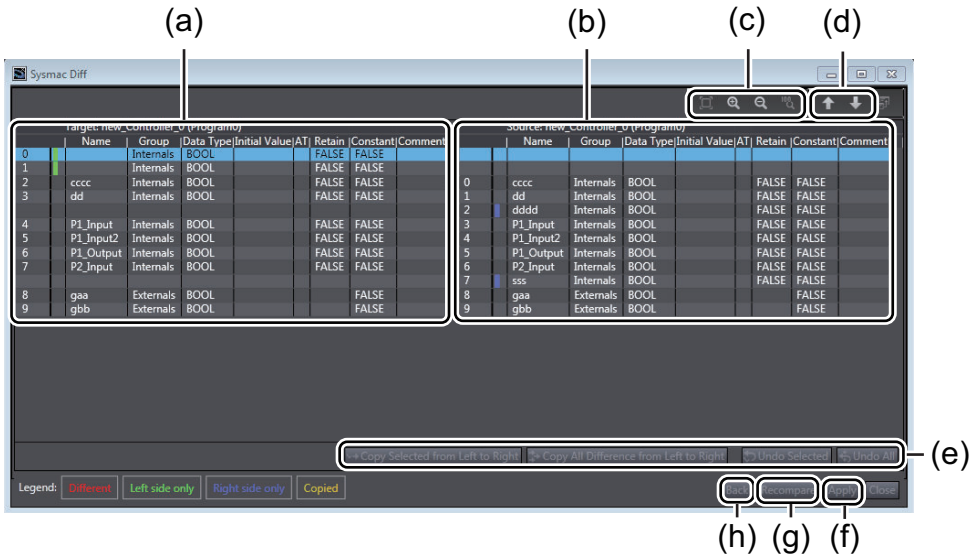
For the following devices, the comparison results are displayed in the order they are displayed in the Multiview Explorer. For details on the displayed items, refer to the description of the *project comparison window* for each device.

Device	Project comparison window
Controllers	*Project Comparison Window* on page 4-11
HMIs	*Project Comparison Window* on page 4-13
Drives	*Project Comparison Window* on page 4-16

Detailed Comparison Window

The table below describes the **Detailed Comparison** window of the **Sysmac Diff** dialog box.
The **Detailed Comparison** window of the **Sysmac Diff** dialog box displays the detailed comparison results for the selected item for each device.
You can display the **Detailed Comparison** window for items that have detailed comparison results by clicking the **Detailed Comparison** button in the project comparison window.
In the **Detailed Comparison** window, you can overwrite the contents of the source project with the contents of the target project for each item to compare.

The description uses as an example the **Detailed Comparison** window for a local variable table for a Controller.



	Item		Description	Remarks
(a)	Target		The contents of the target item are displayed.	
(b)	Source		The contents of the source item are displayed.	
(c)	Zoom-in and zoom-out buttons		Use these buttons to zoom in or zoom out the window display. The buttons are arranged from the left to the right as follows. • Fit to display • Zoom in • Zoom out • Reset zoom in/out	
(d)	Next Difference button		Use this button to show the next difference.	
(e)	Copy difference buttons	**Copy Selected from Left to Right**	Use this button to copy the contents of the selected row from the target to the source.	To apply the results of copying, click the **Apply** button.
		Copy All Difference from Left to Right	Use this button to copy the contents of all the rows that contain differences from the target to the source.	
		Undo Selected	Use this button to undo the results of **Copy Selected from Left to Right**.	You cannot undo the changes once you click the **Apply** button.
		Undo All	Use this button to undo the execution results of **Copy Selected from Left to Right**.	
(f)	**Apply** button		Use this button to apply the results of copying executed by **Copy difference** buttons to the source project.	
(g)	**Recompare** button		Use this button to display the results of re-comparison between the target and the source.	
(h)	**Back** button		Use this button to return to the project comparison window.	

Additional Information

The **Close** button is always disabled.

4-3-2 Controllers

Information on Controller-specific windows is given below.

Project Comparison Window

The project comparison window for Controllers displays the following items.

Item	Support of detailed comparison	Description
Sharing setting between devices		
Configurations and Setup		The displayed items depend on the functions provided by the Controller.
EtherCAT Configuration		
EtherCAT Configuration and Setup		
CPU/Expansion Racks		
CPU Rack		
Units		The Power Supply Unit, CPU Unit, and registered Units are displayed.
I/O Map		
Controller Setup		
Operation Settings		
Built-in EtherNet/IP Port Settings		
Motion Control Setup		
Axis Settings		Under **Axis Settings**, registered items are displayed.
Axes Group Settings		Under **Axes Group Settings**, registered items are displayed.
Cam Data Settings		Under **Cam Data Settings**, registered items are displayed.
Event Settings		
Task Settings		
Data Trace Settings		Under **Data Trace Settings**, registered items are displayed.
EtherNet/IP Connection Settings		
Comparison ID for Configurations and Setup		This ID shows whether the **Configurations and Setup** of the project match the Configurations and Setup of the actual Controller.
3D equipment model		
Library		Under **Library**, registered items are displayed.
Programming		
POUs		
Programs		Under **Programs**, registered programs are displayed.

Item				Support of detailed comparison	Description
		Program Name (Ladder)			
			Variables	○	
			Section Name	○	Registered section items are displayed.
		Program Name (ST)			
			Variables	○	
			ProgramBody	○	
	Functions				Under **Functions**, registered function items are displayed.
		Function Name (Ladder)			
			Variables	○	
			LadderBody	○	
		Function Name (ST)			
			Variables	○	
			ProgramBody	○	
	Function Blocks				
		FunctionBlock Name (Ladder)			
			Variables	○	
			LadderBody	○	
		FunctionBlock Name (ST)			
			Variables	○	
			ProgramBody	○	
Data					
	Data Types			○	
	Global Variables			○	
NC Programs				○	

Detailed Comparison Window

The table below describes the Controller-specific items displayed in the **Detailed Comparison** window of the Sysmac Diff dialog box.

Item	Description	Detailed comparison window
Ladder	The program (sections), functions, and function blocks are compared. You can overwrite the source with the contents of the target in units of rows.	These items are the same as those of the common window. Refer to *Detailed Comparison Window* on page 4-9 .
ST	The structured text of the program, functions, and function blocks are compared. You can overwrite the source with the contents of the target in units of rows.	
Variables	Local variables and global variables are compared. You can overwrite the source with the contents of the target in units of variables.	
Data Type	The members and attributes of a data type are compared. You can overwrite the source with the contents of target in units of members.	
NC Programs	The NC programs are compared. You can overwrite the source with the contents of the target in units of rows.	

4-3-3 HMIs

Information on HMI-specific windows is given below.

Project Comparison Window

The project comparison window for HMIs displays the following items.

Item	Support of detailed comparison	Description
Sharing setting between devices		
Configurations and Setup		
Device References		
Internal Devices		
Controller name		Controllers that are registered in the project are displayed.
External device name		The external Controller name is displayed.
Data Types	○	
Variables	○	
Variable Mapping		
HMI Settings		
Security Settings		
Troubleshooter		
Language Settings		
HMI		
Page		Under **Page**, registered items are displayed.
PageName	○	
PageName.vb	○	

Item	Support of detailed comparison	Description
Group name		
User Alarms		Under **User alarms**, registered items are displayed.
Group name		
Controller Events		
User Events		
Data Logging		Under **Data Logging**, registered items are displayed.
DataSetName		
Data Groups		Under **Data Groups**, registered items are displayed.
DataGroup name		
Recipes		Under **Recipes**, registered items are displayed.
Recipe name.Field	○	
Recipes	○	
Custom Keypads		Under **Custom Keypads**, registered items are displayed.
Group name		
Data		
Data Types	○	
Global Variables	○	
Global Events	○	
Global Subroutines		Under **Global Subroutines**, registered items are displayed.
SubroutineGroup name	○	
Resources		Under **Resources**, registered items are displayed.
Group name	○	
Imported IAGs		
IAG collection name		
IAG name		
Data Types		
Resources		
Scale Transformations		

Detailed Comparison Window

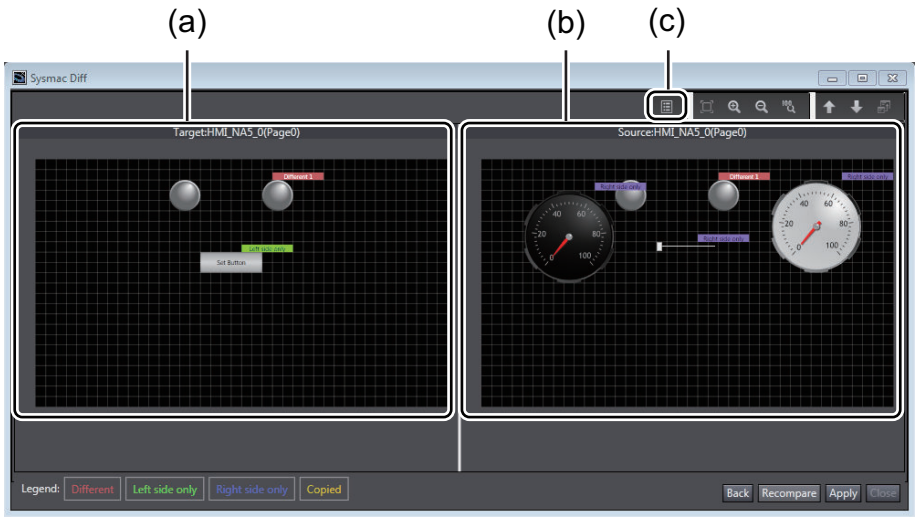
The table below describes the HMI-specific items that are displayed in the **Detailed Comparison** window of the Sysmac Diff dialog box.

Item	Description	Detailed comparison window
Page	The presence/absence and layout of objects on the page as well as property information are compared. You can overwrite the source with the contents of the target in units of objects.	Refer to *Detailed Comparison Window for a Page* on page 4-15.

Item	Description	Detailed comparison window
Page Subroutines	The source code of page subroutines is compared. You can overwrite the source with the contents of the target in units of source code rows.	These items are the same as those of the common window. Refer to *Detailed Comparison Window* on page 4-9 .
Variables	The attributes of global variables are compared. You can overwrite the source with the contents of target in units of variable rows.	
Data Type	The members and attributes of a data type are compared. You can overwrite the source with the contents of target in units of member rows.	
Recipes	The settings of recipes are compared. You can overwrite the source with the contents of the target in units of recipe rows.	
Fields	The settings of fields are compared. You can overwrite the source with the contents of target in units of field rows.	
Global Events	Global events are compared. You can overwrite the source with the contents of target in units of event rows.	
Global Subroutines	The source code of global subroutines is compared. You can overwrite the source with the contents of the target in units of source code rows.	
Resources	The settings of resources are compared. • General-purpose text strings • Alarm text strings • Documents • Pictures • Videos You can overwrite the source with the contents of the target in units of rows, respectively.	This item the same as that of the common window. Refer to *Detailed Comparison Window* on page 4-9 .

● Detailed Comparison Window for a Page

The table below describes the **Detailed Comparison** window for the selected *page* of HMIs.



	Item	Description	Remarks
(a)	Target	The contents of the target page are displayed.	You can copy an object to the source by right-clicking it and selecting **Merge** from the pop-up menu. To apply the results of copying, click the **Apply** button. To undo the changes, click the **Recompare** button.
(b)	Source	The contents of the source page are displayed.	
(c)	Properties button	Use this button to display property information on the selected object in the detailed comparison window. The following property information is displayed. • Properties • Event and action • Animation The property information displayed on the corresponding tab page.	

Additional Information

Other buttons are the same as those of the common window. Refer to *Detailed Comparison Window* on page 4-9 for details.

4-3-4 Drives

Information on windows specific to Drives (Servo Drives/Servomotors) is given below.

Project Comparison Window

The project comparison window for Drives displays the following items.

Item	Support of detailed comparison	Description
Device Group		Under **Device Group**, registered items are displayed.
Drive name		
Parameters	○	

Detailed Comparison Window

The table below describes the Drive-specific items displayed in the **Detailed Comparison** window of the Sysmac Diff dialog box.

Item	Description	Detailed comparison window
Parameters	The object dictionary parameter values are compared. You can overwrite the source with the contents of the target by applying object dictionary values.	These items are the same as those of the common window. Refer to *Detailed Comparison Window* on page 4-9 . For G5-series Servo Drives, however, you cannot use the **Copy Selected from Left to Right** and **Undo Selected** buttons.

A

Appendices

The appendices provide the option settings, error messages lists, and other supplemental information for the body of this manual.

A-1 Errors and Troubleshooting

This appendix describes the cause and handling of each error that may occur during the use of the Sysmac Studio project version control function.

When You Operate on the Start Page

Error message	Cause	Handling
Necessary version control tool is not installed. Please confirm that the following tools are installed. • Tortoise GIT Ver2.4.0 or higher • GIT Ver2.13.0 or higher	"TortoiseGit" or Git is not installed, or the installed version of "TortoiseGit" or Git is lower than the specified version.	Install the specified version of "TortoiseGit" or Git.
	You cannot use Git commands from the Sysmac Studio because you selected **Use Git from Bash only** on the **Adjusting your PATH environment** page during Git installation.	Reinstall Git and, on the **Adjusting your PATH environment** page that is displayed during installation, select **User Git from the Windows Command Prompt**.

When You Open the Project

Error message	Cause	Handling
Since file change/addition/deletion was detected in the files that are not version controlled, clean up of the project is performed. Please do not modify the file from other than the version control tool.	When you edit the project in the Sysmac Studio, some files that are not version controlled (build results etc.) were modified from other than the Sysmac Studio.	The project is cleaned up automatically (all files that are not version controlled in the source are deleted) and then opened.
Failed to clean up the project. Close the files if the files are opened with a program like a file editor, and then reopen the project.	An error occurred in the process to delete all files that are not version controlled.	Make sure that the files are not opened with an application other than the Sysmac Studio, and then reopen the project.
Data that does not support the development by multiple developers were edited outside the Sysmac Studio. Do not edit the data outside the Sysmac Studio (including conflict resolution), cancel the changes and restore the data to the state that does not include the changes made outside the Sysmac Studio.	Among files that are version controlled, a file that does not support the development by multiple developers was edited from other than the Sysmac Studio.	Perform *Revert*, and then reopen the project. Refer to *3-2-16 Canceling Changes* on page 3-31 for the procedure to *Revert*.
Some project files do not exist. Restore the files by canceling the changes.	Among files that are version controlled, more than one file does not exist in the project folder.	Perform *Revert*, and then reopen the project.
Some project files are read only. Clear read only attribute of them.	There are project files that are read only.	Clear read only attribute of the files in the project folder, and then reopen the project.

Error message	Cause	Handling
Some project files are broken. Do not edit the data outside the Sysmac Studio (including conflict resolution), cancel the changes and restore the data to the state that does not include the changes made outside the Sysmac Studio.	The project contains invalid data and cannot be opened.	Perform *Revert*, and then reopen the project.
There are unnecessary files in the project folder. Do you want to open the project after deleting the files?	There are files that are not necessary for the project in the project folder or lower-level folders.	If you want to delete the unnecessary files, select **Yes**. If you do not want to delete the files automatically, select **No**, select the files to delete, and then reopen the project.
Failed to remove unnecessary files. Close the files if the files are opened with a program like a file editor, and then reopen the project.	The unnecessary files cannot be deleted.	If the files are opened with a program like a file editor, close the files, and then reopen the project.
This project is conflicted and cannot be opened. Resolve the conflict to open the project.	A conflict exists in the project.	Perform *Resolve a conflict*. Refer to *3-2-11 Resolving a Conflict* on page 3-23 for details on *Resolve a conflict*.
The automatic merge can not be performed because there are devices duplicated by uploading from the controller.	The project contains both the devices uploaded from the Controller and the devices located in the download source.	Delete either of these devices, or revert the project to the state before you performed the upload operation.
Inconsistencies were found in the project data. Cancel the modification to revert the data, resolve inconsistencies, and then reopen the project. <Details>	Since you modified project files outside the Sysmac Studio, the project data is in a state described in <Details>.	Cancel the changes made outside the Sysmac Studio to revert the data, resolve inconsistencies, and then reopen the project.

When You Operate the Version Control Function

Error message	Cause	Handling
An error occurred when executing the command of XXX.	The Git command failed during the use of the Sysmac Studio version control function.	Perform troubleshooting according to the Git message displayed after this message.

When You Edit the Project

Error message	Cause	Handling
The project files were updated. In order to reload the updated files, please reopen the project.	When you edit the project in the Sysmac Studio, some files that are not version controlled (build results etc.) were modified from other than the Sysmac Studio.	Close and reopen the project.

When You Save the Project

Error message	Cause	Handling
Save the project again.	An error occurred when the project was saved, in the Git processing to register a file in the version control system.	Perform troubleshooting according to the Git error message displayed after this message, and save the project again.

When You Resolve a Conflict

Error message	Cause	Handling
Processing was interrupted because an error occurred.	An error occurred when you perform *Resolving a conflict*.	Perform troubleshooting according to the instruction displayed after this error message.
Conflicts other than the merge operation can not be resolved. Please cancel the changes to the project.	The conflict occurred because you executed **Cherry pick this commit (Use)** in the "TortoiseGit" **Log Messages** dialog box.	Perform *Revert*.

Sysmac Diff Dialog Box

Error message	Cause	Handling
Sysmac Diff will now shutdown. Please try the operation again.	An error occurred during interaction with "TortoiseGit".	Try the same operation again.
Fail to open diff view	The operation was performed when the changes was not committed.	Commit the changes, and then try the operation.
	An error occurred when the Detailed Comparison window was displayed.	Try the same operation again. If the error message still appears, perform *Revert* and open the project again.
An error occurred while overwriting. The processing will be aborted and the project will be closed.	An error occurred during interaction with "TortoiseGit".	Perform *Revert*, and then reopen the project.

A-2 Exporting a Version-controlled Project

This appendix describes the procedure to export a version-controlled project.
By exporting a version-controlled project, you can handle it as a project that is not version controlled.

1 Open the target project in the **Version Control Projects** window.

2 Select **Export** from the **File** menu.
The **Export file** dialog box is displayed.

3 Specify the storage location and file name, select the file type, and then click the **Save** button.
The project is exported to a file.

Additional Information

- You can import and then open the exported file as a project that is not version-controlled in the Sysmac Studio. Refer to *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for the import procedure.
- Refer to *Sysmac Studio Version 1 Operation Manual (Cat. No. W504)* for information about the file extensions of the exported file.

I

Index

Index

OMRON Corporation **Industrial Automation Company**
Kyoto, JAPAN
Contact: www.ia.omron.com

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

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

OMRON ASIA PACIFIC PTE. LTD.
438B Alexandra Road, #08-01/02 Alexandra
Technopark, Singapore 119968
Tel: (65) 6835-3011/Fax: (65) 6835-2711

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

Authorized Distributor:

Cat. No. W589-E1-04 0822