OMRON

Sysmac Library

User's Manual
for Packaging Machine Library

SYSMAC-XR012

always in control

W572-E1-04

NOTE

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. 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

Copyrights

Introduction

Thank you for purchasing an NJ/NX-series CPU Unit or an NY-series Industrial PC.

This manual contains information that is necessary to use the function blocks in the Packaging Machine Library. (“Function block” is sometimes abbreviated as “FB”.) Please read this manual and make sure you understand the functionality and performance of the NJ/NX-series CPU Unit before you attempt to use it in a control system.

This manual provides function block specifications. It does not describe application restrictions or combination restrictions for Controllers, Units, and components.

Refer to the user’s manuals for all of the products in the application before you use any of the products.

Keep this manual in a safe place where it will be available for reference during operation.

Features of the Library

The packaging machine library is used to control various packaging machines with NJ/NX/NY-series Controller. The packaging machine library enables accurate and high-speed packaging processing, and reduction of programming man-hour as well as man-hour to start up packaging system.

The Packaging Machine Library consists of three types of FBs according to the application.

- Rotary knife

  FB that controls rotary knives which continuously cut sheets or end seals for pillow packaging machine.
- Winder, Unwinder

  FB that controls tension of materials for continuous packaging when the materials are wound or unwound.
- PackML Support

  FB or FUN that comforms to PackML standard (Packaging Machine Language), which was defined by OMAC (Organization for Machine Automation and Control), standardizing device modes and interface with other devices for packaging machine.

Each FB consists of following FB and FUN.

FB group type	FB/FUN name	Name
Rotary Knife	RotaryKnife	Rotary Knife
Winder, Unwinder	WinderVelCtrl_Servo	Velocity Control Winder (for Servo)
	WinderVelCtrl_Inverter	Velocity Control Winder (for Inverter)
PackML Support	PackMLModeStateMachine	PackML Mode/State Machine
	PackMLModeStateTimer	Dwell Time Measure
	PMLCtrlCmd_**	Transition Command Display
	PMLState_Is**	State Output
	PMLTransitionCmd_ResetAll	Transition Command All Reset
	PMLTransitionCmd_ResetAllCmd SetAllSC	Transition Command Reset State Set
	PMLTransitionCmd_Summarize	Transition Command Summarize
	PMLTransitionCmd_Summarize PackTagCtrlCmd	Pack Tag Transition Command
	Alarm	Alarm
	Alarm2	Alarm2
	AlarmStatus_Update	EM Alarm Status Update
	AlarmStatus_Update2	EM Alarm Status Update2
	AlarmSummation_Add	UN Alarm Status Add
	AlarmSummation_Add2	UN Alarm Status Add2
	AlarmSummation_SortFilter	Alarm Sort and Filter
	AlarmSummation_SortFilter2	Alarm Sort and Filter2
	DT_TO_PackTagDINTarray	DATE_AND_TIME Type Pack Tag Array Conversion

Intended Audience

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

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

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

Applicable Products

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

Manual Structure

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 CPU Units and Industrial PCs with different unit versions and for different versions of the Sysmac Studio are given.

Note References are provided to more detailed or related information.

CONTENTS

Terms and Conditions Agreement

Warranty, Limitations of Liability

Warranties

- **Exclusive Warranty**

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

- **Limitations**

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

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

- **Buyer Remedy**

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

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

Limitation on Liability; Etc

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

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

Application Considerations

Suitability of Use

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

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

Programmable Products

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

Disclaimers

Performance Data

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

Change in Specifications

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

Errors and Omissions

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

Safety Precautions

Definition of Precautionary Information

The following notation is used in this user's manual to provide precautions required to ensure safe usage of an NJ/NX-series Controller and an NY-series Industrial PC.

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

Warings

⚠ WARNING

Emergency stop circuits, interlock circuits, limit circuits, and similar safety measures must be provided in external control circuits. If not, abnormal operation may be caused and it could result in serious accident.	❗
Using this function block (FB) in a device, confirm that the program and FB operates properly. Design a program so that safety measures such as fail-safe circuits are implemented outside of the FB.	❗

Cautions

⚠ Caution

Read all related manuals carefully before you use this library.	⚠
Check the user program, data, and parameter settings for proper execution before you use them for actual operation.	⚠
The Sysmac Library and manuals are assumed to be used by personnel that is given in Intended Audience in this manual. Otherwise, do not use them.	⚠
The sample programming shows only the portion of a program that uses the function or function block from the library.	⚠
When using a sample program in actual devices, also use safety circuits for the device operation, device interlocks, I/O with other devices, and user programs regarding control procedures.	⚠
Understand the contents of sample programming before you use the sample programming and create the user program.	⚠
Create a user program so that the actual device operates as intended.	⚠
If the rotation direction of the motor is not set correctly depending on the device, unexpected operation may occur. Make sure to set the rotation direction of the servo drive correctly depending on the device.	⚠
If FB is used by setting input parameters which are different from the description specified in this manual, it could result in injury or property damage. Make sure to set the input parameters for this FB correctly depending on the device.	⚠
If multiple applications are started in *SlaveAxis* by other motion FB when this FB is executed, the execution of this FB ends. In the case, command to the *SlaveAxis* is changed suddenly and excessive load may be given to the device. When this FB is executed, make sure not to execute other motion commands to *SlaveAxis*.	⚠
Do not delete the instance from the program with online editing during an execution of this FB. The SlaveAxis may move unexpectedly.	⚠
IF *Enable* = TRUE -> FALSE is executed when this FB is executed and *SlaveAxis* is moving, the *SlaveAxis* stops immediately and property damage may occur. After the *SlaveAxis* stops, execute *Enable* = TRUE -> FALSE.	⚠
Before adjusting device by installing this FB to the device, make sure to check the surrounding situations.	⚠
When this FB is installed in the device to confirm operation of the FB, make sure to perform sufficient test run such as moving *MasterAxis* slowly, to make adjustment. If not, it may lead to injury or property damage.	⚠
For Rotary Knife When using this FB, set the Counter Mode of the master and slave Axis parameters to Rotary Mode.	⚠

For Rotary Knife If the Position Count Setting for *MasterAxis* and *SlaveAxis* is not done correctly depending on the device, unintended operation may occur. Make sure to configure the setting correctly depending on the device.	⚠
For Rotary Knife When resuming the operation, use *MasterCfg* and *SlaveCfg*, *CuttingCfg* with which you saved continuous operation data. At the time of continuous operation, if each setting has been changed, there is a risk of unintended behavior.	⚠
For Winder/Unwinder When using this function block, set the count mode of the Winder/Unwinder axis to Rotary Mode.	⚠
For Winder/Unwinder If the Position Count Setting for Winding Axis and Unwinding Axis is not done correctly depending on the device, unintended operation may occur. Make sure to configure the setting correctly depending on the device.	⚠

Precautions for Safe Use

Operation

- Execute this FB in the primary periodic task of the Controller. When it is executed in another type of task, SlaveAxis may operate unexpectedly.
- For Rotary Knife

  If the home position of the *SlaveAxis* and the blade position are set in a position different from the position described in this manual, the blade may move to an unexpected position. The setting of the position must be the same as the one described in the manual.
- The number of array elements for *Alarm[]* effects the non-retained variable memory usage and the task execution time. Check the number of array elements for proper execution before you use it for actual operation.

Precautions for Correct Use

Operation

- Specify the input parameter values within the valid range.
- For Rotary Knife
  Do not change the set value of *CuttingCnfig.OffsetFromMark* more than CutLengh, too big change value may cause error.

Related Manuals

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

Manual name	Cat. No.	Model numbers	Application	Description
NX-series CPU Unit Hardware ManualUser's	W535	NX701-□□□□	Learning the basic specifications of the NX-series NX701 CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided	An introduction to the entire NX701 CPU Unit system is provided along with the following information on the CPU Unit. Features and system configuration Overview Part names and functions General specifications Installation and wiring Maintenance and inspection
NX-series NX102 CPU Unit Hardware User's Manual	W593	NX102-□□□□	Learning the basic specifications of the NX102 CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NX102 system is provided along with the following information on the CPU Unit. Features and system configuration Overview Part names and functions General specifications Installation and wiring Maintenance and Inspection
NX-series NX1P2 CPU Unit Hardware User's Manual	W578	NX1P2-□□□□	Learning the basic specifications of the NX-series NX1P2 CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided	An introduction to the entire NX1P2 CPU Unit system is provided along with the following information on the CPU Unit. Features and system configuration Overview Part names and functions General specifications Installation and wiring Maintenance and Inspection
NJ-series CPU Unit Hardware User's Manual	W500	NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning the basic specifications of the NJ-series CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided	An introduction to the entire NJ-series system is provided along with the following information on the CPU Unit. Features and system configuration Overview Part names and functions General specifications Installation and wiring Maintenance and inspection
NY-series IPC Machine Controller Industrial Panel PC Hardware User's Manual	W557	NY532-□□□□	Learning the basic specifications of the NY-series Industrial Panel PCs, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided	An introduction to the entire NY-series system is provided along with the following information on the Industrial Panel PC. Features and system configuration Introduction Part names and functions General specifications Installation and wiring Maintenance and inspection

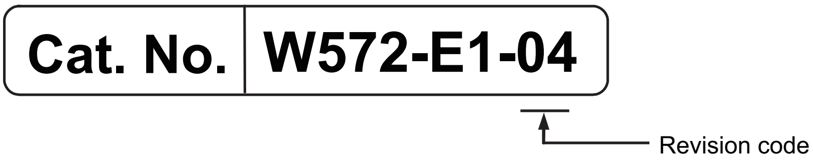
Manual name	Cat. No.	Model numbers	Application	Description
NY-series IPC Machine Controller Industrial Box PC Hardware User's Manual	W556	NY512-□□□□	Learning the basic specifications of the NY-series Industrial Box PCs, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided	An introduction to the entire NY-series system is provided along with the following information on the Industrial Box PC. Features and system configuration Introduction Part names and functions General specifications Installation and wiring Maintenance and inspection
NJ/NX-series CPU Unit Software User's Manual	W501	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning how to program and set up an NJ/NX-series CPU Unit. Mainly software information is provided	The following information is provided on a Controller built with an NJ/NX-series CPU Unit. CPU Unit operation CPU Unit features Initial settings Programming based on IEC 61131-3 language specifications
NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Software User's Manual	W558	NY532-□□□□ NY512-□□□□	Learning how to program and set up the Controller functions of an NY-series Industrial PC	The following information is provided on NY-series Machine Automation Control Software. Controller operation Controller features Controller settings Programming based on IEC 61131-3 language specifications
NJ/NX-series Instructions Reference Manual	W502	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning detailed specifications on the basic instructions of an NJ/NX-series CPU Unit	The instructions in the instruction set (IEC 61131-3 specifications) are described.
NY-series Instructions Reference Manual	W560	NY532-□□□□ NY512-□□□□	Learning detailed specifications on the basic instructions of an NY-series Industrial PC	The instructions in the instruction set (IEC 61131-3 specifications) are described.
NJ/NX-series CPU Unit Motion Control User's Manual	W507	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning about motion control settings and programming concepts of an NJ/NX-series CPU Unit.	The settings and operation of the CPU Unit and programming concepts for motion control are described.
NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Motion Control User's Manual	W559	NY532-□□□□ NY512-□□□□	Learning about motion control settings and programming concepts of an NY-series Industrial PC.	The settings and operation of the Controller and programming concepts for motion control are described.
NJ/NX-series Motion Control Instructions Reference Manual	W508	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning about the specifications of the motion control instructions of an NJ/NX-series CPU Unit.	The motion control instructions are described.
NY-series Motion Control Instructions Reference Manual	W561	NY532-□□□□ NY512-□□□□	Learning about the specifications of the motion control instructions of an NY-series Industrial PC.	The motion control instructions are described.

Manual name	Cat. No.	Model numbers	Application	Description
NJ/NY-series NC Integrated Controller User's Manual	O030	NJ501-5300 NY532-5400	Performing numerical control with NJ/NY-series Controllers.	Describes the functionality to perform the numerical control. Use this manual together with the *NJ/NY-series G code Instructions Reference Manual* (Cat. No. O031) when programming.
NJ/NY-series G code Instructions Reference Manual	O031	NJ501-5300 NY532-5400	Learning about the specifications of the G code/M code instructions.	The G code/M code instructions are described. Use this manual together with the *NJ/NY-series NC Integrated Controller User's Manual* (Cat. No. O030) when programming.
Sysmac Studio Version 1 Operation Manual	W504	SYSMAC -SE2□□□	Learning about the operating procedures and functions of the Sysmac Studio.	Describes the operating procedures of the Sysmac Studio.
CNC Operator Operation Manual	O032	SYSMAC -RTNC0□□□D	Learning an introduction of the CNC Operator and how to use it.	An introduction of the CNC Operator, installation procedures, basic operations, connection operations, and operating procedures for main functions are described.
NX-series EtherCAT® Coupler Unit User's Manual	W519	NX-ECC□□□	Learning how to use an NXseries EtherCAT Coupler Unit and EtherCAT Slave Terminals.	The system and configuration of EtherCAT Slave Terminals, which consist of an NX-series EtherCAT Coupler Unit and NX Units, are described along with the hardware, setup, and functions of the EtherCAT Coupler Unit that are required to configure, control, and monitor NX Units through EtherCAT.
NX-series NX Units User's Manuals	W521	NX-ID□□□□ NX-IA□□□□ NX-OC□□□□ NX-OD□□□□	Learning how to use NX Units.	Describes the hardware, setup methods, and functions of the NX Units. Manuals are available for the following Units. Digital I/O Units, Analog I/O Units, System Units, and Position Interface Units.
	W522	NX-AD□□□□ NX-DA□□□□ NX-TS□□□□		
	W523	NX-PD1□□□ NX-PF0□□□ NX-PC0□□□ NX-TBX□□		
	W524	NX-EC0□□□ NX-ECS□□□ NX-PG0□□□		
GX-series EtherCAT Slave Units User's Manual	W488	GX-ID□□□□ GX-OD□□□□ GX-OC□□□□ GX-MD□□□□ GX-AD□□□□ GX-DA□□□□ GX-EC□□□□ XWT-ID□□ XWT-OD□□	Learning how to use the EtherCAT remote I/O terminals.	Describes the hardware, setup methods and functions of the EtherCAT remote I/O terminals.
AC Servomotors/Servo Drivers G5-series with Built-in EtherCAT® Communications User's Manual	I573	R88M-K□ R88D-KN□-ECT-R	Learning how to use the AC Servomotors/Servo Drivers with built-in EtherCAT Communications.	Describes the hardware, setup methods and functions of the AC Servomotors/Servo Drives with built-in EtherCAT Communications. The linear motor type model and the model dedicated for position controls are available in G5-series.
	I576	R88M-K□ R88D-KN□-ECT		
	I577	R88L-EC-□ R88D-KN□-ECT-L		
AC Servomotors/Servo Drivers 1S-series with Built-in EtherCAT® Communications User's Manual	I586	R88M-1□ R88D-1SN□-ECT	Learning how to use the Servomotors/Servo Drives with built-in EtherCAT Communications.	Describes the hardware, setup methods and functions of the Servomotors/Servo Drives with built-in EtherCAT Communications.
Inverter RX Series LCD Digital Operator User's Manual	I579	3G3AX-OP05	Learning how to use the LCD Digital Operator.	Describes the setup methods and functions of the LCD Digital Operator.
Inverter MX2/RX Series EtherCAT® Communication Unit User's Manual	I574	3G3AX-MX2-ECT 3G3AX-RX-ECT	Learning how to use the Inverter EtherCAT Communication Unit.	Describes the hardware, setup methods and functions of the Inverter EtherCAT Communication Unit.

Manual name	Cat. No.	Model numbers	Application	Description
High-function General-purpose Inverter RX Series Type V1 User’s Manual	I578	3G3RX-□□□□□-V1	Learning how to use the High-function General-purpose Inverter.	Describes the hardware, setup methods and functions of the High-function General-purpose Inverter.
3G3AX-PG Encoder Feedback Board User’s Manual	I564	3G3AX-PG□□	Learning how to use the Encoder Feedback Board.	Describes the hardware, setup methods and functions of the Encoder Feedback Board.

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 2016	Original production
02	November 2016	Corrected mistakes.
03	January 2019	• Added compatible models. • Added FBs/FUNs that support variable-length arrays.
04	January 2021	Corrected mistakes.

Procedure to Use Sysmac Libraries

Procedure to Use Sysmac Libraries Installed Using the Installer

This section describes the procedure to use Sysmac Libraries that you installed using the installer.

There are two ways to use libraries.

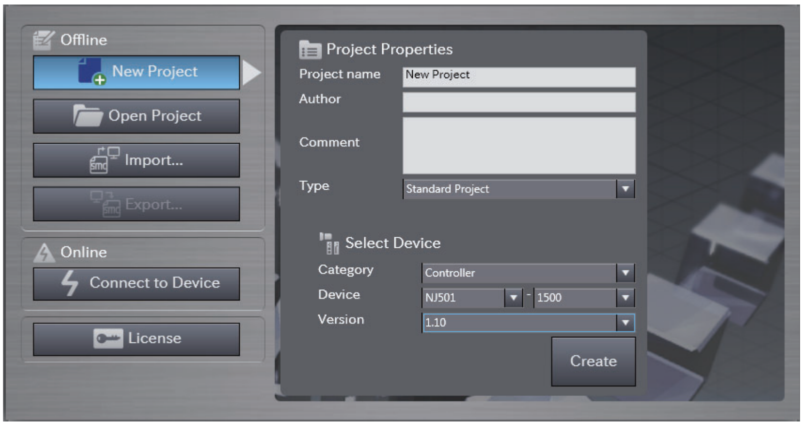
- Using newly installed Sysmac Libraries
- Using upgraded Sysmac Libraries

Version Information

To use Sysmac Libraries, you need the Sysmac Studio version 1.14 or higher.

Using Newly Installed Libraries

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



Project Properties

Project name: New Project

Author:

Comment:

Type: Standard Project

Select Device

Category: Controller

Device: NJ501 - 1500

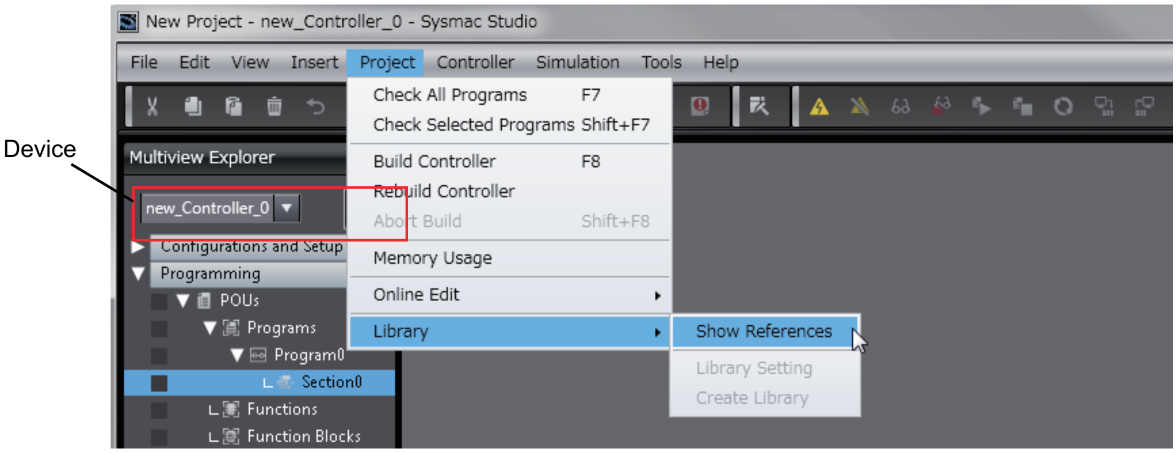
Version: 1.10

Precautions for Correct Use

If you create a new project, be sure to configure the settings as follows to enable the use of Sysmac Libraries. If you do not configure the following settings, you cannot proceed to the step 2 and later steps.

- Set the project type to Standard Project or Library Project.
- Set the device category to Controller.
- Set the device version to 1.01 or later.

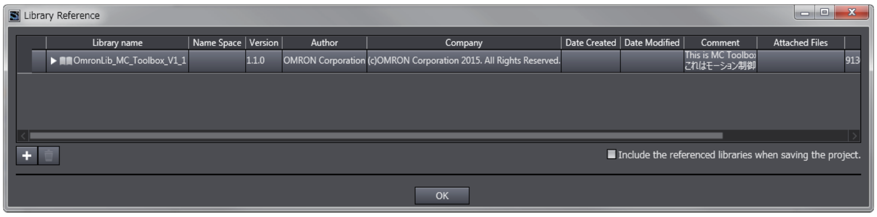
2 Select **Project** – **Library** – **Show References**.



Precautions for Correct Use

If you have more than one registered device in the project, make sure that the device selected currently is an NJ/NX-series CPU Unit or an NY-series Industrial PC. If you do not select an NJ/NX-series CPU Unit or an NY-series Industrial PC as the device, Library References does not appear in the above menu. When the device selected currently is an NJ/NX-series CPU Unit or an NY-series Industrial PC, the device icon ▮▮ is displayed in the Multiview Explorer.

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


Library Reference

Library name	Name Space	Version	Author	Company	Date Created	Date Modified	Comment	Attached Files	
▶ OmronLib_MC_Toolbox_V1_1		1.1.0	OMRON Corporation	(c)OMRON Corporation 2015. All Rights Reserved.			This is MC Toolbo これはモーション制御		913

□ Include the referenced libraries when saving the project.

OK

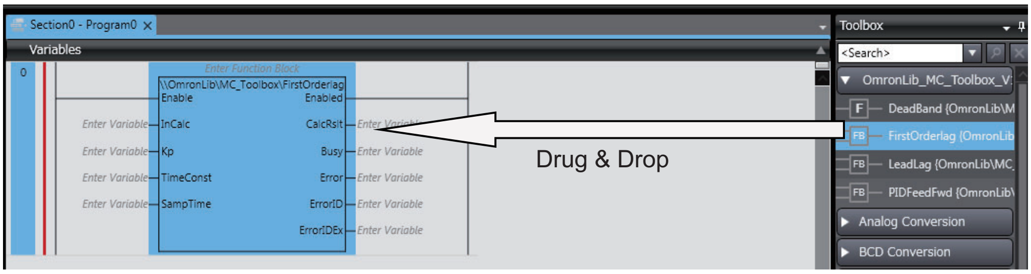
The Sysmac Library file is read into the project.

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

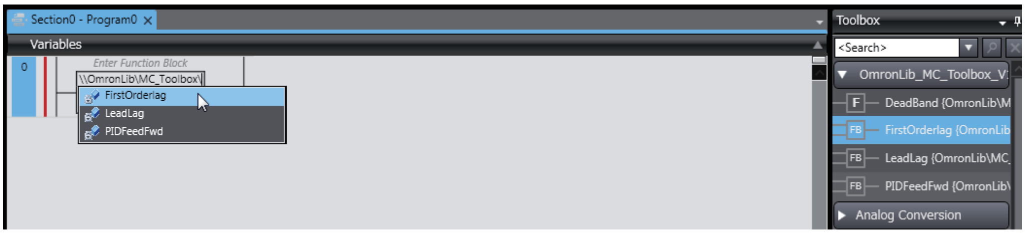
For the procedure for adding and setting libraries in the above screen, refer to the *Sysmac Studio Version 1 Operation Manual* (Cat. No. W504).

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

- Select the desired function block or function in the Toolbox and drag and drop it onto the programming editor.



- Right-click the programming editor, select **Insert Function Block** in the menu, and enter the fully qualified name (\\name of namespace\name of function block).



Precautions for Correct Use

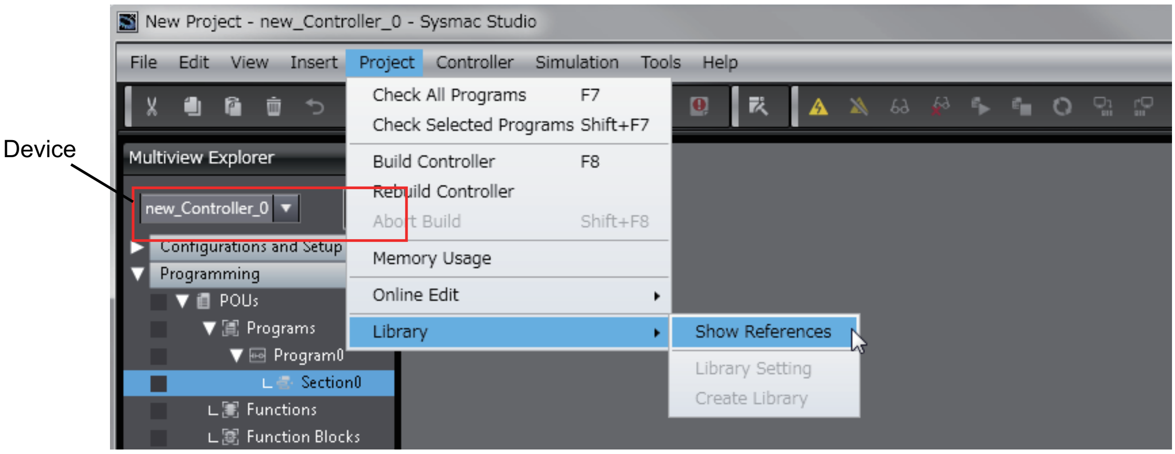
After you upgrade the Sysmac Studio, check all programs and make sure that there is no error of the program check results on the Build Tab Page.

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

Using Upgraded Libraries

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

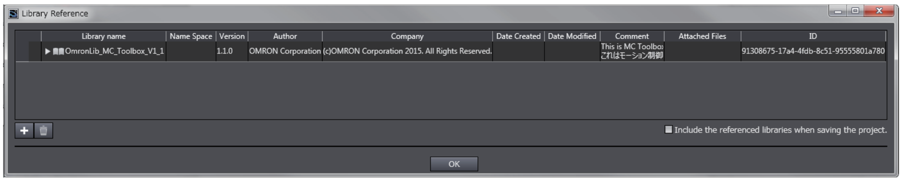
2 Select **Project** – **Library** – **Show References**.



Precautions for Correct Use

If you have more than one registered device in the project, make sure that the device selected currently is an NJ/NX-series CPU Unit or an NY-series Industrial PC. Otherwise, Library References does not appear in the above menu. When the device selected currently is an NJ/NX-series CPU Unit or an NY-series Industrial PC, the device icon ▮ is displayed in the Multiview Explorer.

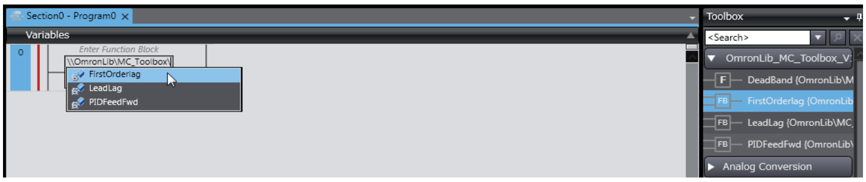
3 Select an old-version Sysmac Library and click the **Delete Reference** Button.


Library Reference

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

☐ Include the referenced libraries when saving the project.

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



Procedure to Use Sysmac Libraries Uploaded from a CPU Unit or an Industrial PC

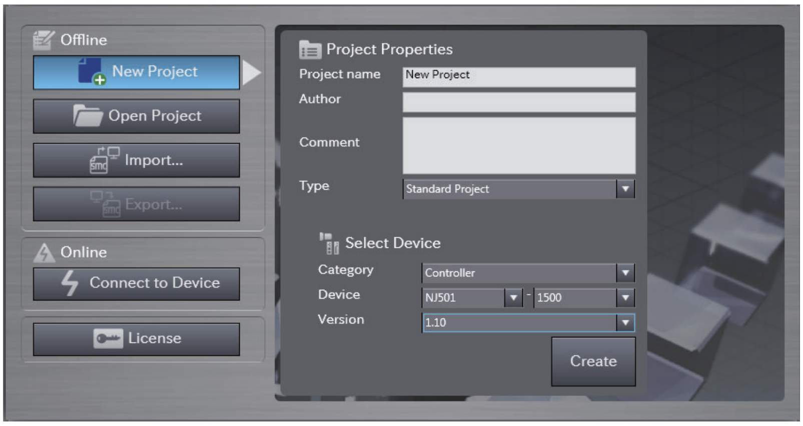
You can use Sysmac Libraries uploaded from a CPU Unit or an Industrial PC to your computer if they are not installed.

The procedure to use uploaded Sysmac Libraries from a CPU Unit or an Industrial PC is as follows.

Version Information

To use Sysmac Libraries, you need the Sysmac Studio version 1.14 or higher.

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



Project Properties
Project name: New Project
Author:
Comment:
Type: Standard Project

Select Device
Category: Controller
Device: NJ501 - 1500
Version: 1.10

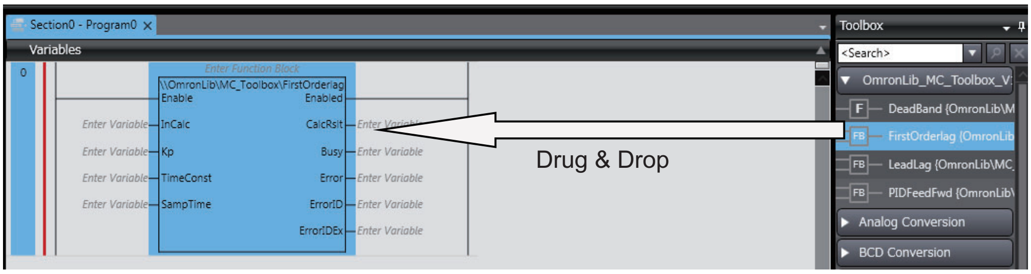
2 Connect the computer to the CPU Unit or the Industrial PC and place it online.

3 Upload POUs in which any Sysmac Library is used to the computer.

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

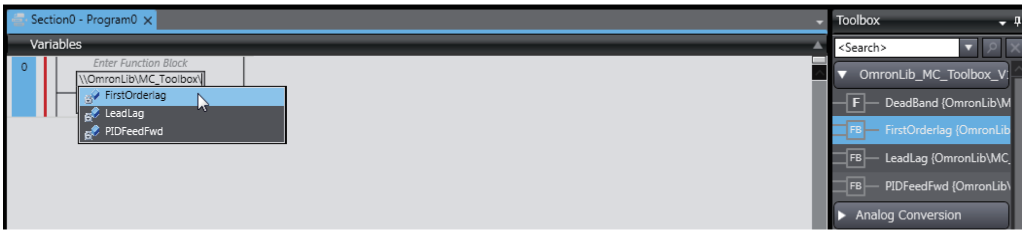
4 Insert the Sysmac Library's function blocks and functions into the circuit using one of the following two methods.

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



Drug & Drop

- Right-click the programming editor, select **Insert Function Block** in the menu, and enter the fully qualified name (\\name of namespace\name of function block).



Precautions for Correct Use

- The Sysmac Studio installs library files of the uploaded Sysmac Stutio to the specified folder on the computer if they are not present. However, the Sysmac Studio does not install library files to the specified folder on the computer if they are present.

  The specified folder here means the folder in which library files are installed by the installer.
- Note that uploading Sysmac Libraries from a CPU Unit or an Industrial PC does not install the manual and help files for the Sysmac Libraries, unlike the case where you install then using the installer. Please install the manual and help files using the installer if you need them.

Common Specifications of Function Blocks

Common Variables

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

Definition of Input Variables and Output Variables

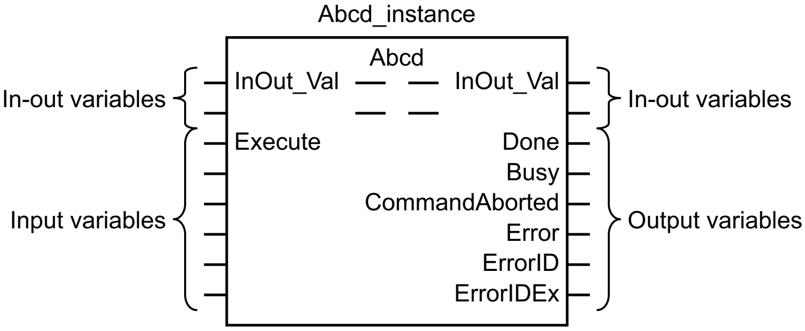
Common input variables and output variables used in functions and function blocks are as follows.

Variable	I/O	Data type	Function/function block type to use			Meaning	Definition
			Function block		Function		
			Execute-type	Enable-type			
EN	Input	BOOL			OK	Execute	The processing is executed while the variable is TRUE.
Execute			OK			Execute	The processing is executed when the variable changes to TRUE.
Enable				OK		Run	The processing is executed while the variable is TRUE.
Abort		BOOL	OK			Abort	The processing is aborted. You can select the aborting method.

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

Execute-type Function Blocks

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

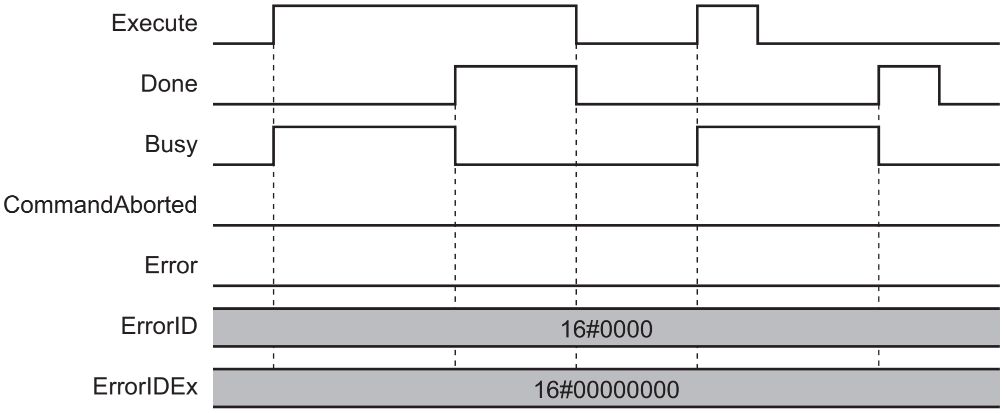


- If *Execute* is TRUE and *Done*, *CommandAborted*, or *Error* changes to TRUE, *Done*, *CommandAborted*, and *Error* changes to FALSE when *Execute* is changed to FALSE.
- If *Execute* is FALSE and *Done*, *CommandAborted*, or *Error* changes to TRUE, *Done*,*CommandAborted*, and *Error* changes to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

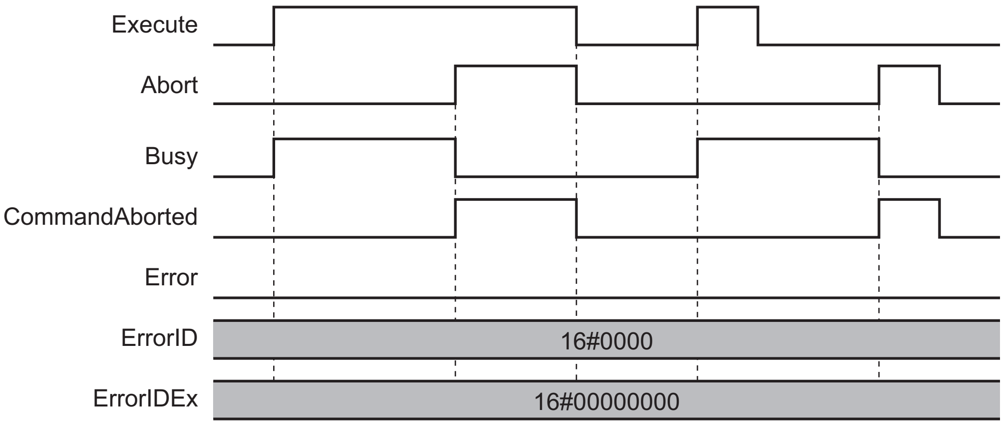
Timing Charts

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

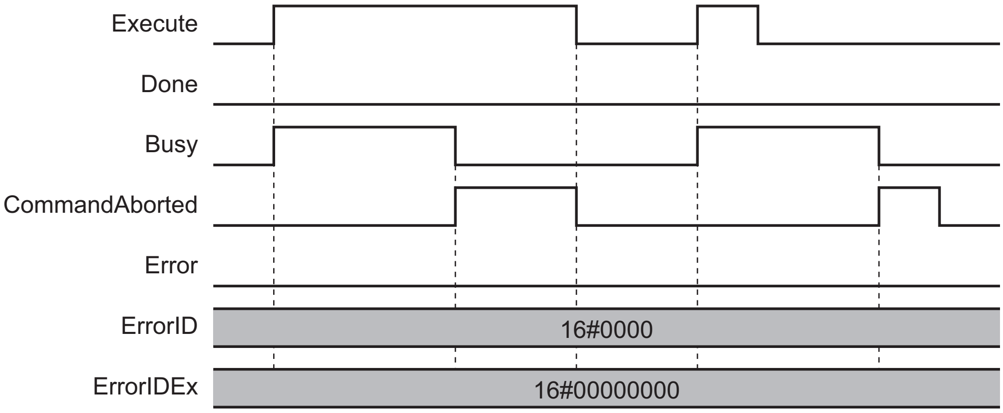
● Normal End



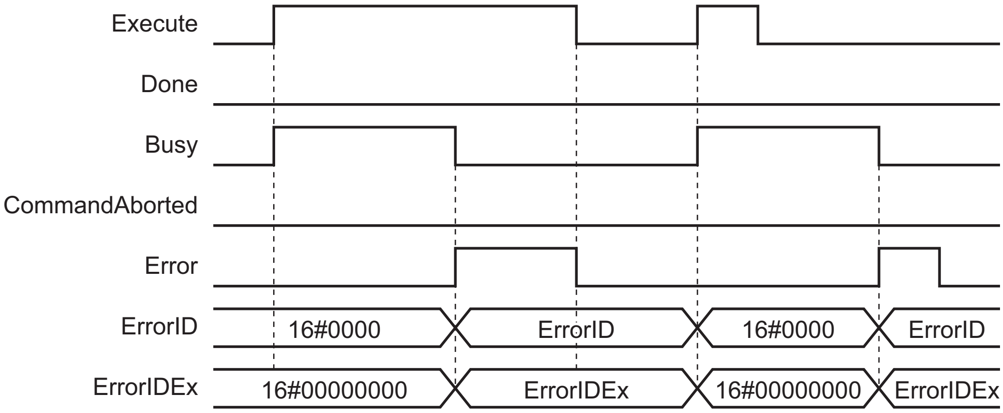
- **Canceled Execution**



- **Aborted Execution**

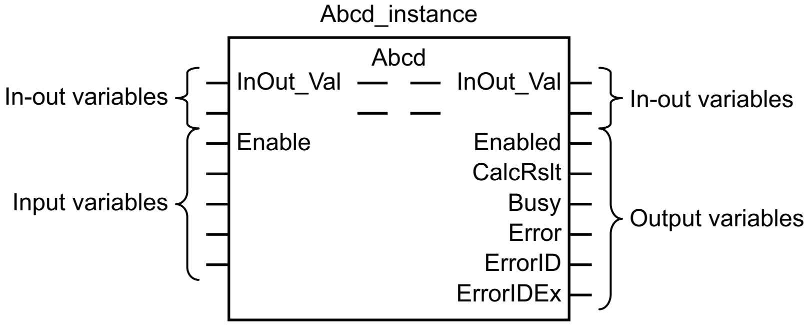


- **Errors**



Enable-type Function Blocks

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

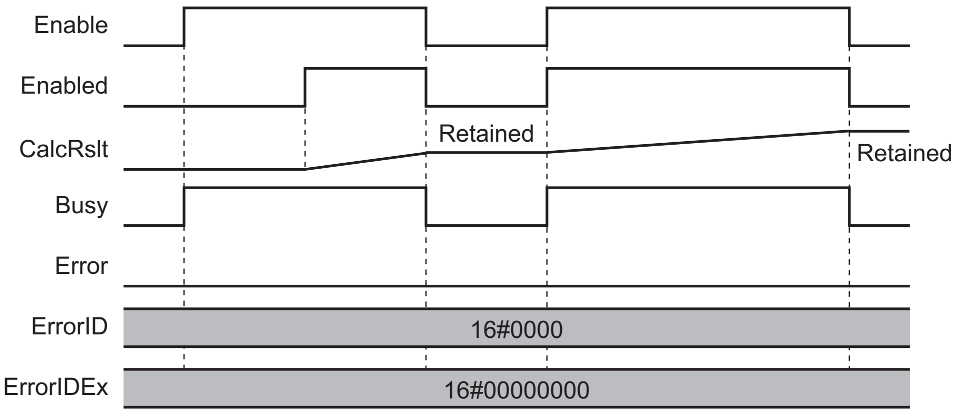


- If an error occurs, the relevant error code and expansion error code are set in *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code). The error codes are retained even after *Error* changes to FALSE, but *ErrorID* is set to 16#0000 and *ErrorIDEx* is set to 16#0000 0000 when *Enable* changes to TRUE.
- For function blocks that calculate the control amount for motion control, temperature control, etc., *Enabled* is FALSE when the value of *CalcRslt* (Calculation Result) is incorrect. In such a case, do not use *CalcRslt*. In addition, after the function block ends normally or after an error occurs, the value of *CalcRslt* is retained until *Enable* changes to TRUE. The control amount will be calculated based on the retained *CalcRslt* value, if it is the same instance of the function block that changed *Enable* to TRUE. If it is a different instance of the function block, the control amount will be calculated based on the initial value.

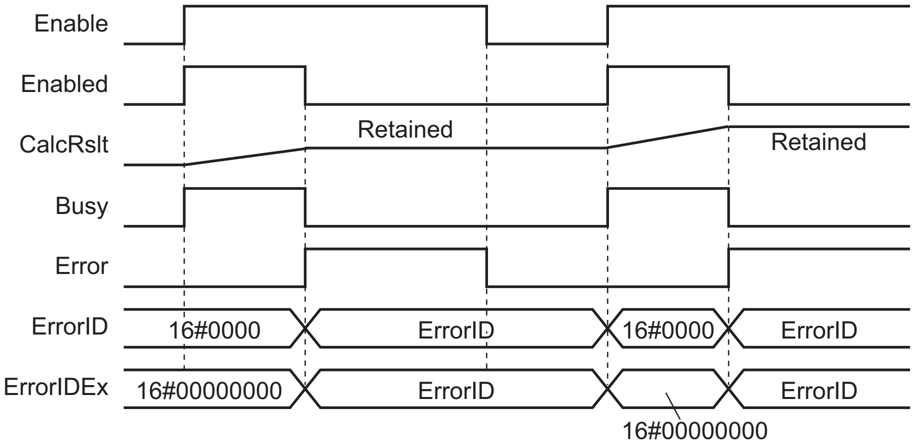
Timing Charts

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

● Normal End



● Errors


Enable
Enabled
CalcRslt
Busy
Error
ErrorID
ErrorIDEx
Retained
Retained
16#0000
ErrorID
16#0000
ErrorID
16#00000000
ErrorID
ErrorID
16#00000000

Precautions

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

Nesting

You can nest calls to this function block for up to four levels.

For details on nesting, refer to the software user’s manual.

Instruction Options

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

Re-execution of Function Blocks

Execute-type function blocks cannot be re-executed by the same instance.

If you do so, the output value will be the initial value.

For details on re-execution, refer to the motion control user’s manual.

RotaryKnife

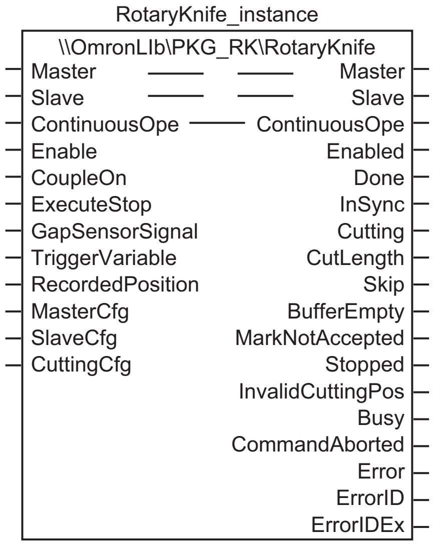
Function block name	Name	Page
RotaryKnife	RotaryKnife	P.38

RotaryKnife

RotaryKnife is a function block (FB) that controls the rotary knife.

This FB is a software module consisting of the functions necessary for the following applications and was developed for reducing man-hours required for programming.

- Cutting web-like products such as sheets
- End sealing and cutting operation of the pillow packaging machine
- Labeling
- Applications that follow the product-flow and process the products

Function block name	Name	FB/ FUN	Graphic expression	ST expression
RotaryKnife	RotaryKnife	FB		RotaryKnife_instance(Master, Slave, ContinuousOpe, Enable, CoupleOn, ExecuteStop, GapSensorSignal, TriggerVariable, RecordedPosition, MasterCfg, SlaveCfg, CuttingCfg, Enabled, Done, InSync, Cutting, CutLength, Skip, BufferEmpty, MarkNotAccepted, Stopped, InvalidCuttingPos, Busy, CommandAborted, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PKG_RK_Vx_x.slr*1
Namespace	OmronLib\PKG_RK
Function block and function number	00064
Publish/Do not publish source code	Not published.
Function block and function version	1.00

*1. Vx_x indicates version.

Precautions for Correct Use

- The nesting depth of this FB limits fifth or less level in an user program.
  Sixth or more depth causes error at the program building.
- Regarding the setting of NJ101-1000, primary periodic task is recommended more than 2ms.
  Setting 1ms or less for primary periodic task may cause Task Period Exceeded Error.

Compatible Models

Item	Name	Model numbers	Version
Device	AC Servo Driver (G5 Series)	R88M-KN□-ECT	Version 2.1 or later
	AC Servo Motor (G5 Series)	R88M-K□	-
	AC Servo Drive (1S series)	R88D-1SN□□□-ECT	Version 1.0 or later
	AC Servo Motor (1S series)	R88M-1□	-
	Multi-function Compact Inverter MX2 Series V1 Type	3G3MX2-V1	-
	NX-series Digital Input Unit	NX-ID□□□□	Revision 1.0 or later
	NX-series Encoder Input Unit*1	NX-EC□□□□	Revision 1.0 or later
	Mark Sensor (recommended)	E3NX (Amplifier) E32 (Fiver Unit)	-
	Gap Sensor (recommended)	E3NX (Amplifier) E32 (Fiber Unit)	

*1. There is no restriction on the models of encoders used for data input to the NX-series Encoder Input Unit. Select an encoder according to the specifications of the NX-series Encoder Input Unit.

Hardware and Software Configuration

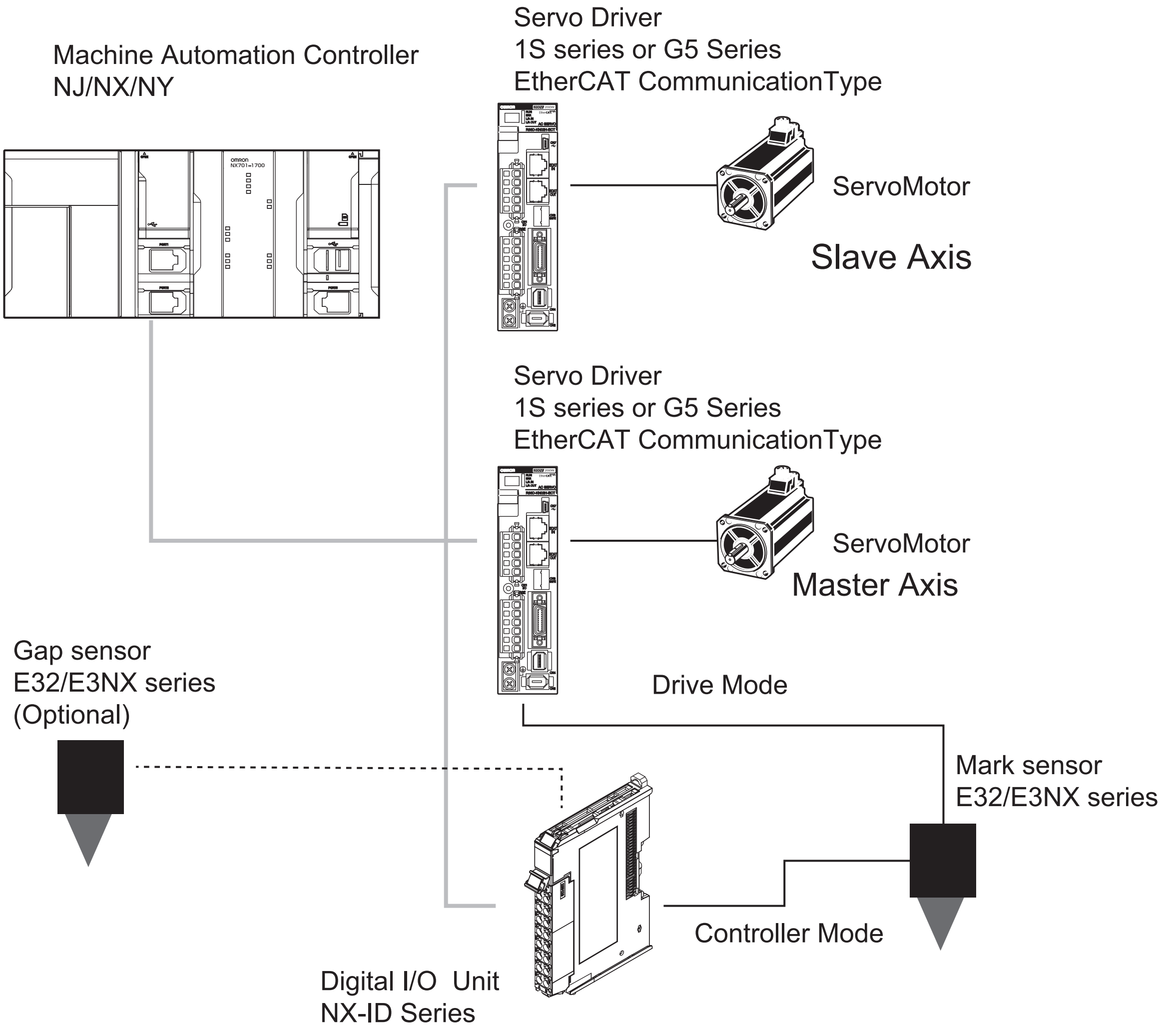
This section describes the hardware and software configuration when this FB is used. As examples of the hardware configuration, configuration by servo motors, and configuration by frequency inverter and servo motors are shown as follows.

Case 1: By Servo Motors

This section describes hardware configuration, device type, and for software configuration.

● Hardware Configuration

The following figure shows the hardware configuration when MasterAxis and SlaveAxis are operated by servo motor.



- Virtual servo axes can be set as the axis types of the MasterAxis and the SlaveAxis. In this case, a Servo Drive is unnecessary.
- The actual current value or command value can be selected as the value accessed by the MasterAxis.
- One or more SlaveAxes can also be used in ExternalMarkToMark mode. For details, refer to *External Mark To Mark* on page 66.
- Connect the mark sensor to ServoMoter or Remote I/O according to the setting of Mode: Drive Mode/Controller Mode of the *_sTRIGGER_REF* of structure variable.
- Refer to *Setting TriggerInput* on page 90 for *_sTRIGGER_REF.*

- Gap sensor is connected only when using the NoGapNoSeal function.

● Software Configuration

When using this FB, configure the MasterAxis and the SlaveAxis as shown below.
The Axis settings which are not described in the table below, should match the machine specification, referring motion control user’s manual.

Master Axis Configuration

Classification	Parameter name	Set value
Axis Basic Settings	Axis Type	Servo axis[*1]
Position Count Settings	Count Mode	1: Rotary Mode
	Modulo Maximum Position Setting Value	Positive value[*2]
	Modulo Minimum Position Setting Value	0

*1. When using a virtual servo axis, set the axis type to a virtual servo axis.

*2. Set the value according to the device where this FB is used.

Slave Axis Configuration

Classification	Parameter name	Set value
Axis Basic Settings	Axis Type	Servo axis[*1]
Position Count Settings	Count Mode	1: Rotary Mode
	Modulo Maximum Position Setting Value	Circumference of the cutter
	Modulo Minimum Position Setting Value	0
Operation Setting	In-position Check Time	0

*1. When using a virtual servo axis, set the axis type to a virtual servo axis.

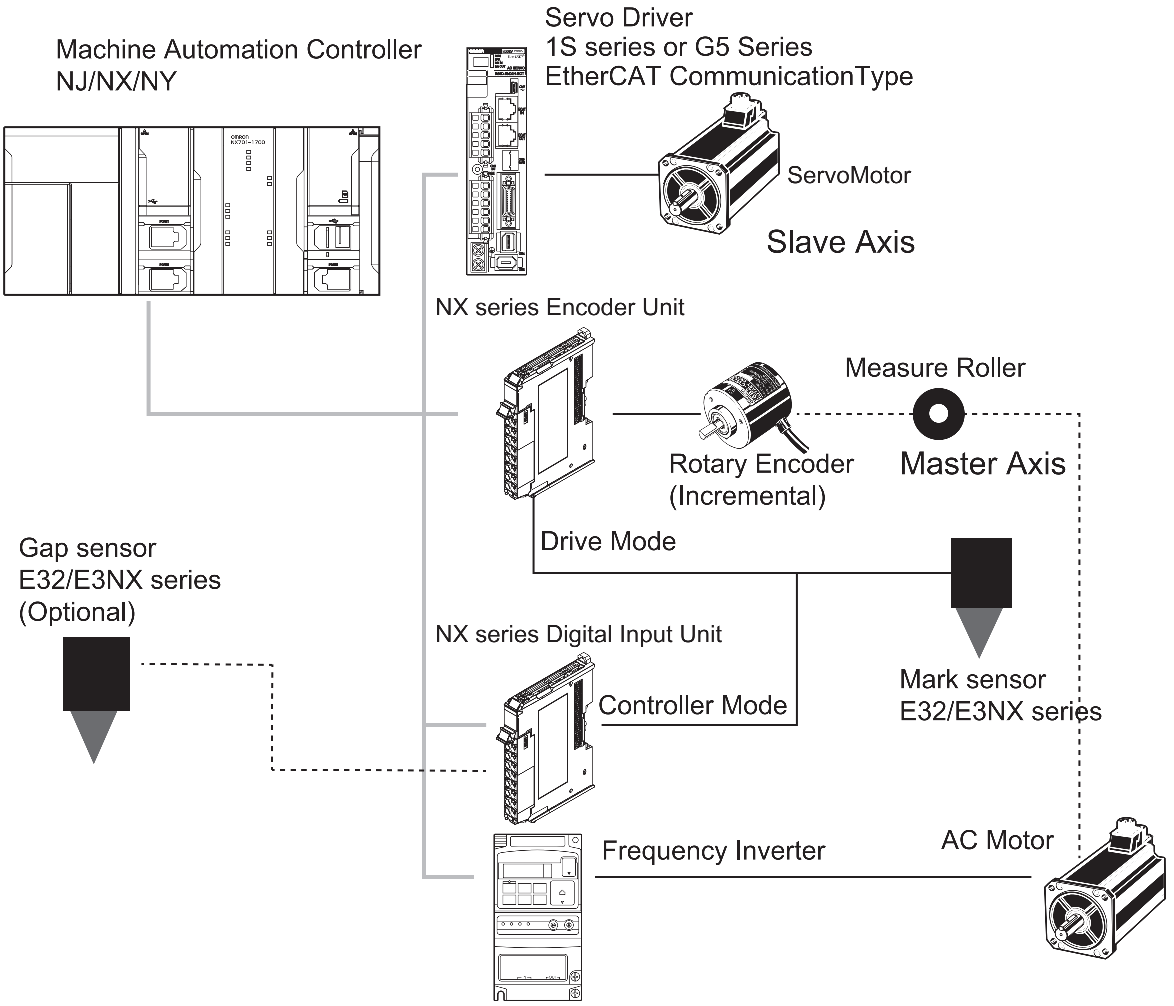
⚠ WARNING	
Do not use this FB in a system that consists of devices and versions not specified in this document. Doing so may cause the system to operate in unintended ways.	❗
When using this FB, set the count mode of the MasterAxis and SlaveAxis to Rotary Mode. The SlaveAxis may operate in unintended ways if you set the MasterAxis and SlaveAxis settings that are different from those specified in this document.	⚠

Case 2: By Frequency Inverter and Servo Motor

The following figure shows the hardware configuration, supported type, and software configuration in the case 2.

● Hardware Configuration

The master axis is driven by the Frequency Inverter and the slave axis is driven by the servo motor.



- Virtual servo axes can be set as the axis types of the SlaveAxis. In this case, a Servo Drive is unnecessary.
- The actual current value or command value can be selected as the value accessed by the MasterAxis.
- One or more SlaveAxes can also be used in ExternalMarkToMark mode. For details, refer to *External Mark To Mark* on page 66.
- Connect the mark sensor to NX-series Encoder Unit or NX-series Digital Input Unit according to the setting of Mode: Drive Mode/Controller Mode of the *_sTRIGGER_REF* of structure variable.
- Refer to *Setting TriggerInput* on page 90 for *_sTRIGGER_REF*.
- Dashed line in the figure represents mechanical connection.
- Gap sensor is connected only when using the NoGapNoSeal function.

● Software Configuration

Configure the MasterAxis and the SlaveAxis as shown below.
The Axis settings which are not described in the table below, should match the machine specification, referring to motion control user’s manual

Master Axis Configuration

Classification	Parameter name	Set value
Axis Basic Settings	Axis Type	Encoder axis*1
Position Count Settings	Count Mode	1: Rotary Mode
	Modulo Maximum Position Setting Value	Positive value*2
	Modulo Minimum Position Setting Value	0

*1. When using a virtual servo axis, set the axis type to a virtual servo axis.

*2. Set the value according to the device where this FB is used.

Slave Axis Configuration

Classification	Parameter name	Set value
Axis Basic Settings	Axis Type	Servo axis*1
Position Count Settings	Count Mode	1: Rotary Mode
	Modulo Maximum Position Setting Value	Circumference of the cutter
	Modulo Minimum Position Setting Value	0
Operation Setting	In-position Check Time	0

*1. When using a virtual servo axis, set the axis type to a virtual servo axis.

⚠ WARNING	
Do not use this FB in a system that consists of devices and versions not specified in this document. Doing so may cause the system to operate in unintended ways.	❗
When using this FB, set the count mode of the MasterAxis and SlaveAxis to Rotary Mode. The SlaveAxis may operate in unintended ways if you set the MasterAxis and SlaveAxis settings that are different from those specified in this document.	⚠

Usage Conditions of the FB

The chapter describes usage conditions of the FB. The usage conditions are such as execution conditions of the FB and setting conditions for equipment.

Execution Conditions of the FB

This section describes the execution conditions of this FB.

● Execution Conditions of the MasterAxis and SlaveAxis

If all of the following conditions are not met, an error will occur at an execution.

For information on the error codes, refer to *ErrorIDEx* on page 111.

No	Condition	Sysyem defined Variable or references
1	While *Enable* is TRUE, the Servo of the SlaveAxis must be ON. (only when it is set to the Servo axis)	*_sAXIS_REF.DrvStatus.ServoOn* is TRUE.
2	While *Enable* is TRUE, the home position must be defined for the SlaveAxis.	*_sAXIS_REF.Details.Homed* is TRUE
3	There must be no error in SlaveAxis.	*_sAXIS_REF.MfaultLvl.Active* is FALSE.
4	When *Enable* changes from FALSE to TRUE, the SlaveAxis must be stopped.	*_sAXIS_REF.Status.Standstill* is TRUE
5	Setting of *SlaveCfg.WaitPos* (SlaveWaitPosition) must be made as described in this manual.	For more information, please refer to *Setting SlaveWaitPosition Input Variable* on page 89.
6	Settings of *SlaveCfg.SyncPos* (SlaveSyncPosition) and *SlaveCfg.EndPos* (SlaveEndPosition) must be set as described in this manual.	For more information, please refer to *Setting SlaveSyncPosition and SlaveEndPosition* on page 91.

● Condition for Setting Device Parameters

The physical setting condition is described below. This is not detected as an error.
When using the MarkToMark Mode and ExternalMarkToMark Mode, the relationship of the distance between the marks (cut length) on a product, and the distance (*CuttingCfg.DistanceToMarkSensor*) between the mark sensor and the cutting position shall be as follows:

Distance between the marks on a product (cut length) should be < (less than) the distance between the mark sensor and the cutting position (*CuttingCfg.DistanceToMarkSensor*).

● Condition of Task Settings

It is recommended that you execute this FB in the primary periodic task.

Precautions for Safe Use

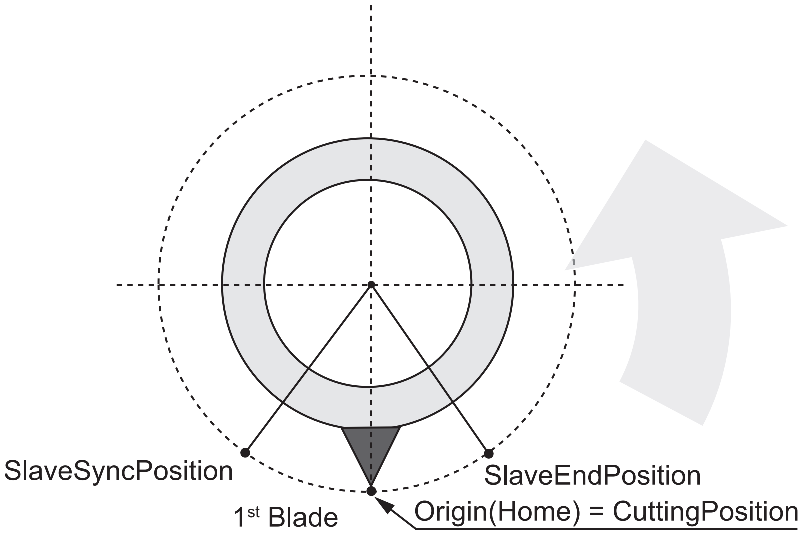
If this FB is not executed in the primary periodic task, the slave axis causes unintended operation.

Setting conditions for device

As the setting conditions of the device, there are Home of SlaveAxis, the number of blades, blade-mounting position, and servo motor rotation-direction setting.

● Home of SlaveAxis

Set the home of the SlaveAxis for this FB as shown in the following figure.



- Match the home position and the cutting position.
- Match the blade position and the home position when homing is completed.

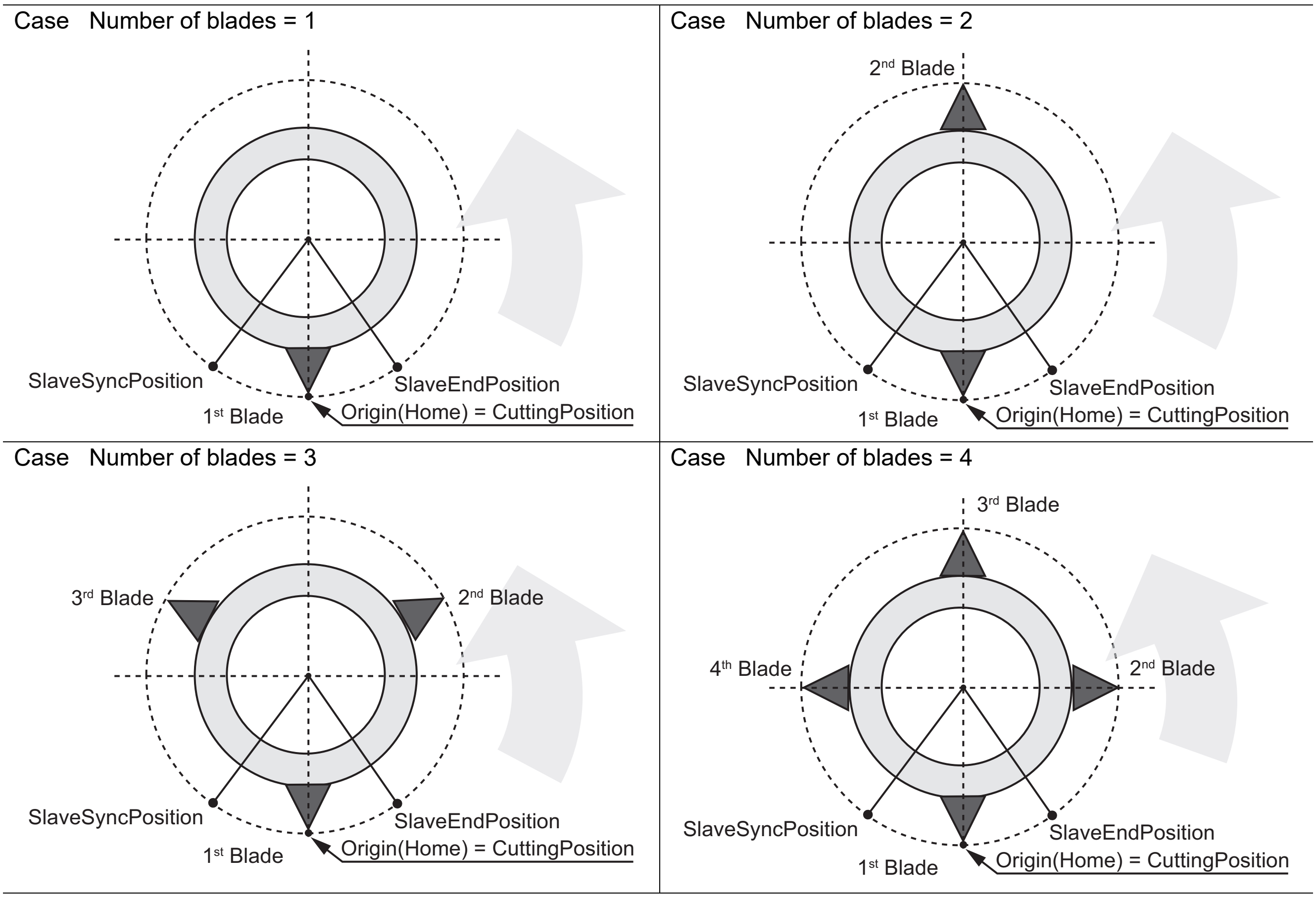
When multiple blades are mounted, any of the blades must meet above conditions.

● Number of Blades on the Rotary knife

With this FB, you can mount one to four blades.

● Mounting Position of Blades

When mounting multiple blades, mount them on the circumference at regular intervals or opposite angle.



Case Number of blades = 1

Case Number of blades = 2

Case Number of blades = 3

Case Number of blades = 4

● Setting Rotation Direction of the Servo Motor

When using this FB, the forward direction of the MasterAxis and SlaveAxis should be set as follows.

MasterAxis: set moving direction of the product to the forward direction.

SlaveAxis: set rotation direction of the rotary knife to the forward direction.

As descrived above, set rotation direction of the servo drive for the MasterAxis and SlaveAxis.

Configure the following servo parameter object.

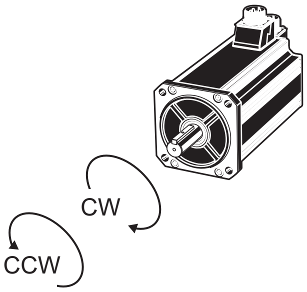
3000hex	Rotation Direction Switching All						
Setting range	0 to 1	Unit	-	Default setting	1	Data attribute	C
Size	2bytes(INT16)		Access	RW	POD map	Not possible.	

- Motor rotation direction to the direction of position command, speed command, and torque command can be switched.

Explanation of Set Values

Set value	Descripition
0	A forward direction command sets the motor rotation direction to clockwise.
1	A forward direction command sets the motor rotation direction to counterclockwise.

- The motor rotation direction when viewing the shaft from the load side is called clockwise (CW) or counterclockwise (CCW).



For details, refer to the *AC Servomotors/Servo Drivers 1S-series with Built-in EtherCAT Communications User's Manual* (Cat. No. I586).

Variables

Input Variables

Name	Data type	Default	Valid range	Change over during execution[*1]	Description
Enable	BOOL	FALSE	TRUE/ FALSE	-	*Enable* changes from FALSE to TRUE, then this FB is executed. *Enable* changes from TRUE to FALSE, then this FB stops.
CoupleOn	BOOL	FALSE	TRUE/ FALSE	Yes	Synchronization execution flag
ExecuteStop	BOOL	FALSE	TRUE/ FALSE	Yes	Executes Stop. Stops SlaveAxis depending on the StopMode of sSLAVE_CFG.
GapSensorSignal	BOOL	FALSE	TRUE/ FALSE	Yes	Inputs gap sensor detection signal. • TRUE: product exists • FALSE: no product exists
Trigger Variable	BOOL	FALSE	TRUE/ FALSE	Yes	Trigger variable[*2]
Recorded Position	LREAL	0.0	Negative, Positive, 0	Rising edge of *TriggerVariable*[*3]	Latched Position.[*2]
MasterCfg	sMASTER_CFG	Refer to *sMASTER_CFG* on page 49			MasterAxis configuration
SlaveCfg	sSLAVE_CFG	Refer to *sSLAVE_CFG* on page 50			SlaveAxis configuration
CuttingCfg	sCUTTING_CFG	Refer to *sCUTTING_CFG* on page 52			Cutting configuration

[*1] This column shows whether each parameter can be changed when FB is executed (when *Enable* is TRUE).

[*2] This variable is used in the following two cases.

Case1:
CuttingCfg.Mode=1(MarkToMark) and *MasterCfg.TriggerInput.mode*=1(_mcController)

Case2:
CuttingCfg.Mode=2(ExternalMarkToMark)

[*3] Value of input variable is enabled in the rising edge of Trigger Variable.

For input variables, structures defined by user are used. For details of the structures, refer to the following sub-items.

● sMASTER_CFG

Name	Data type	Default	Valid range	Change over during execution	Description
TriggerInput	_sTRIGGER_REF	Refer to *Setting TriggerInput* on page 90.		No*1	Conditions of Trigger Input are specified.
CmdPosMode	BOOL	FALSE	TRUE: MasterAxis.Cmd.Pos FALSE: MasterAxis.Act.Pos	No*1	The type of a position, which is referred to as a master axis, is changed to *MasterAxis.Cmd.Pos*. This setting can be used only for the Servo axis.
EnableAdvanceAngleCorr	BOOL	FALSE	TRUE/ FALSE	No*1	The advance angle compensation function is enabled.
AdvanceAngleCorrValue	LREAL	0.0	Depends on the data type. When a value is five times as much as the moved distance from the previous cycle position or is larger, the value is treated as the fivefold value.*2	Yes	The value to compensate an advance angle is input. The compensation amount must be specified with a relative value. This value is read when the advance angle compensation function is enabled.

*1. No: When *Enable* changes to TRUE, the value of the input variable is enabled.

*2. When the previous cycle position is 100 and the current cycle position is 120, the difference is 20. When a value greater than 100 is input as the value to compensate the advance angle, the value is treated as 100.

● sSLAVE_CFG

Name	Data type	Default	Valid range	Change over during execution	Description
SyncPos	LREAL	0.0	0 to Modulo Maximum Position Setting Value of Slave	No[*1]	Synchronization start position of SlaveAxis (SlaveSyncPosition)
EndPos	LREAL	0.0	0 < Modulo Maximum Position Setting Value of Slave	No[*1]	Synchronization end position of SlaveAxis (SlaveEndPosition)
WaitPos	LREAL	0.0	Positive Number (Less than Modulo Maximum Position Setting Value of Slave)	No[*1]	Wait position of SlaveAxis (SlaveWaitPosition)
MaxVelocity	LREAL	0.0	Positive Number (Less than Maximum Speed of Slave axis)	No[*1]	Maximum allowable velocity of SlaveAxis
Acceleration	LREAL	0.0	Positive Number (Less than Maximum Acceleration of Slave axis)	Yes	Acceleration of SlaveAxis
Deceleration	LREAL	0.0	Positive Number (Less than Maximum Deceleration of Slave axis)	Yes	Deceleration of SlaveAxis
BladeNum	UDINT	1	1 to 4	No[*1]	Number of blades
BladeMount-ErrorComp	ARRAY[1..4] OF LREAL	0.0	Values in the following range. BladeMountErrorComp[1]: 0 BladeMountErrorComp[2] to [4]: ±(Modulo maximum position of the slave axis /the number of blades)	No[*1]	The value to compensate an advance angle for correcting the mounting error is input. The rotation direction is indicated with a positive value and the counter rotation direction is indicated with negative value.
ProfileType	UDINT	10#0	0:Trapezoidal 5:5-order polynomial	No[*1]	Set the cam profile type.
VelocityTo WaitPos1	LREAL	0.0	Positive Number (Less than Maximum Velocity of Slave axis)	No[*1]	Approaching velocity to *WaitPos* at startup and buffer empty.
VelocityTo WaitPos2	LREAL	0.0	Positive Number (Less than Maximum Velocity of Slave axis)	No[*1]	Approaching velocity to *WaitPos* at the time of cutting position indefinite

Name	Data type	Default	Valid range	Change over during execution	Description
VelFactor	UDINT	100	50 < VelFactor < 200	Yes	Set the velocity override factor during the InSyncPhase. The setting unit is %.
StopMode	_eMC_STOP_MODE	0: _mcDecceleratio nStop	0: _mcDeccelerationStop 1: _mcImmediateStop	Yes[*2]	Stop mode

*1. No: When *Enable* changes to TRUE, the value is read.

*2. Yes: When *ExecuteStop* changes to TRUE, the value is read.

sCUTTING_CFG

Name	Data type	Default	Valid range	Change over during execution	Description
Mode	UDINT	10#0	0:Continuous 1:Mark to Mark 2:External Mark to Mark	No*1	Sets the operating mode.
CutLength	LREAL	0.0	0 and greater	Yes	Sets the cut length. For details, refer to *Cut-Length Function* on page 75.
Tolerance ForMark	LREAL	25.0	0.0 to 25.0	Yes	Sets the tolerance value for the mark sensor. When *CuttingCfg.Mode* is 1 or 2, this function is valid. For details, refer to *Tolerance Function* on page 76.
WindowFor-Gap	LREAL	0.0	0.0 and greater	Yes	Sets product detection range for the gap sensor. For details, refer to *Detection range setting* on page 83.
MasterSync-Pos	LREAL	0.0	0 and greater	No*1	Synchronous start position of the Master-Axis.This setting is enabled only when *CuttingCfg.Mode* is 0. You can set a value that exceeds the Modulo Maximum Position Setting Values.
Distance-ToMarkSensor	LREAL	0.0	0 and greater	Yes	Distance between the mark sensor and the cutting position. When *CuttingCfg.Mode* is 0, this variable is invalid.
EnableNGNS	BOOL	FALSE	TRUE/ FALSE	No*1	Enable the NoGapNo-Seal function.
Distance-ToGapSensor	LREAL	0.0	2**CuttingCfg.CutLength* or greater	No*1	Distance from the gap sensor to the cutting position. Mount the gap sensor to the position of the two times greater than the value of the cut length, and set the distance to this variable.

Name	Data type	Default	Valid range	Change over during execution	Description
Offset From Mark	LREAL	0.0	0 to *Cutlength*	Yes	Sets the cut length (offset value) from the mark.If 0 is set, the product is cut at the mark position. When *CuttingCfg.Mode* is 0, this setting is disabled.
SkipCountLimit	UDINT	0	0 to 9	No[*1]	Sets the skip count to be allowed. When the skip count reaches this set value, an error will occur.

[*1] No: When *Enable* changes to TRUE, the value is read.

Output Variables

Name	Data type	Description
Enabled	BOOL	TRUE while FB is executing.
Done	BOOL	TRUE while the SlaveAxis is waiting at the SlaveWaitPosition.
InSync	BOOL	TRUE while the MasterAxis and SlaveAxis are being synchronized.
Cutting	BOOL	TRUE for one task period during cutting operation.
CutLength	LREAL	Outputs the cut length.[*1]
Skip	BOOL	TRUE for one task period during cutting operation when it is determined that the next cutting operation is skipped.
BufferEmpty	BOOL	TRUE for only one task period during cutting operation when the next cut length is unknown because the buffer of the cut length is empty. This variable is output only when *CuttingCfg.Mode* is 1 and 2.
MarkNotAccepted	BOOL	When BufferEmpty occurs and the mark sensor detects a mark on the way to Slave Wait Position, the variable becomes TRUE only for one task cycle.[*2] The variable is output only when *CuttingCfg.Mode* is 1 or 2.
Stopped	BOOL	When the blade stop function stops the blade completely, this becomes TRUE. TRUE output is held while Input *Enable* is TRUE.
InvalidCuttingPos	BOOL	When the next cut position is indefinite due to the NoGapNoSeal function, TRUE is output. After the blade moves back to the wait position and then the next cutting position is fixed, FALSE is output.
Busy	BOOL	TRUE when the SlaveAxis is moving. When *Done* changes TRUE, *Busy* becomes FALSE. FALSE during non-execution status when there is an error or the operation is aborted.
CommandAborted	BOOL	TRUE when an execution of the instruction is aborted.
Error	BOOL	TRUE when an error occurs.
ErrorID	WORD	Contains the error code when an error occurs.[*3] #0000 is the initial value and indicated as normal end.
ErrorIDEx	DWORD	Contains the error code when an error occurs.[*3] #00000000 is the initial value and indicates normal end.

[*1] When *CuttingCfg.Mode* is 0 (*Continuous*), this variable outputs the cut length that is set with the input variable. When *CuttingCfg.Mode* is 1 (*MarkToMark*) or 2(*ExternalMarkToMark*), this variable outputs the distance between marks that is detected by the mark sensor. If the distance between marks is greater than the *DistanceToMarkSensor*, this variable outputs the *DistanceToMarkSensor*.

[*2] For “*BufferEmpty*”, refer to *Buffer Function* on page 73.

[*3] Refer to *Troubleshooting* on page 111 for details.

In-Out Variables

Name	Data type	Valid range	Change over during execution	Description
Master	_sAXIS_REF	-	No	MasterAxis
Slave	_sAXIS_REF	-	No	SlaveAxis
ContinuousOpe	sCONTINUOUS _OPE	-	-	Continuous operation data

- **sCONTINUOUS_OPE**

The specifications of sCONTINUOUS_OPE are as follows:

Name	Data type	Default	Valid range	Change over during execution	Description
Store	BOOL	FALSE	TRUE/FALSE	Yes	Storing
Stored	BOOL	FALSE	TRUE/FALSE	Yes	Stored
ValidData	BOOL	FALSE	TRUE/FALSE	No	Continuous operation data is valid.
Data	sCONTINUOUS _DATA	-	-	No	Continuous operation data

- **sCONTINUOUS_DATA**

The specifications of sCONTINUOUS_DATA are as follows:

Name	Data type	Default	Valid range	Change over during execution	Description
MasterAxisPosition	LREAL	0.0	0 < modulo maximum position of the master axis	No	Master axis position
MovingAmount FromCuttingPos	LREAL	0.0	0 and greater	No	Moving amount of the master axis from the cutting position
Current	sCUTTING_DATA	-	-	No	An object to be cut during interruption
Buffer	ARRAY[0..63] OF sCUTTING_DATA	-	-	No	Buffered cut data

- **sCUTTING_DATA**

The specifications of sCUTTING_DATA are as follows:

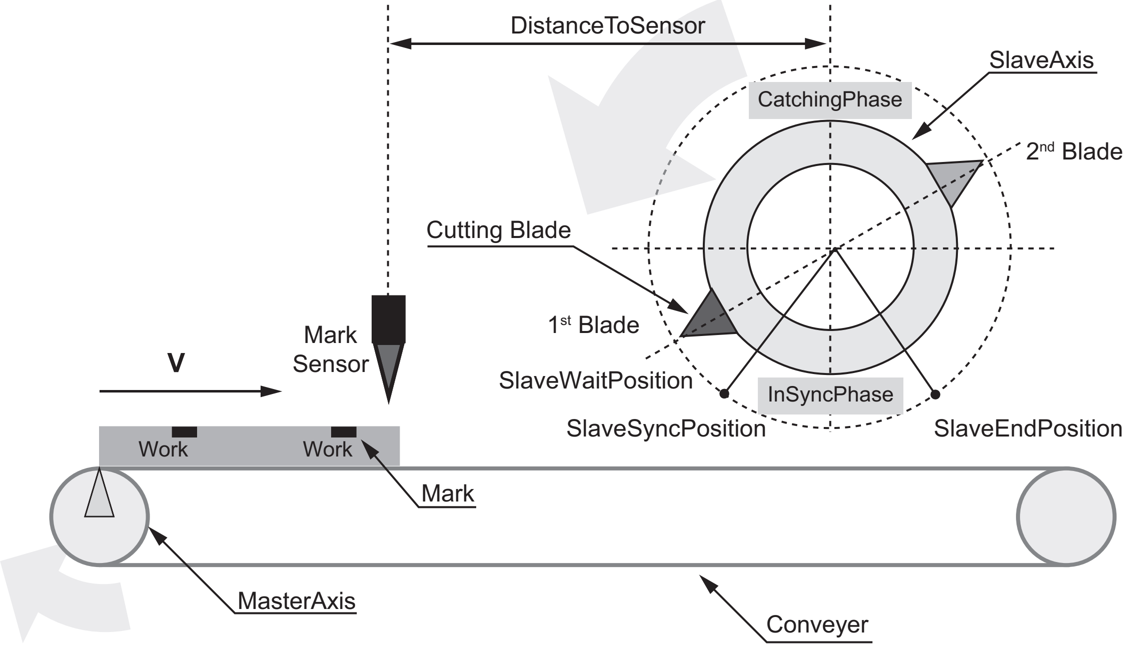
Name	Data type	Default	Valid range	Change over during execution	Description
CutLength	LREAL	0.0	Positive number	No	Cut length
State	UINT	0	0: invalid 1: not checked NoGapNoSeal 2: checked NoGapNoSeal and no product 3: checked NoGapNoSeal and product exists	No	State of the product detection by Gap sensor.

Function

Application Configuration

This section describes an example of application supported by this FB.
The figure below shows an application in which conveying products are cut at certain intervals.

The product is conveyed by the MasterAxis and is cut by the rotary knife of SlaveAxis.
The Mark Sensor detects the Mark on the product to cut.
The MasterAxis and the SlaveAxis are synchronized during a SlaveAxis is in InSyncPhase and the product is cut by the blade(s) of the rotary knife.

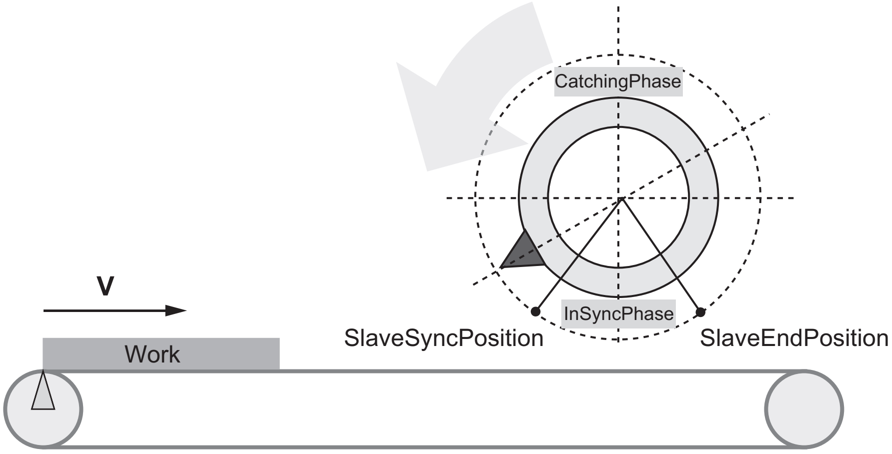
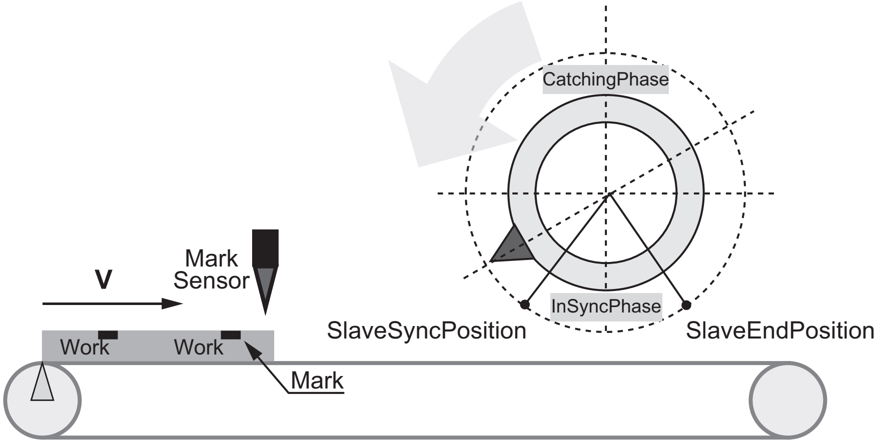


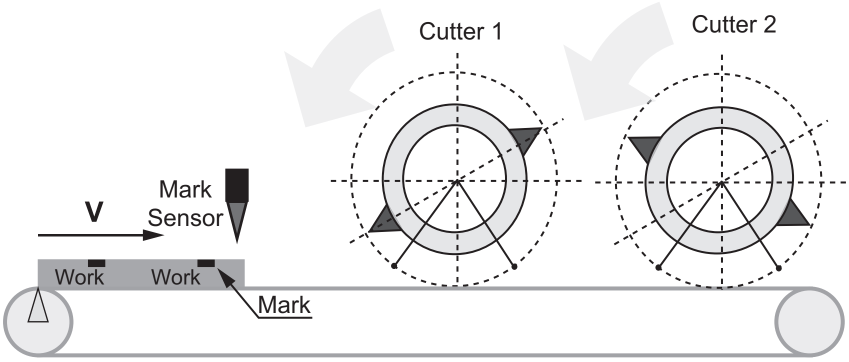
This chapter describes the functions this FB uses. The functions are as follows:

- Operating mode selecting function
- Skip Function
- Cam profile Function
- Cycle Stop Function
- Offset From Mark Function
- Buffer Function
- Velocity Override Function
- Cut Length Function
- Tolerance Function
- Advance Angle Compensation
- Blade Mount Error Compensation
- Adjust DistanceToMarkSensor in operation
- NoGapNoSeal
- Continuous Operation
- Stop Blade

Operating mode selecting function

This FB performs three types of operation depending on the set value of *CuttingCfg.Mode* of input variable. For details, refer to *Continuous* on page 58, *Mark To Mark* on page 62, and *External Mark To Mark* on page 66.

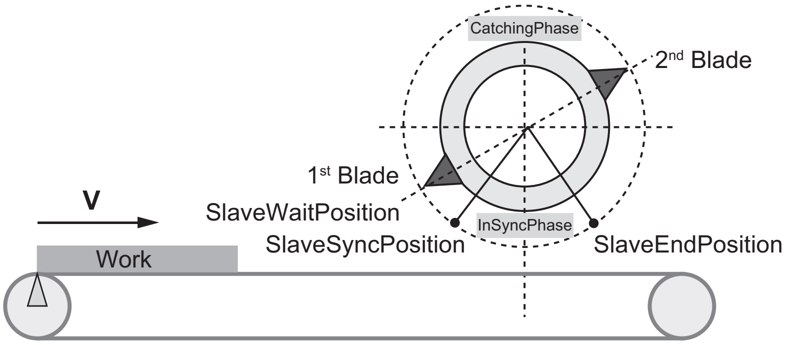
CuttingCfg.Mode	Description
0: Continuous	When the blade(s) is cutting in continuous Mode the blade(s) will continuously perform a calculated profile that defines a fix cutting length based on the master axis increment. This Mode is the simplest rotary cutting operation and is not taking in account the mark sensor probe input.  CatchingPhase V InSyncPhase SlaveSyncPosition SlaveEndPosition Work
1: Mark To Mark	In this Mode the mark sensor will calculate the distance between two probe detection in order to ensure that cut always fits the mark. Cut length input parameter is ignored. At least two probe detections with the mark sensor are necessary to calculate the cut length.  CatchingPhase Mark Sensor V InSyncPhase SlaveSyncPosition SlaveEndPosition Work Work Mark

CuttingCfg.Mode	Description
2: External Mark To Mark	Two or more rotary knives can be set for a pair of the MasterAxis. For details, refer to the following figure. The rotary knives perform the same operation as one in Mark to Mark mode, however, a program to detect marking position should be added. MC_TouchProbe command is needed. Enter output of MC_TouchProbe command to *TriggerVariable* and *RecordedPosition* of this FB.  Cutter 1 Cutter 2 Mark Sensor V Work Work Mark

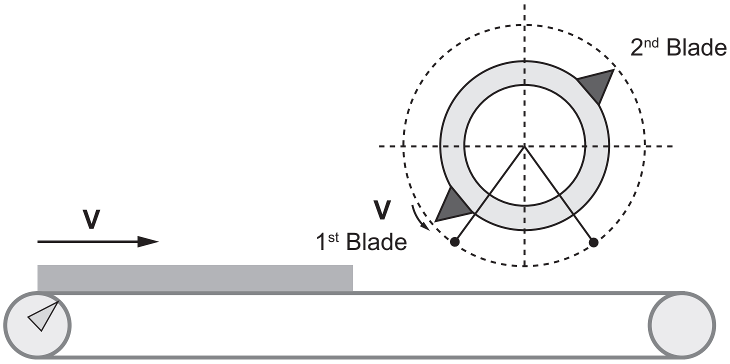
● Continuous

This section describes the operating mode of *Continuous*.

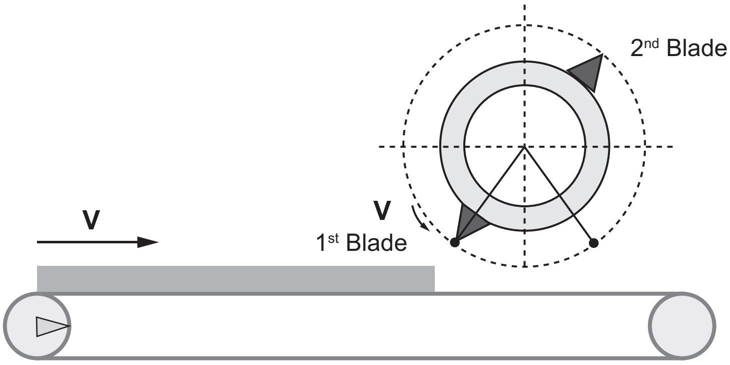
1 When *Enable* of input variable changes to TRUE, the blade which operates in forward rotation and can move to the SlaveWaitPosition in the shortest distance moves to the SlaveWaitPosition and waits here until *CoupleOn* changes to TRUE.



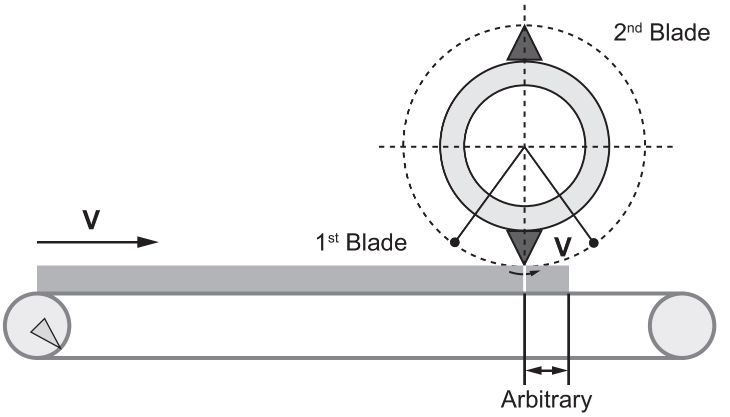
2 After *CoupleON* of input variable changes to TRUE, the rotary knife starts rotation.



3 When the MasterAxis reaches the MasterSyncPosition, the rotary knife performs synchronization.
InSync changes to TRUE.

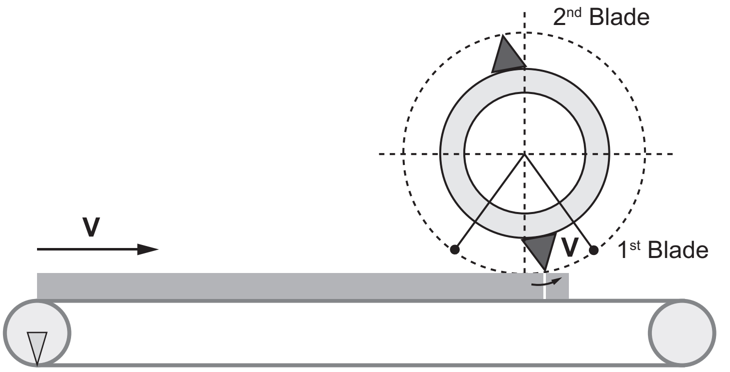


4 The rotary knife rotates and passes the cutting position.
At this time, a product is cut if there is any at the cutting position. The cutting position is not defined.
When the blade passes the cutting position, *Cutting* changes to TRUE.

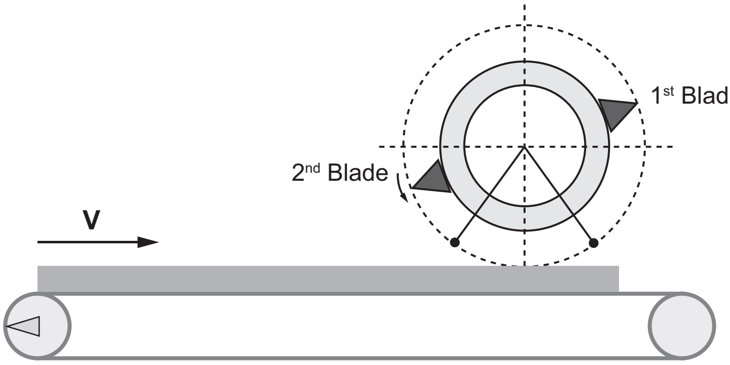


5 After the first cutting, a cam profile is calculated for the next cutting.
After this, the rotary knife will operate according to this cam profile.

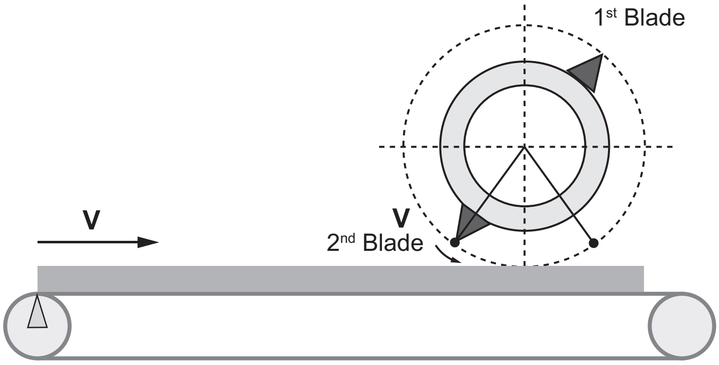
You can change the cut length before the previous cutting operation executes.



6 The 2nd blade is moving in the CatchingPhase preparing for the next cutting.

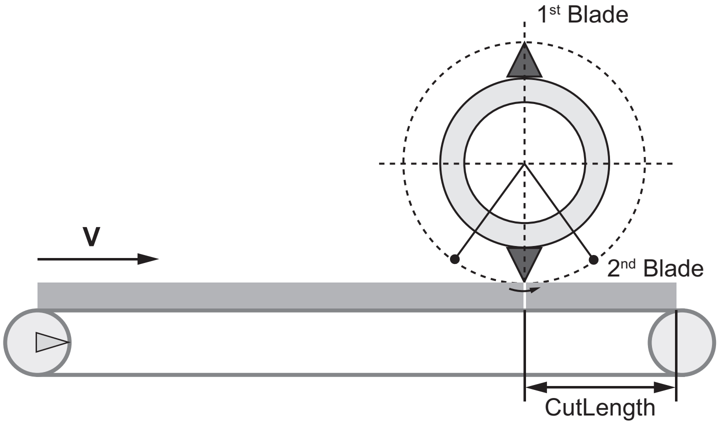


7 The 2nd blade starts synchronization at the SlaveSyncPosition.

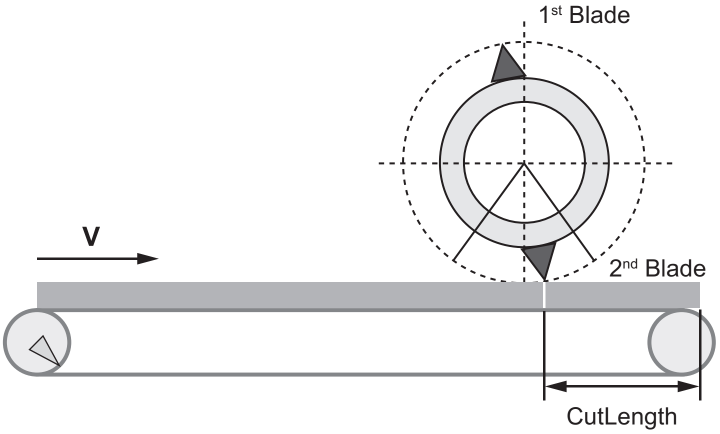


8 The 2nd blade cuts a product. The cut length at this time is the value set with *CuttingCfg.CutLength* of input variable.
During cutting operation, *Cutting* is TRUE.

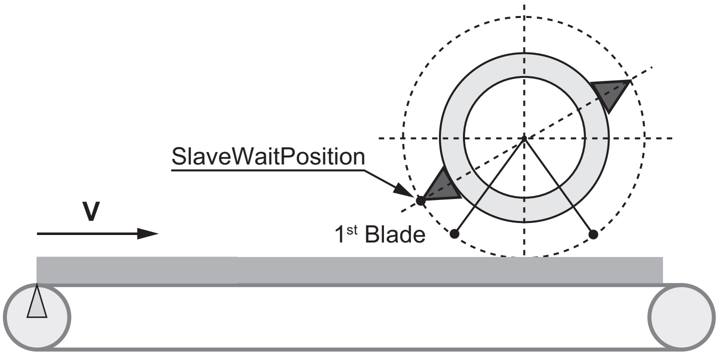
You can change the cut length before calculation is executed.



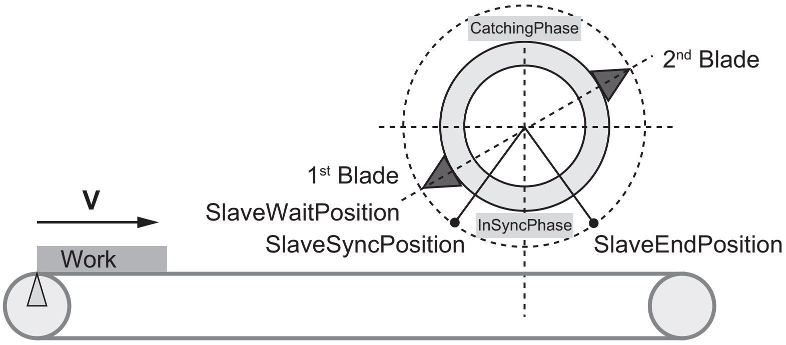
9 After that, the same operation is performed repeatedly and products are cut.



10 When *CoupleOn* changes to FALSE, the blade performs a CycleStop considering the SlaveWaitPosition as the target position.



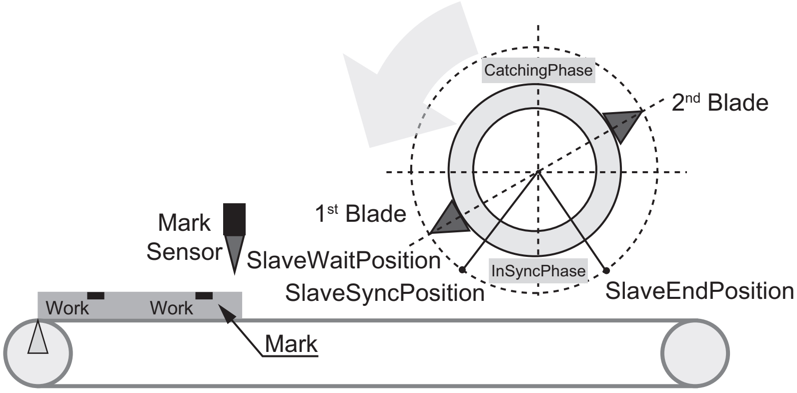
11 After a CycleStop, if a *CoupleOn* change to TRUE again while *Done* is TRUE, the operation resumes from step 1.



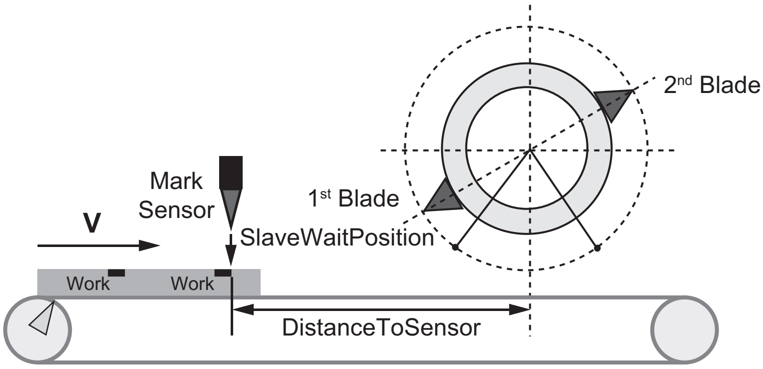
● Mark To Mark

This section describes the operation of the Mark to Mark operating mode.

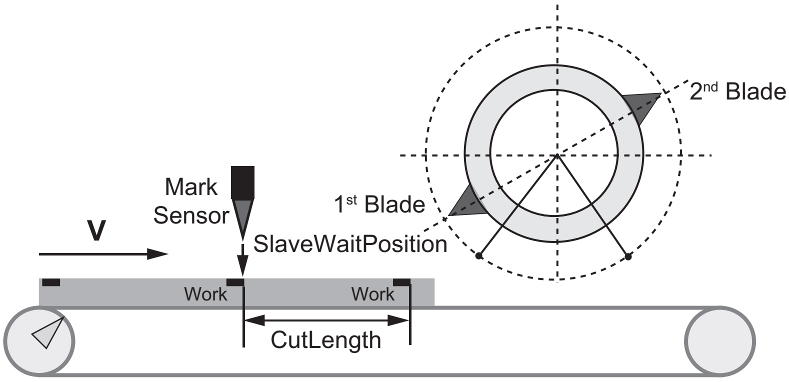
1 When *Enable* of input variable changes to TRUE, the blade which operates in forward rotation and can move to the SlaveWaitPosition in the shortest distance moves to the SlaveWaitPositon.



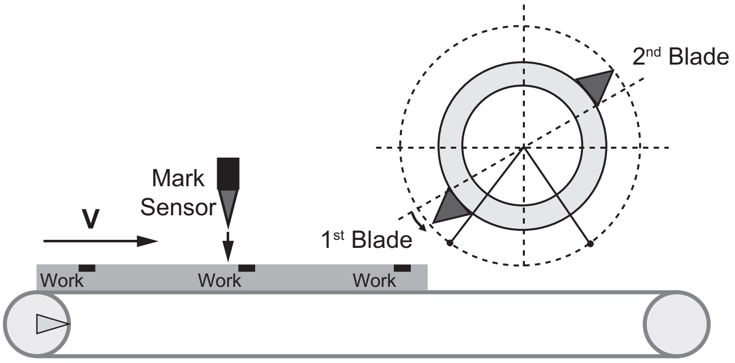
2 After moving to the SlaveWaitPosition is completed, when *CoupleOn* of input variable is TRUE and the mark on a product is detected by the mark sensor, synchronization starts.



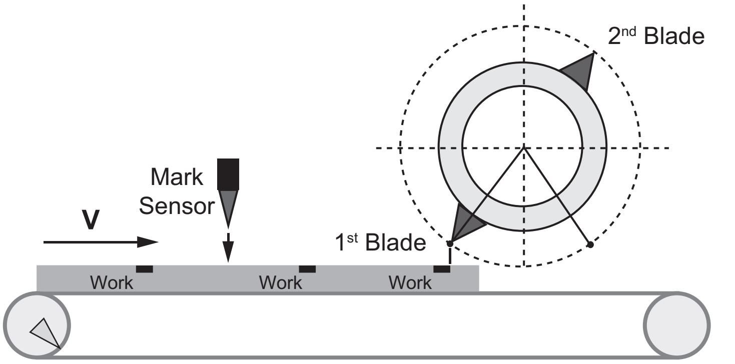
3 When the second mark is detected, the first cut length is determined.



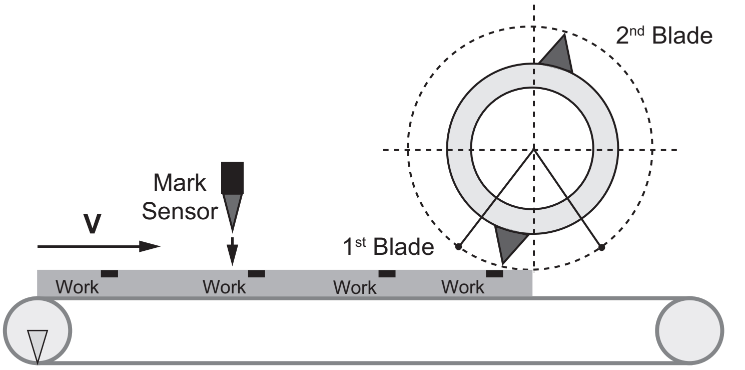
4 The 1st blade is synchronized with the first mark. If the *OffsetFromMark* of input variable is set, the blade synchronizes behind the mark with the set value.



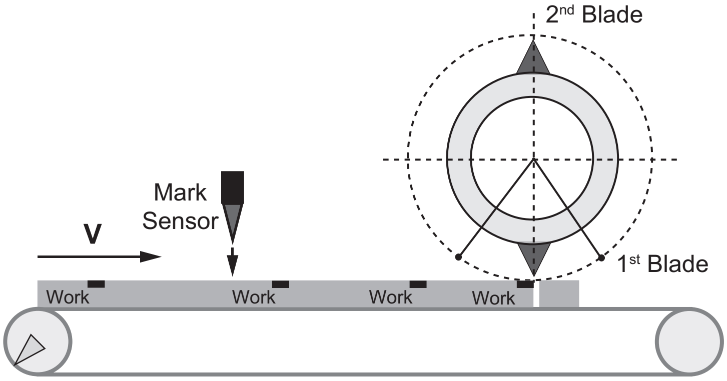
5 The 1st blade is synchronized with the mark at the synchronization start position (SlaveSyncPosition).



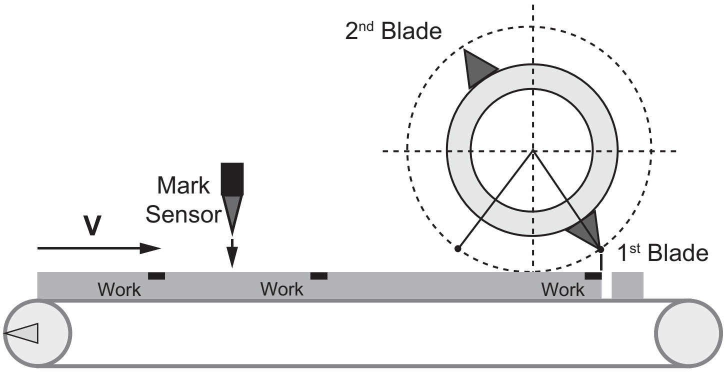
6 The 1st blade is in synchronization in the InSyncPhase. Value of *InSync* changes to TRUE.



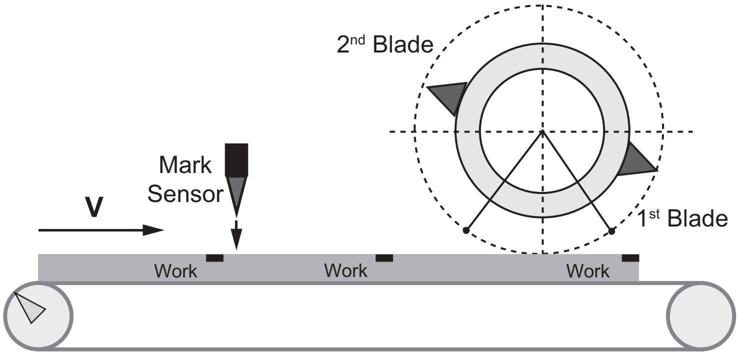
7 The 1st blade cuts the product at the first mark position.
If the *OffsetFromMark* input variable is set, the blade cuts the product behind the mark position for the set value.
After the cutting operation, the cam profile is calculated again to prepare for the next cutting operation by the 2nd blade.



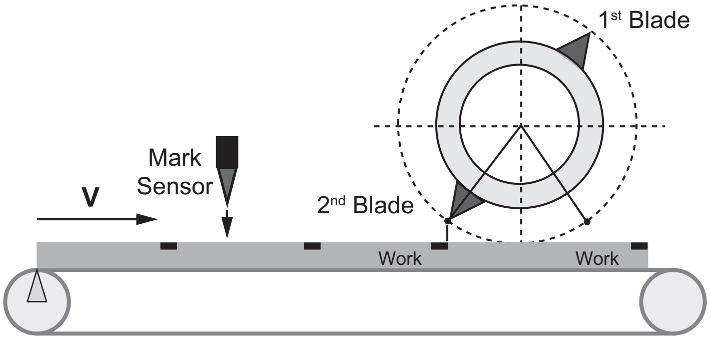
8 After the cutting operation, the 1st blade is synchronized with the mark position until it exits the InSyncPhase.



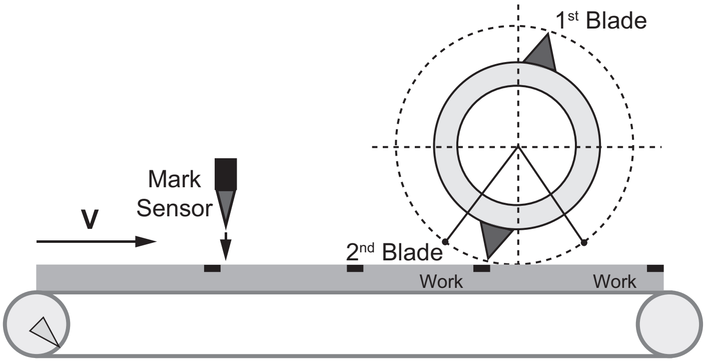
9 The 2nd blade exits the InSyncPhase and the 2nd blade is moving in the CatchingPhase.



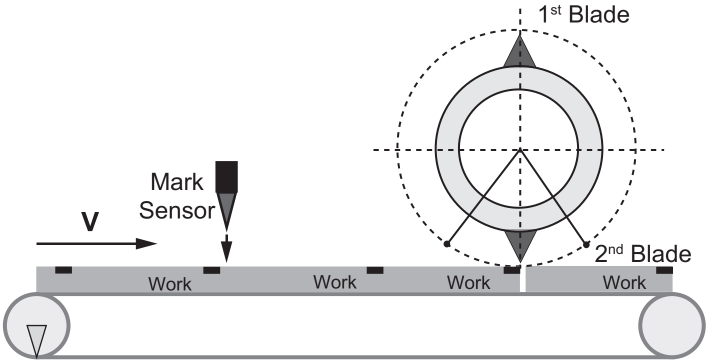
10 The 2nd blade is synchronized with the mark position.



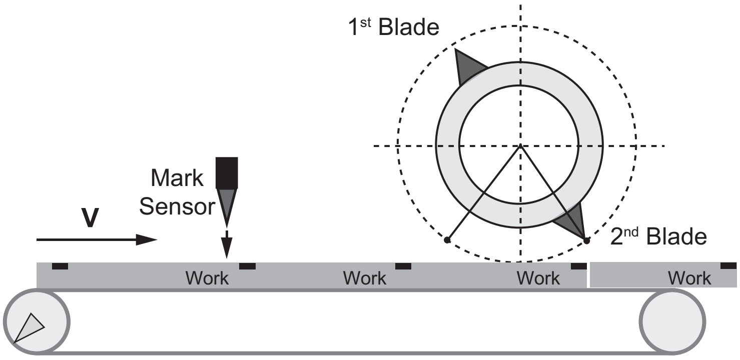
11 The 2nd blade is in synchronization. *InSync* changes to TRUE.



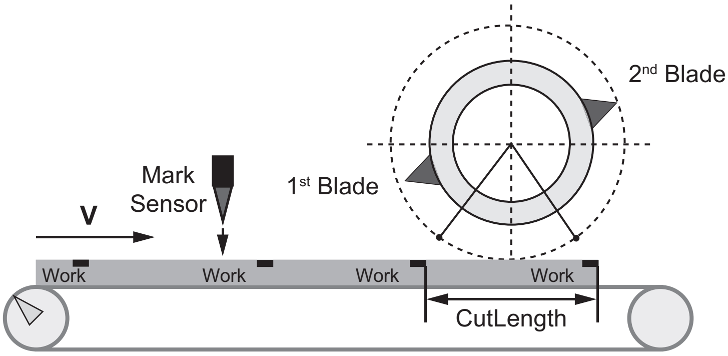
12 The 2nd blade cuts the product. If the *OffsetFromMark* of input variable is set, the blade cuts the product behind the mark position for the set value.
After the cutting operation, a cam profile is calculated again for the 1st blade to prepare for the next cutting.



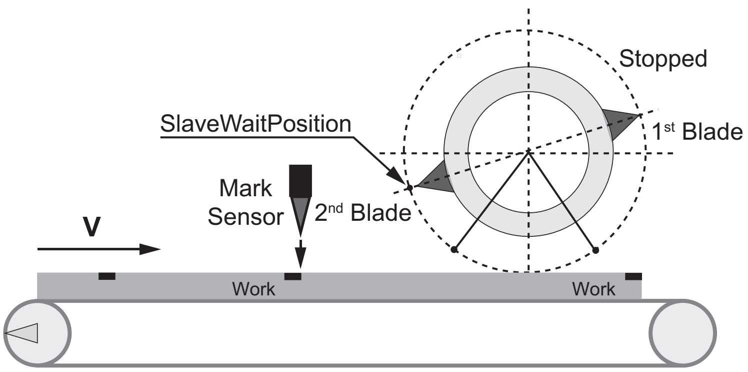
13 The 2nd blade reaches the end of the InSyncPhase.



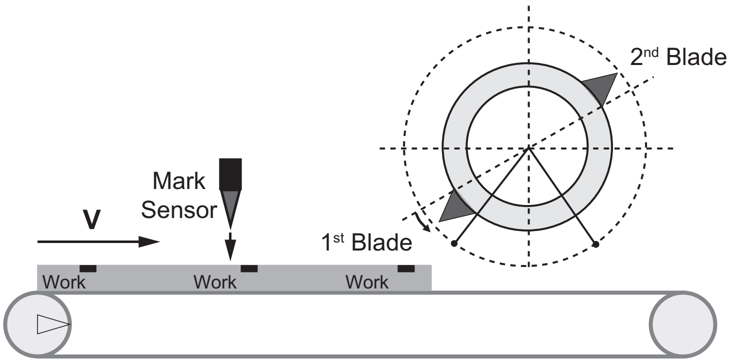
14 The 1st blade executes the follow operation according to the cam profile.



15 When *CoupleOn* changes to FALSE, a CycleStop is performed.



16 After a CycleStop, when *CoupleOn* changes to TRUE again, the operation resumes from step 2.



Precautions for Correct Use

When you set the position of the mark sensor, the DistanceToMarkSensor must be longer than the distance between the marks (cut length).

External Mark To Mark

This section describes the operation of the External Mark to Mark operating mode.
This mode is similar to Mark To Mark mode.
However, unlike the Mark To Mark mode, more than one rotary knife is controlled synchronously for a pair of the master axis and the mark sensor.

The following consideration is required when you use this FB.

- In this mode, the function block MC_TouchProbe shall be used in the same program to detect marks and measure the cut length to the marks.
- Input the *Done* and *RecordedPosition* of MC_TouchProve in the *TriggerVariable* and *RecordedPosition* of this FB, respectively.

The sample program using this mode is described in *Sample Programming1* on page 116.

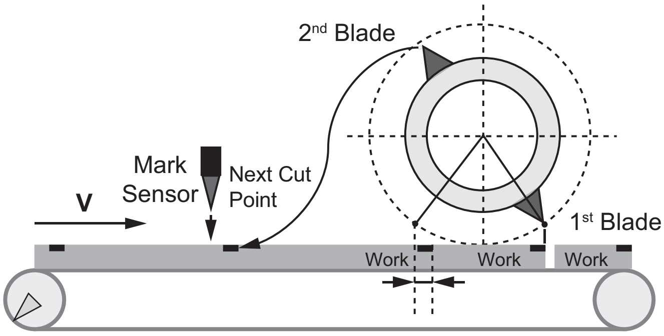
Skip Function

This section describes the skip function.
If the cut length is too long or the product is moving at an extremely rapid velocity (V), the rotary knife may fail to catch up with the cutting position.
In this case, the rotary knife of SlaveAxis will skip the current cutting position and target the next cutting position.
If the distance is still insufficient, the rotary knife will target the following cutting position.
If the skip count reaches SkipCountLimit, an error will occur.
When a cutting operation is executed successfully, the skip count is cleared to zero.
Skip count limit should be input into *CuttingCfg.SkipCountLimit*.

The following figures show cases in which the rotary knife fails to catch up with a cutting position.

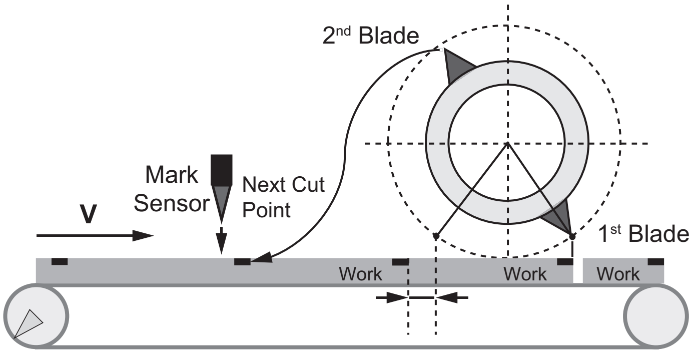
● Case 1

When cutting a product, the next cutting position is already in the InSyncPhase.



● Case 2

After cutting, the next cutting position is outside the InSyncPhase. However, the rotary knife fails to catch up with it moving at the specified acceleration/deceleration.



ProfileType Function

This section describes the Profile Type Function.

With the Profile Type Function, two cam profiles can be selected according to the value of *SlaveCfg.ProfileType*. With this FB, Trapezoid cam profile and Polynomic 5 cam profile can be selected. However, the cam profile changes depending on the relation between cut length and perimeter of the rotary knife. Details of the change are described as follows.

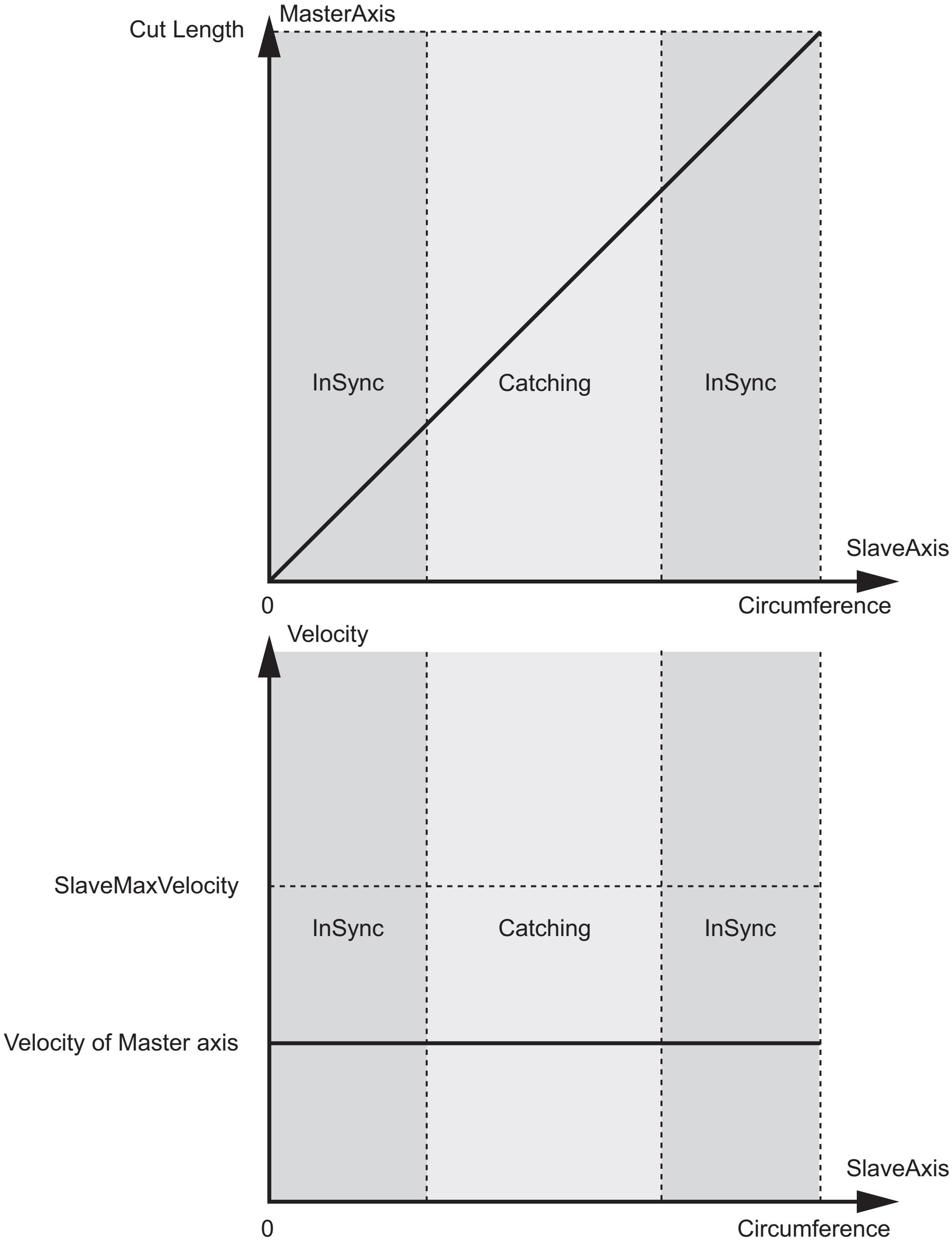
● When the Cut Length is Same as the Perimeter of the Rotary knife

This section shows the cam profiles and the velocity waveforms of the SlaveAxis when the cut length is the same as the perimeter of the rotary knife.

The cam profiles are straight lines during both InSync and Catching regardless of the setting of *SlaveCfg.ProfileType*.

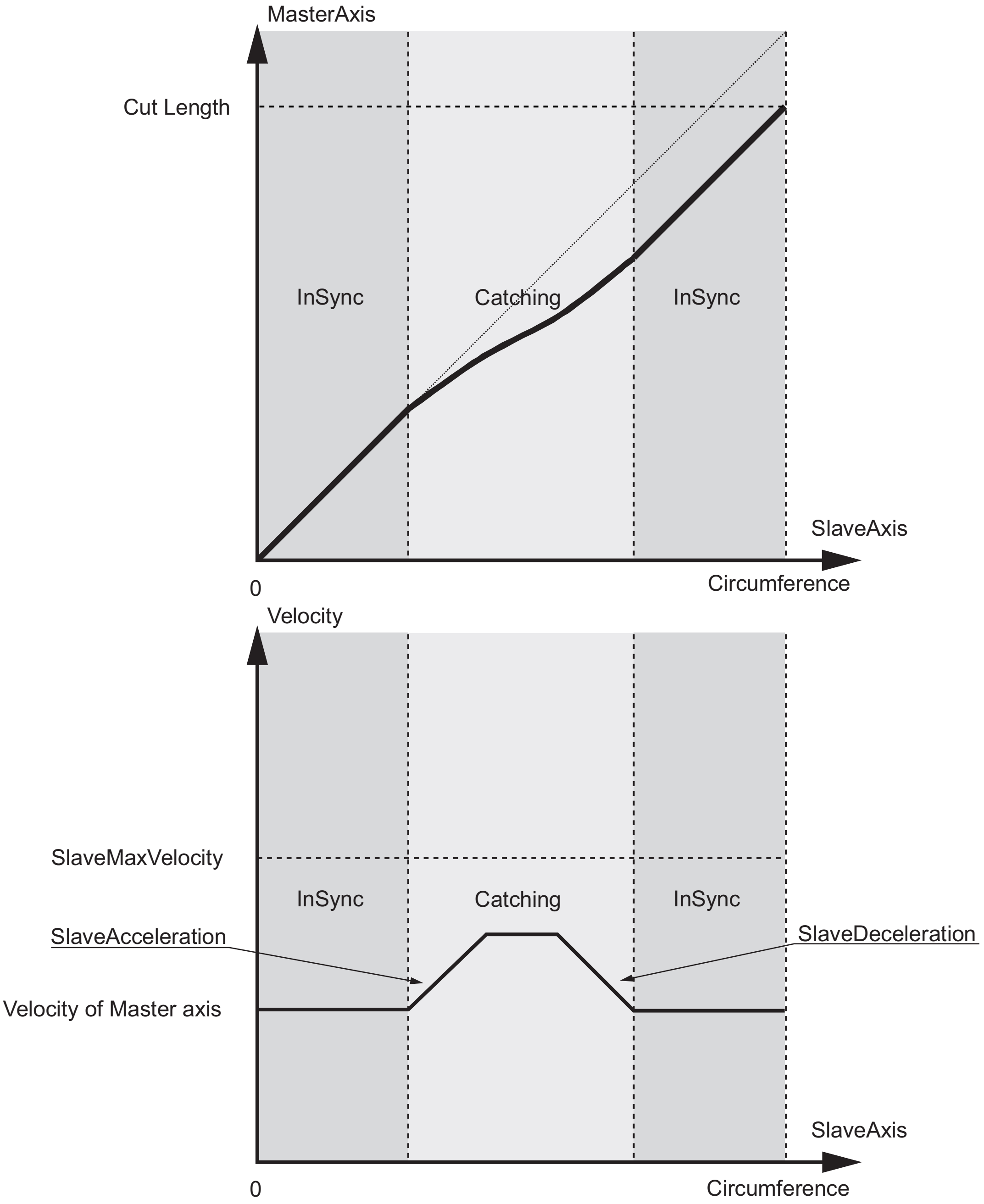
InSync: Synchronized section

Catching: Catching section outside of the Synchronized section



● When the Cut Length is shorter than the perimeter of the rotary knife

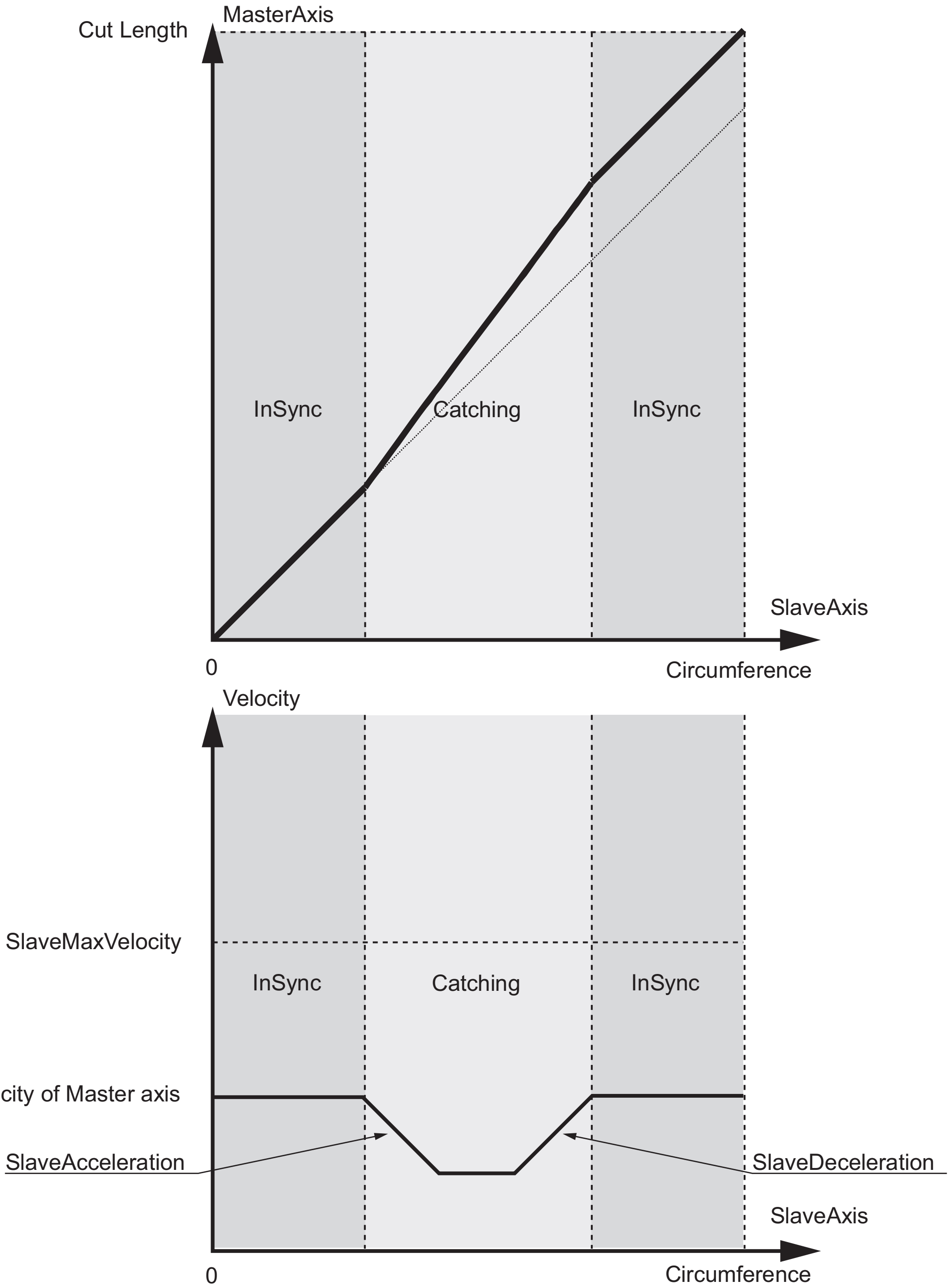
This section shows the cam profiles and the velocity waveforms of the SlaveAxis when the cut length is less than the perimeter of the rotary knife.
The cam profiles during Catching change according to the setting of the *SlaveCfg.ProfileType* of input variable.



During Catching, the SlaveAxis accelerates and decelerates based on the velocity profile according to the setting of the *SlaveCfg.ProfileType* of input variable. The figure above is for when *SlaveCfg.ProfileType* is set to Trapezoid.

When the Cut Length is longer than the perimeter of the rotary knife

This section shows the cam profiles and the velocity waveforms of the SlaveAxis when the cut length is greater than the perimeter of the rotary knife.
The cam profiles during Catching change according to the setting of the *SlaveCfg.ProfileType* of input variable.

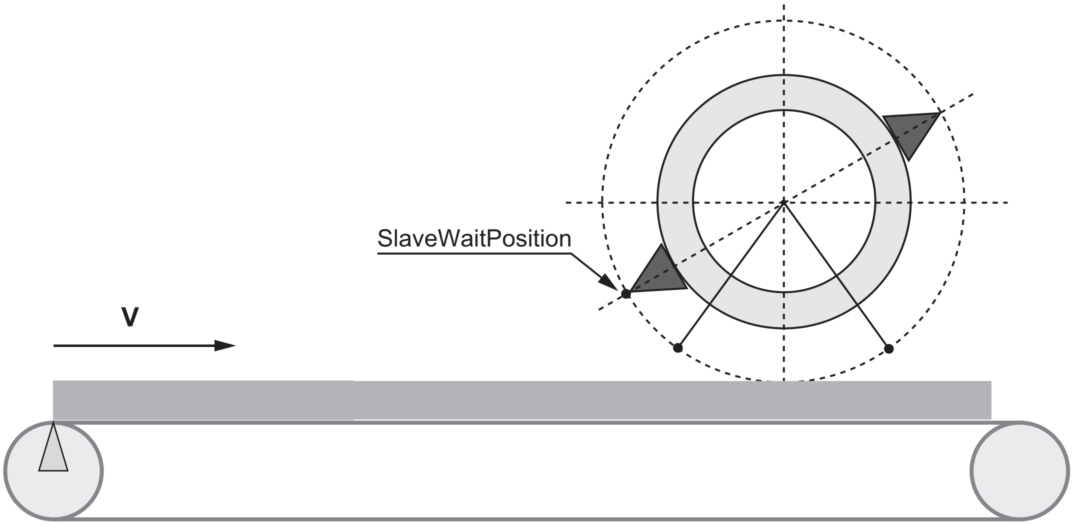


During Catching, the SlaveAxis accelerates and decelerates based on the velocity profile according to the setting of *SlaveCfg.ProfileType* of input variable. The figure above is for when *SlaveCfg.ProfileType* is set to Trapezoid.

CycleStop Function

This section describes the CycleStop function.
“Cycle” is the period between the moment when *CoupleOn* becomes FALSE ->TRUE and the moment when *CoupleOn* becomes TRUE->FALSE.

CycleStop is the motion stopping at SlaveWaitPosition after finishing Rotary Cutting by setting *CoupleOn* TRUE->FALSE.
Done is set TRUE after completing positioning at SlaveWaitPosition.
An in-position check is performed for this instruction according to the settings in In-position Range and In-position Check Time for axis parameters.
In-position Range and In-Position Check Time should be set according to machine configuration. Regarding setting procedure, refer to the motion control user’s manual.



OffsetFromMark Function

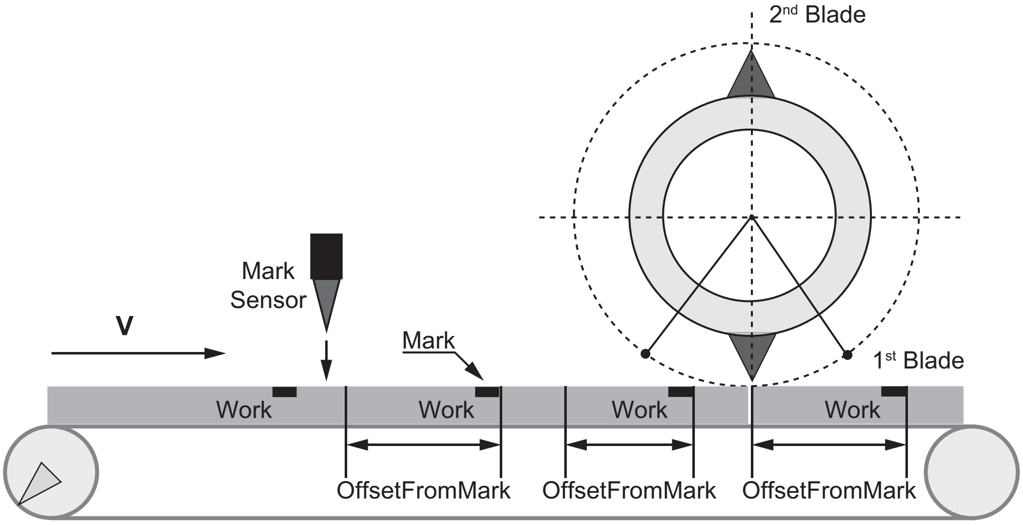
This section describes the *CuttingCfg.OffsetFromMark* of input variable.

The OffsetFromMark function can shift cutting position as much as the value which is set by the offset value. With this function, cutting position can be adjusted. The offset value should be input into *CuttingCfg.OffsetFromMark*.

Details of the OffsetFromMark function are described as follows.

- This function is enabled only when *CuttingCfg.Mode* is set to 1 (Mark To Mark mode) and 2 (External Mark To Mark mode).
- Set *CuttingCfg.OffsetFromMark* the distance from the cutting position to a Mark. See figure below.
- Setting value of *CuttingCfg.OffsetFromMark* is changeable during FB execution.
- When you change *CuttingCfg.OffsetFromMark* longer than its cutting length which is same as distance between two marks during one-cutting-cycle, it causes error.

The following figure shows the offset function. The offset function can shift the cutting position backward from the product as much as the offset value (OffsetFromMark).



⚠ Caution

Do not change the value of *CuttingCfg.OffsetFromMark* longer than cutting length. Doing so causes error.	⚠

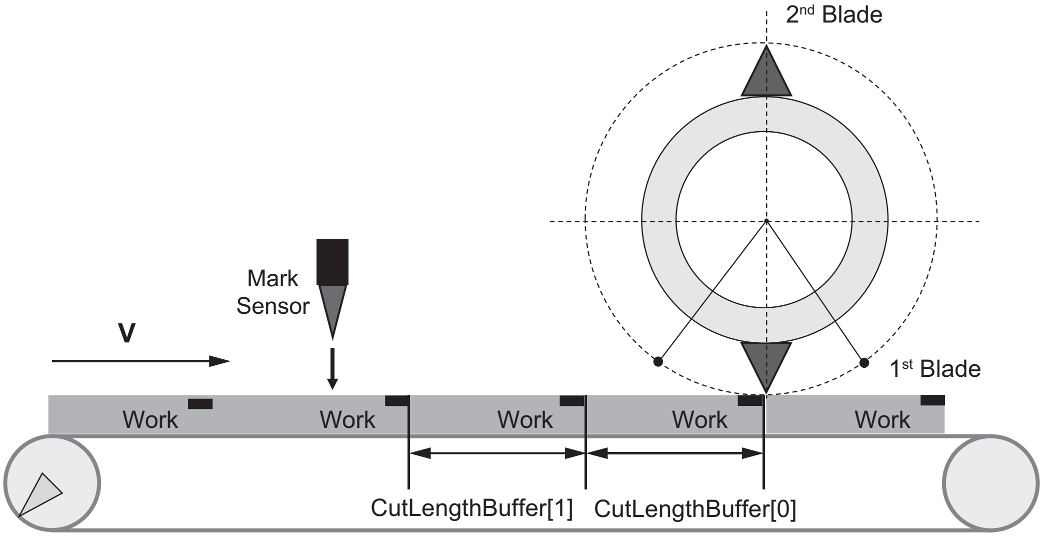
Buffer Function

This section describes the buffer function of the cut length.

When the cut length is less than *CuttingCfg.DistanceToMarkSensor*, the cut length that is measured each time the mark sensor detects a mark and the actual position of the cutting position of the product are stored in the buffer.

- This function is enabled when *CuttingCfg.Mode* is 1 (Mark To Mark) and 2 (External Mark To Mark).
- The distance between the position of the mark sensor and the cutting position must be longer than the distance between marks on the products.
- This function is used to buffer the mark information from the mark sensor to the cutting position.
- The size of buffer is 64. When this buffer size is exceeded, the buffer full error will be output and the FB operation will end.

The following figure shows buffer function of cut length. It shows that the next and subsequent cutting positions are stored in *CutLengthBuffer*.



● BufferEmpty

This section describes the BufferEmpty function.

The BufferEmpty function advises that the next cut length is unknown if the buffer is empty when cutting is executed. If the buffer is empty when cutting is executed, the output variable *BufferEmpty* is set to TRUE for one task period. Also, because the next cut length is unknown at this time, the blade moves to SlaveWaitPosition by the approaching velocity (*VelocityToWaitPos1*). Even if a mark is detected while the blade is moving to SlaveWaitPosition after the buffer empty occurs, this detection is ignored. To avoid *BufferEmpty* from setting to TRUE, refer to *Countermeasure for BufferEmpty* on page 114.

VelFactor Function

This function describes the VelFactor Function.

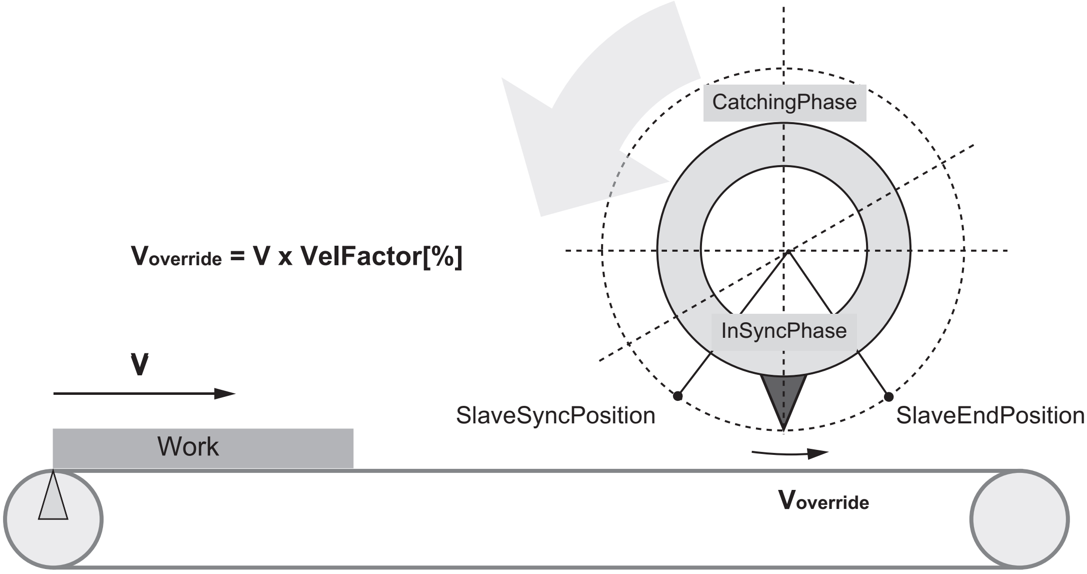
You can override the velocity of the SlaveAxis during InSyncPhase by setting the *CuttingCfg.VelFactor* input variable.

The VelFactor value should be input into the *CuttingCfg.Velfactor*.

Details of the VelFactor Function are as follows.

- The setting unit is %. Set a factor as a percentage of the MasterAxis.
- You can change the setting of *CuttingCfg.VelFactor* while moving (*Enable* is TRUE).
- The new setting will be enabled after the current cutting operation.

The following figure shows the VelFactor Function. It shows that Synchronous velocity of SlaveAxis (Voverride) is to be a value that the velocity of MasterAxis (V) is multiplied by the velocity of override factor (VelFactor).



Precautions for Correct Use

Do not input a large value into *VelFactor*. Doing so may change the command to the SlaveAxis suddenly and give excessive load to the device.

CutLength Function

This section describes the input variable *CuttingCfg.CutLength*.

Even if the mark is detected incorrectly, CutLength Function cuts the product in the fixed-length set for the cut length (input variable). The cut lengths are the input variable (*CuttingCfg.CutLength*) and the output variable (*CutLength*). The input variable is input into *CuttingCfg.CutLength*. The output variable outputs the result that is calculated based on the mark position and the input variable, to *CutLength*.

Operation of CutLength Function differs depending on the values of *CuttingCfg.Mode* and *CuttingCfg.Tolerance*. Common operation when *CuttingCfg.Mode* is set for 0 to 2 (all operating modes); Operation when *CuttingCfg.Mode* is set for 0 (Continuous mode); and Operation when *CuttingCfg.Mode* is set for 1 or 2 (MarkToMark or External Mark To Mark) are as follows:

● Common Features

- The value of out length (input variable) can be changed during execution.

● Case of [*CuttingCfg.mode* = 0] (Continuous mode)

- Cut length (output variable) outputs a value which was set for the cut length (input variable).

● Case of [*CuttingCfg.mode* =1 or 2] (MarkToMark or ExternalMarkToMark)

The set value is the cut length in the case that mark misdetection occurs. (This is Constant Length Cut function).

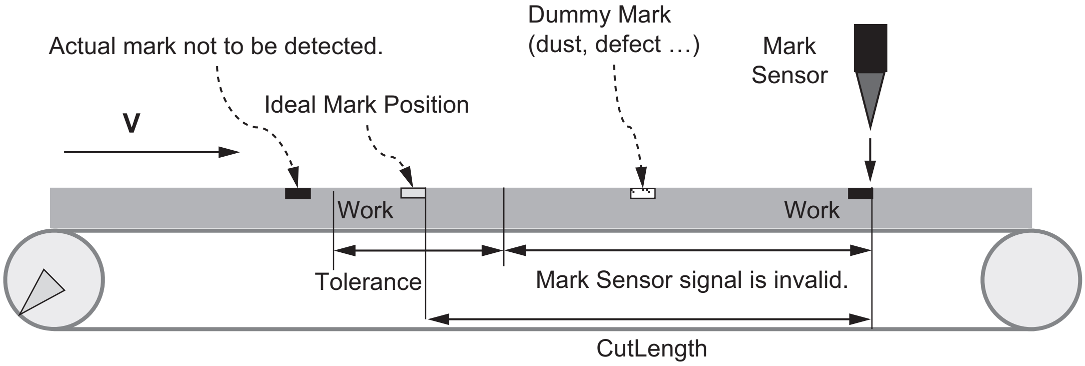
- The Constant Length Cut function prevents a machine from producing too long cut length. (See the following figure).
- When the set value is 0.0, Constant Length Cut function is disabled.
- Mark misdetection may occur in the following conditions.

a) There are some malfunctions on the mark sensor.

b) There is no mark on the product to cut.

- To use this function, users should also use the input variable *CuttingCfg.Tolerance*.

In the example below, the product will be cut at the length of *CuttingCfg.CutLength* (constant length), because the actual mark is not detected by the mark sensor in the area designated with *CuttingCfg.Tolerance*.



Additional Information

- To use “Constant length cut function”, set the value of *CuttingCfg.CutLength* smaller than that of *CuttingCfg.DistanceToMarkSensor*.
- For *CuttingCfg.ToleranceForMark*, refer to *Tolerance Function* on page 76.
- For Buffer of the FB, refer to *Buffer Function* on page 73.

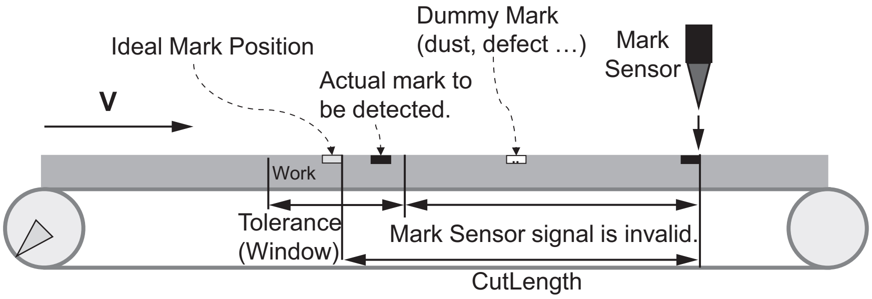
Tolerance Function

This section describes the Tolerance Function.

The Tolerance Function can set the valid mark detection area (window) by the tolerance. With this function, detection error such as sensor-output chattering and unexpected detection by the mark sensor can be avoided. Tolerance should be input into *CuttingCfg.Tolerance*. Details of the Tolerance Function are as follows.

- The variable is valid only when *CuttingCfg.Mode* value is 1 (Mark to Mark) or 2 (External Mark To Mark).
- In the *CuttingCfg.Tolerance*, set a permissive length to allow fluctuations in actual mark positions.
- The setting unit is [%].
- For example, when you set the value “5”, the valid range(window) of mark detection is CutLength±5% x CutLength shown in the following picture.
- If the mark detection doesn’t occur in the valid range (window), the blade will cut the product at the length of *CuttingCfg.CutLength*.
- When a mark detection occurs out of the valid range (window), this detection is ignored (masked).
- When the value of *CuttingCfg.Tolerance* is set “0”, the FB regards as “25.0” (Initial value).
- The value of *CuttingCfg.Tolerance* can be changed during execution.

The following figure shows the Tolerance Function. With the Tolerance, the valid range of mark detection can be set.



Additional Information

- This function will not work correctly when the distances between marks differ greater than the value set with *CuttingCfg.Tolerance*.
- The combinations of *CuttingCfg.CutLength* and *CuttingCfg.Tolerance* settings and each process that is executed is shown in the following table.

Case	CutLength	Tolerance	where to cut
1	Not 0	Not 0	Cut at mark when mark is in tolerance area.
2	Not 0	0	Tolerance is regarded as 25.0.
3	0	Not 0	Constant cut function is invalid.[*1]
4	0	0	Constant cut function is invalid.[*1]

[*1] the “Constant length cut function” is disabled.
For “Constant length cut function”, refer to *CutLength Function* on page 75.

Advance Angle Compensation Function

This section describes the Advance Angle Compensation Function.

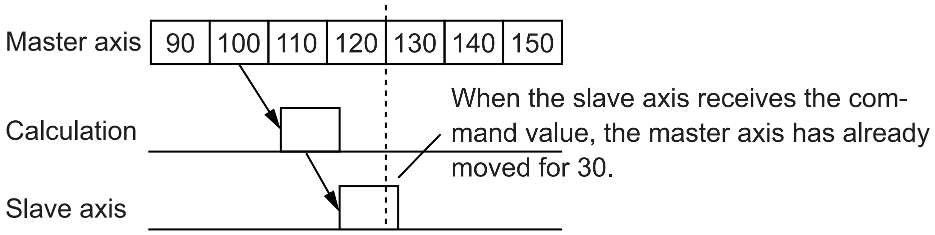
The axis position that is referred to in order to control the slave axis delays from the real time position for one communication cycle. Also, it takes time to calculate the slave axis command value based on the master axis position, and to give the command value to actuators such as servo driver. As a result, the master axis has been moved ahead further when the slave axis command value which is calculated by this FB is reflected on the operation.
To compensate the delays, Advance Angle Compensation adds the expected delay to the master axis.

- To enable the advance angle compensation function, set *MasterCfg.EnableAdvanceAngleCorr* for TRUE.
- The compensation value is input to *MasterCfg.AdvanceAngleCorrValue*. AdvanceAngleCorrValue enables the value to change while this FB is operating.
- By adding this compensation value to the Master axis position (i.e., conducting advance angle compensation), this FB calculates the Slave Axis Command.

When the numerical value, which is greater than the fivefold of difference of the master axis position between the previous cycle and the current cycle, is input, this input numerical value is treated as the fivefold, and then the advance angle compensation function is performed. The reason why cutting the exceeded value down to the fivefold value is to prevent calculating the massive slave axis command due to setting mistake.

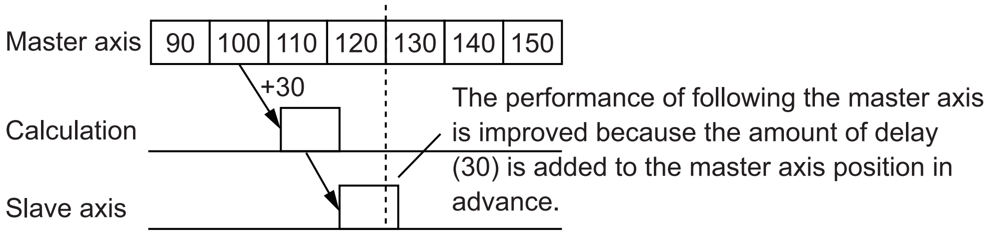
● Compensation Image by the Advance Angle Compensation Function

When Advance Angle Compensation Function not used:



When the slave axis receives the command value, the master axis has already moved for 30.

When Advance Angle Compensation Function used:



The performance of following the master axis is improved because the amount of delay (30) is added to the master axis position in advance.

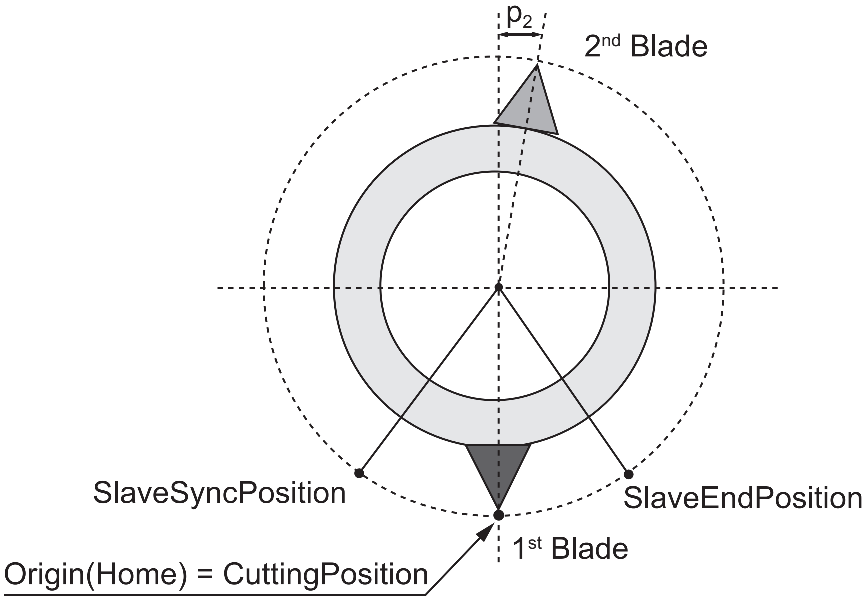
Blade Mount Error Compensation Function

This section describes the Blade Mount Error Compensation Function.

When the 1st blade is at the home position (position 0) of the slave axis (position 0), and there are two or more blades, this FB calculates the slave axis command value by expecting that blades are placed at equal intervals within modulo maximum positions.
The function adds the blade mount error compensation value to the slave axis command value that is calculated based on the ideal blade position, and then gives the command to the slave axis.
When the blade is mounted at the rotation direction away from the ideal position, the blade mount error compensation is specified with a positive value. When the blade is mounted at the counter rotation direction from the ideal position, the compensation is specified with a negative value.

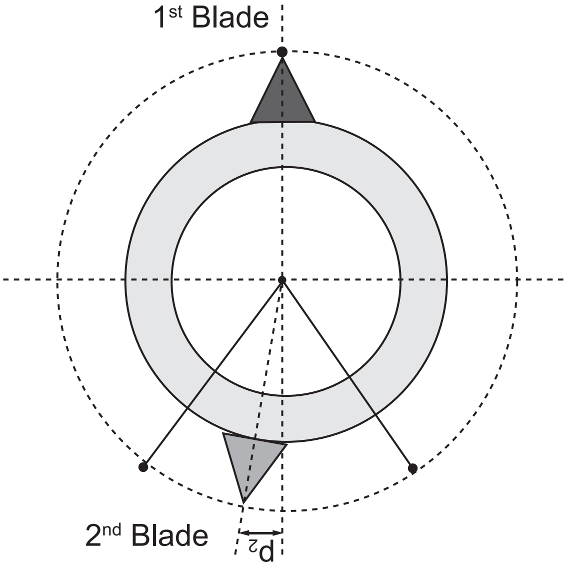
When two blades are mounted at the off-positions as the following figure shows, the values in the following table are set for BladMountErrorComp.

	1st Blade	2nd Blade	3rd Blade	4th Blade
	[1]	[2]	[3]	[4]
BladeMountErrorComp	0.0	$-p_2$	0.0	0.0



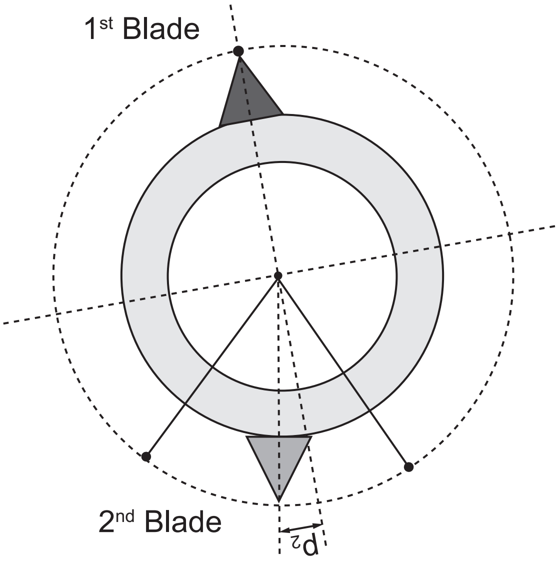
- **The timing of the *Cutting* output variable's turning ON when 2nd blade is displaced**

  As shown in the following figure, *Cutting* changes to TRUE when the 2nd blade is not at the cutting position.



- **The timing of the *Cutting* output variable's turning ON when the 2nd Blade Mount Error Compensation is performed**

  As shown in the following figure, *Cutting* changes to TRUE when the 2nd blade is at the cutting position.



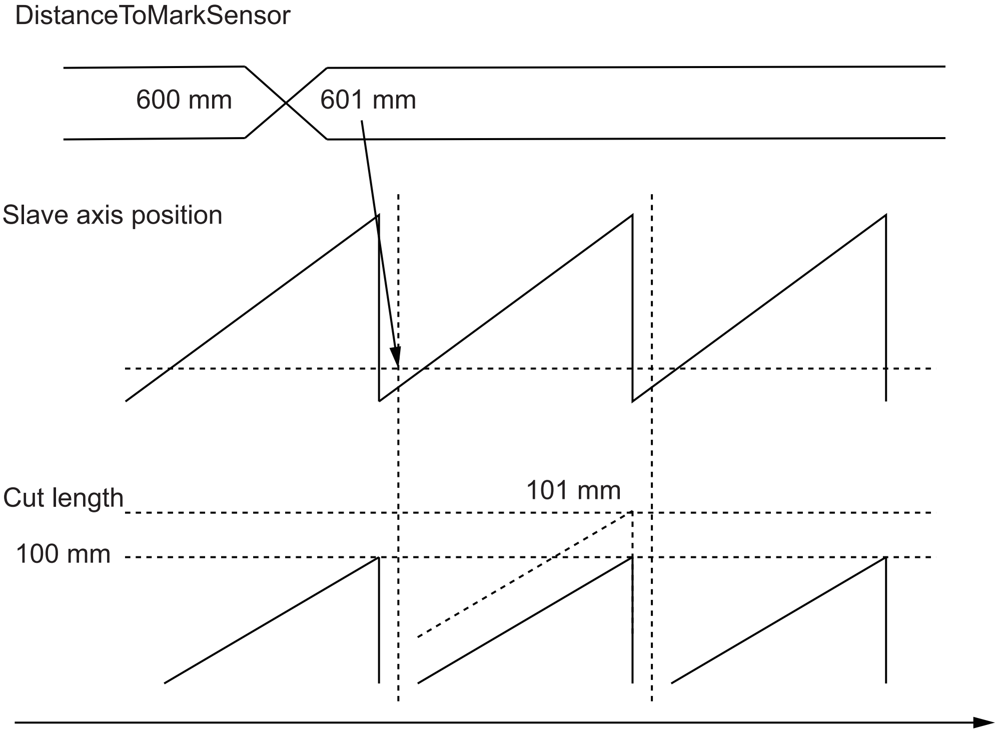
How to Adjust DistanceToMarkSensor during operation

This section describes how to adjust DistanceToMarkSensor while it is running.

After *CuttingCfg.DistanceToMarkSensor* is adjusted, the difference of the distances, between before and after the adjustment, is reflected on the next cut length. By adjusting *CuttingCfg.DistanceToMarkSensor* during operation, it is possible to compensate the mount error caused by the incorrect value that is set in the Mark To Mark mode or the External Mark To Mark mode.

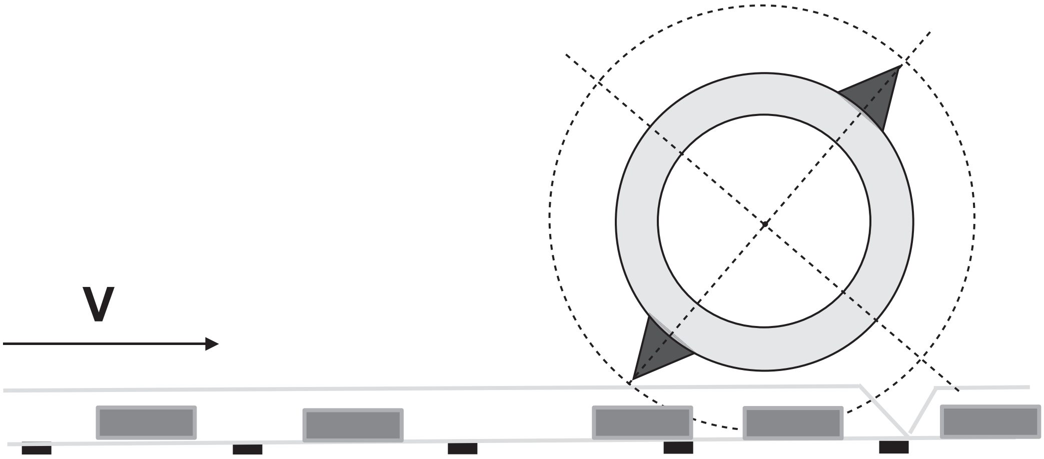
For example, 600mm is set for *CuttingCfg.DistanceToMarkSensor* when the distance between the actual cutting position and the mark sensor is 601mm. In this case, 1mm before the mark sensor is cut continuously. To simplify, the interval of the marks is 100mm.
To compensate this error, the value is adjusted to 601mm while this FB is operating.
When the next cut length is calculated, the interval of the marks (100mm) + the adjusted length (1mm) = 101mm becomes the cut length. By doing so, 1mm is compensated. The next cut length is the interval of the marks (100mm).

● The timing of DistanceToMarkSensor update (for single blade)



Prevention against Bite and Procedure after the Bite

After this FB is used in the packaging machine, the blade of the rotary knife seals and cuts a packaging film. When a product to be packaged blocks a sealing position or cutting position, the product gets caught by the blade. This phenomenon is called bite. The third product from the right in the diagram blocks the next cutting position, thus the rotary cutter will bite the product is it continues cutting operation.



To prevent and get rid of bite, take the following procedures.

First, use the NoGapNoSeal function in order to lower the frequency of bite. For details, refer to *NoGapNoSeal* on page 82.

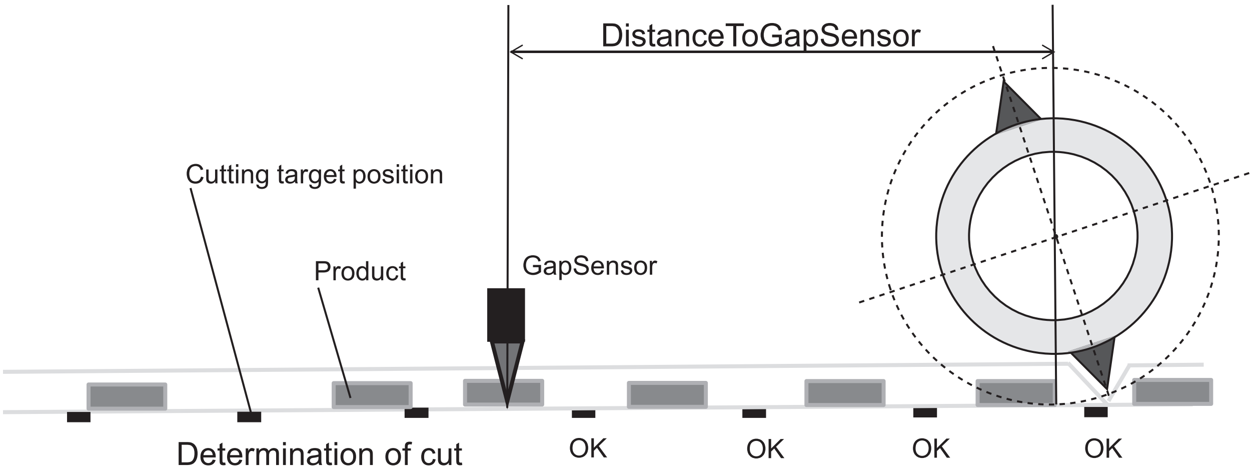
When a bite happens, remove the bitten packaging film and product and then resume packaging.

1. Use the Stop Blade (immediate stop) function to stop the rotary knife (slave axis), and use the external motion control command to stop forwarding film (master axis).
   For details, refer to *Stop Blade* on page 88.
2. After the blade stops (*Stopped*:TRUE), change Input *Enable* and *CoupleOn* of the FB to FALSE.
3. To remove the bitten film and product, move the rotary knife (slave axis) by using an external motion control command.
4. After removing the bitten film and product, change Input *Enable* to TRUE, and move the blade to *WaitPos*.
   Operation and behavior of this FB after the removal will be the same as the initial operation.

NoGapNoSeal

● Overview

The GapSensor, which is installed at DistanceToGapSensor from the cutting position, detects products and determines whether the products can be cut at the Cutting target position. This function is enabled in any operating mode.



To use this function, input the following values in the FB.

Variable Name	Name	Usage
EnableNGNS	Enable NoGapNoSeal function	Specify the use of NoGapNoSeal function
GapSensorSignal	The output signal of the gap sensor	Based on the signal, judge existence of products before and after a cutting target position.
DistanceToGapSensor	The cutting position is set from the gap sensor position.	Calculate time when the cutting target position passes under the Gap sensor.
WindowForGap	Sets in display unit of the MasterAxis based on the cutting target position.	Set detection range of products based on the cutting target position.
VelocityToWaitPos2	Set velocity at which the blade is moved to the WaitPos at the time of cutting indefinite.	Specify velocity at which the blade is moved to WaitPos at the time of cutting indefinite.

To enable this function, input *CuttingCfg.CutLength* or a positive number, which is double of the mark interval or greater, into *CuttingCfg.DistanceToGapSensor*. In the above diagram, the GapSensor is mounted at 3.5 times away of the mark interval.

As shown above, at the time of the blade which completed cutting reaches the *EndPos*, when the determination of cut for the next cutting target position is valid, the next cutting target position is determined as the next cutting target. If the Determination of cut for the next cutting target position is invalid, the blade which executes cutting next moves to the *WaitPos* in an *VelocityToWaitpos2* to the *WaitPos* when the cutting position is indefinite. In the above diagram, the second cutting position from the right is the next cutting position.

● Detection range setting

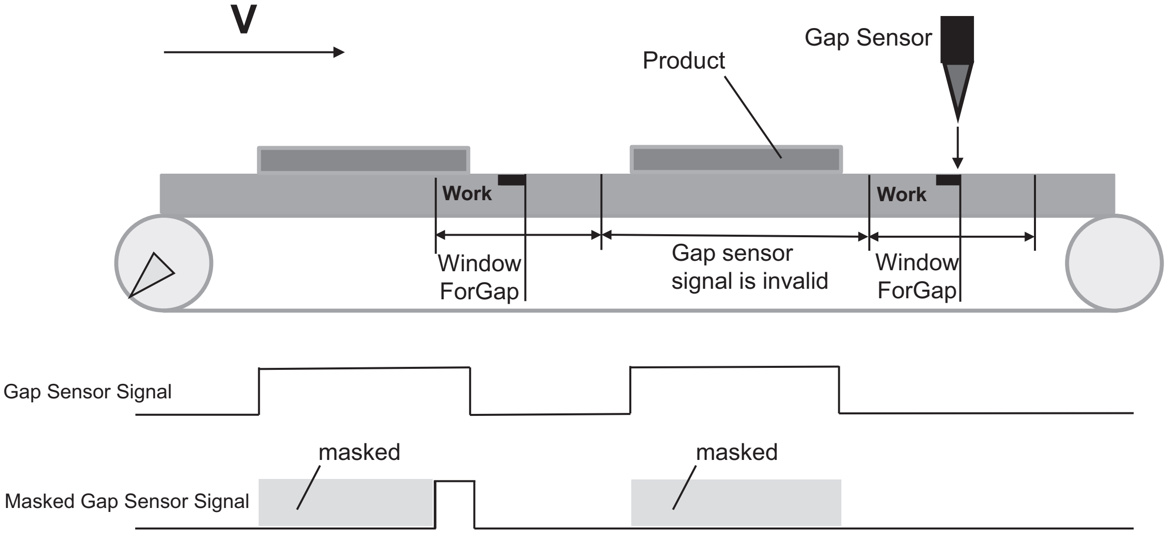
Detection range setting sets the range to enable the Product detected by *CuttingCfg.WindowForGap*.

By the detection range setting, the range, which monitors signal output by the gap sensor when NoGapNoSeal function is enabled, can be limited to before and after the cutting target position.

The following lists the details of the detection range setting.

- It is possible to enable any of the *CuttingCfg.Mode*.
- *CuttingCfg.OffsetFromMark* will make the detection and judgment with respect to the cutting position that has been reflected.
- *CuttingCfg.WindowForGap* to set the range to monitor the position of Product based on the cutting position.
- Setting unit is the display unit of the MasterAxis. For example, if the setting value is 5, CutLength ± 2.5, as shown in the figure below is the valid range of the product detection.
- *GapSensorSignal* for the product is masked and it will be used in the determination of the cut as MaskedGapSensorSignal.
- *CuttingCfg.WindowForGap* can be changed during the FB execution. The changes are immediately reflected in the cutting target position that does not pass through the Gap Sensor.

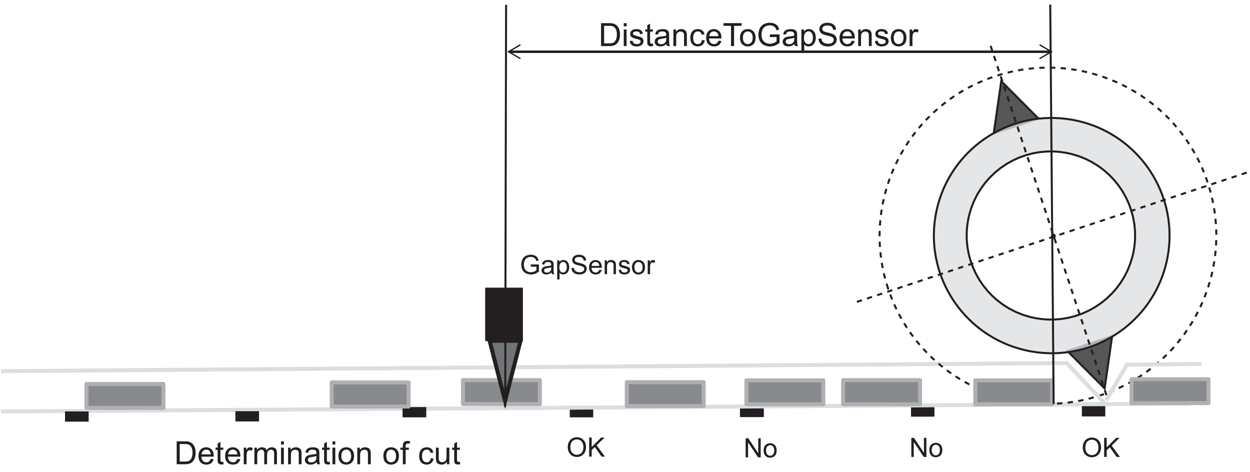
The following diagram shows the detection range setting. It indicates that you can set the range to enable the detection of the product by *CuttingCfg.WindowForGap*.



Skip

As shown in the figure below, when two of the cutting position in the cutting determination is determined to be invalid the blade moves at the speed of *VelocityToWaitPos2* to the *WaitPos* as the next cut position indefinite. After the completion of the movement to the *WaitPos*, it skips two cutting target positions where cutting is not possible, and then restart the cutting operation for the third cuttable cutting position (the fourth cutting target position from the right).

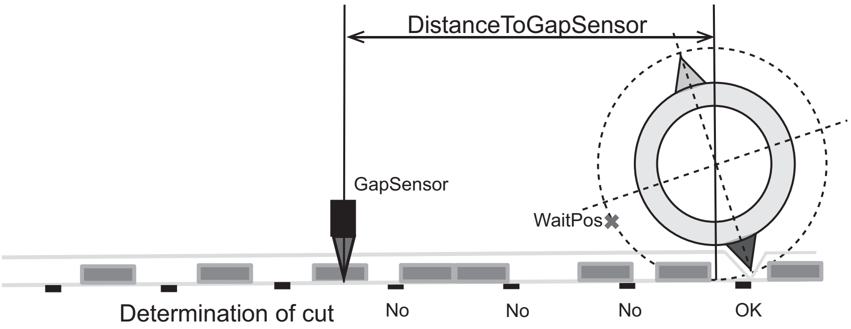
As a result, the packaging which is three times longer than normal packaging in performed.



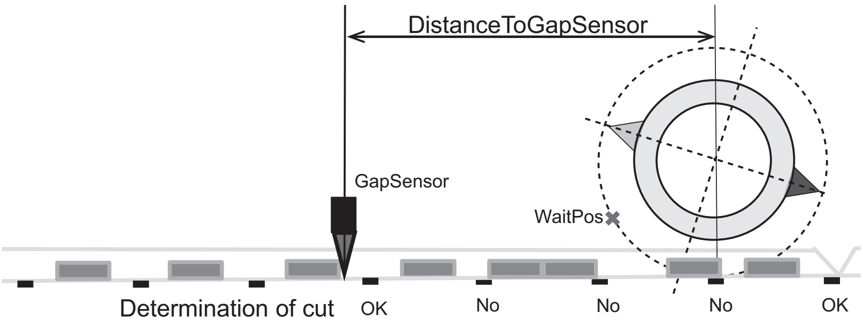
All Cutting Target Positions Are Not Cuttable

When the judged cutting target position are all determined that the cutting is not possible, the operation is as follows:

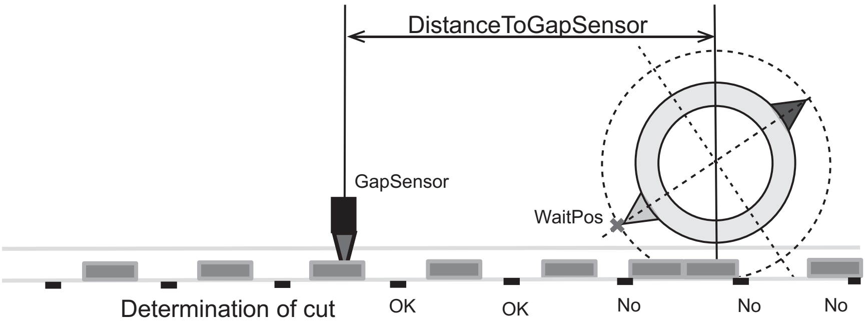
1 Once the blade reaches the *EndPos*, and all cutting target positions are determined not to cut, the next cutting position is not fixed.



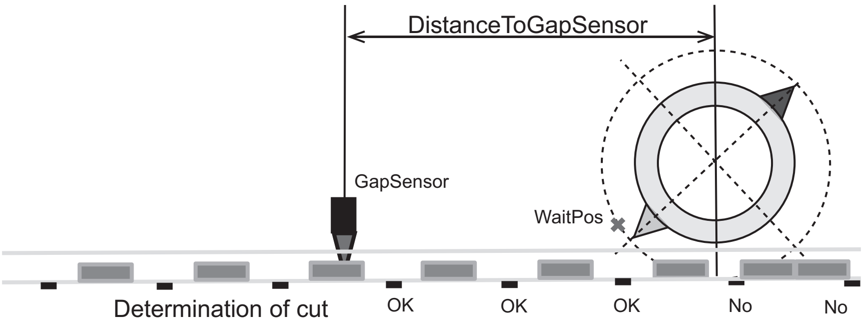
2 The blade which movement is shortest in the forward rotation moves to *WaitPos* at *VelocityToWaitPos2* velocity. While the blade is moving to *WaitPos*, the cutting target positions are updated and *Determination of Cut* is in operation, but cutting is suspended.



3 After the blade is moved to *WaitPos*, the next cutting position is fixed out of the cutting targets.



4 Once the next cutting position is fixed, *InvalidCuttingPos* becomes FALSE. Resume the cutting operation from the next cutting positions.



● Restrictions

- Even though the blade has been moved to *WaitPos* and the next cutting position is fixed, calculation sometimes reveals that the blade cannot reach the synchronous velocity between *WaitPos* and *SyncPos* at the designated velocity. In this case, the FB causes the error stop. When the master axis velocity is faster than the beginning, this error occurs.

Error Contents	Error Code	
	Profile Type 0: Trapezoid	**Profile Type 5: Fifth-order curve**
Insufficient travel distance at first execution	16#00800003	16#00850003
Too long travel distance at first execution	16#00800004	16#00850004
Maximum velocity exceeded	16#00800005	16#00850005

- In the Continuous mode (Mode=0), when *CutLength* is changed dynamically, the function wrongly determines the cutting target positions, and that causes bite. When this function is enabled, do not change *CutLength*.

- If you dynamically change the *OffsetFromMark* when this function is enabled, an erroneous determination is made. A result, bite occurs. Therefore, when this function is enabled, do not change *OffsetFromMark*.

Continuous Operation

Overview

This function keeps the cutting operation of the rotary knife by using the retained variable even when power is OFF or the mode is changed.

To use this function, the master and slave axes need to use the Servo motor having an absolute encoder, or an absolute encoder.

The function uses the following variables.

These variables must be the retained variables in order to retain data.

Variable name	Type	Name	Usage
ContinuousOpe.Store	BOOL	Store Continuous Operation Data	Store continuous operation data in *ContinuousOpe.Data*.
ContinuousOpe.Stored	BOOL	Stored Continuous Operation Data	Show that continuous operation data is stored.
ContinuousOpe.ValidData	BOOL	Valid Continuous Operation Data	Show continuous operation data is valid for the FB.
ContinuousOpe.Data	sCONTINUOUS_DATA	Continuous Operation Data	Continuous Operation Data

Save, Resume

How to save and how to resume are as follows:

- How to Save

1. Stop the master and slave axes.
   Before saving continuous operation data, stop the master axis and the synchronized slave axes. Keep TRUE for *Enable* and *CoupleOn* of this FB.
2. Save continuous operation data.
   Set *ContinuousOpe.Store* is TRUE. Start storing continuous operation data.
3. Has saved continuous operation data.
   Check *ContinuousOpe.Stored* to see whether continuous operation data has been stored.
4. Turn the power OFF or change the mode.
   Turn the power off or change the mode.

- How to Resume

1. Enable continuous operation data.
   Set TRUE to *ContinuousOpe.ValidDat* in order to enable continuous operation data.
2. Move the blade to the waiting position.
   Input TRUE in *Enable* of the FB in order to move the blade to *WaitPos*. The blade that is the most closest to *WaitPos* in the forward rotation moves to *WaitPos*.
3. Start the continuous operation.
   Rotate the master axis and input TRUE in *CoupleON* of the FB.
4. Determine the cutting position.
   Determine the cutting position based on the saved cutting target positions, and then start the cutting operation. When NoGapNoSeal is enabled, determine the cutting position out of the determined cutting target positions based on the determination results, and then start the cutting operation.

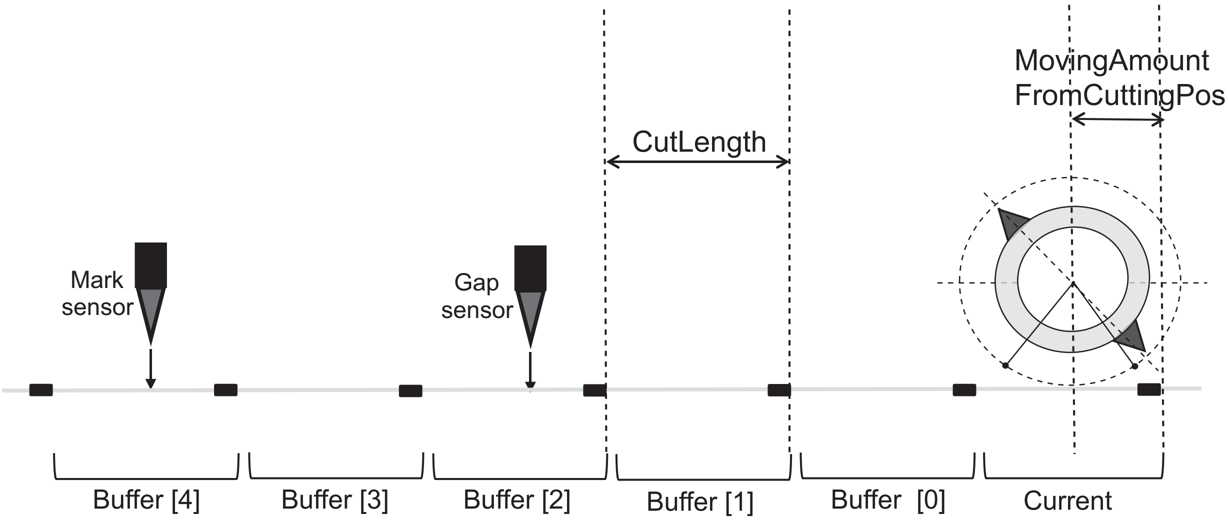
Continuous Operation Data

Data required for continuous operation is retained by external retained variables.

The following shows the details of continuous operation data. Continuous operation data at the time of continuous operation start is checked, and if the incorrect value is entered, this FB ends in error.

Besides this FB, it is possible to run the continuous operation by the values set by user programs.

When the operation continues in the state shown in the following figure, cutting operation is executed for the third Mark from the right. The mark positioned at a downstream side of the *Current* is ignored. At that time, *CutLength* which is output by this FB becomes *Current*+*Buffer[0]*.



sCONTINUOUS_DATA:

Member name	Type	Name	Usage
MasterAxisPosition	LREAL	Master axis position	Specify the master axis position when data is stored.
MovingAmount FromCuttingPos	LREAL	Master axis moving amount from the previous cut.	Specify the travel distance of the master axis between the previous cutting position and the next cutting position.
Current	sCUTTING_DATA	Data of products to be cut when data is stored	Data of product to be cut when continuous operation data is stored.
Buffer	ARRAY[0..63] OF sCUTTING_DATA	Buffer data	Buffered cut data

sCUTTING_DATA:

Member name	Type	Name	Usage
CutLength	LREAL	Cut length	Specify the cut length of product.
State	UINT	Product detection state	Specify the state of product detection at a cutting position. 0: none 1: before conducting NoGapNoSeal 2: NoGapNoSeal checked and no product 3: NoGapNoSeal checked and product exists

Restrictions

- After saving continuous operation data by using this function, do not move the master axis.

  If you carry out the continuous operation after moving the master axis and saving data, the slave axes suddenly start moving.
- When the continuous operation is resumed, the blade which has the shortest travel distance in the forward rotation will move to *WaitPos*. On the way to *WaitPos*, other blade passes onto the cutting positions. To avoid cut and bite, adjust the initial positions of each blade.
- When resuming the operation, use *MasterCfg* and *SlaveCfg*, *CuttingCfg* with which you saved continuous operation data. At the time of continuous operation, if each setting is changed, there is a risk of unintended behavior.

Precautions for Correct Use

When resuming the operation, use *MasterCfg* and *SlaveCfg*, *CuttingCfg* with which you saved continuous operation data. At the time of continuous operation, if each setting has been changed, there is a risk of unintended behavior.

Stop Blade

Overview

This function stops the rotary knife in response to the specified Stop mode.

The operation is depending on the specified Stop mode and blade positions as the following table shows.

Once the blades have stopped, TRUE is output to *Stopped*.

Stop mode	Blade position	Operation
Immediate Stop	Any position	Once a Stop command is input, the slave axes stop immediately.
Deceleration Stop	Synchronized section	After moving to *EndPos*, the knife stops at the specified deceleration.
	The edge of the synchronized section where the blade enters when stopping at the specified deceleration.	Stop before entering the synchronized section at quicker declaration than the specified deceleration. The blade may stop as immediate as Immediate Stop depending on the blade position.
	Catching section	Stop at the specified deceleration.

This function uses the following variables.

Variable name	Type	Name	Usage
SlaveCfg.Deceleration	LREAL	Deceleration	Specify the deceleration to stop.
SlaveCfg.StopMode	_eMC_STOP_MODE	Stop mode	Specify the Stop mode. If anything other than a deceleration stop or immediate stop has been set, it will decelerate to a stop.
ExecuteStop	BOOL	Stop command	Give the stop command to rotary knives in operation.
Stopped	BOOL	Stopped	Check whether or not the blades are stopped.

Restriction

- During deceleration stop instruction, when the current speed of the slave axis is in the negative value due to the reverse rotation of the master axis, immediately stop the blade.

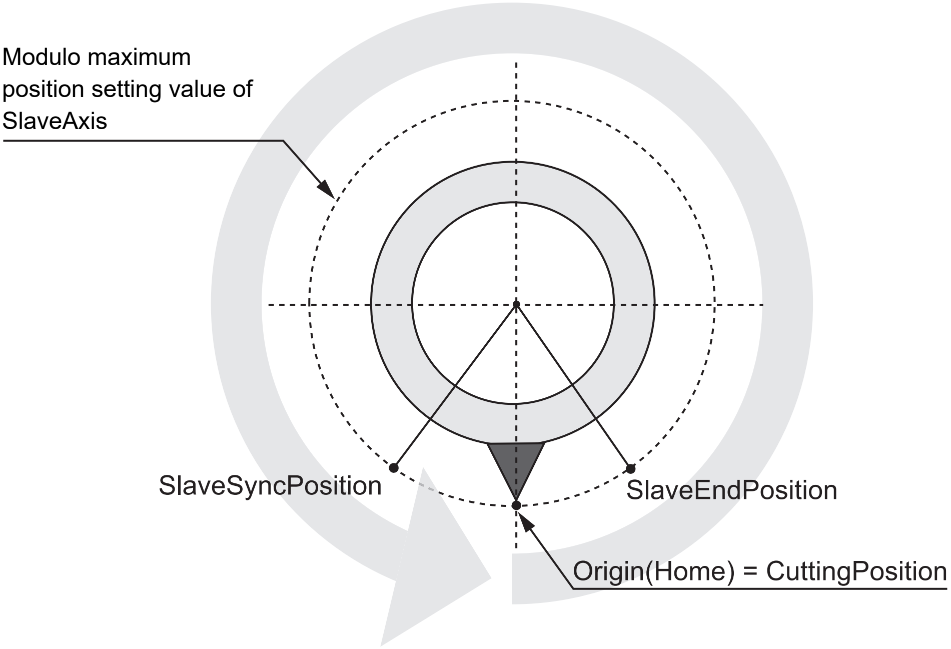
Setting the Input Variable

This section describes how to set the input variable.

This section describes settings of Modulo maximum position setting value of SlaveAxis, wait position of SlaveAxis, input trigger conditions, SlaveSyncPosition and SlaveEndPosition, and synchronization execution flag.

● How to set Modulo Maximum Position Setting Value

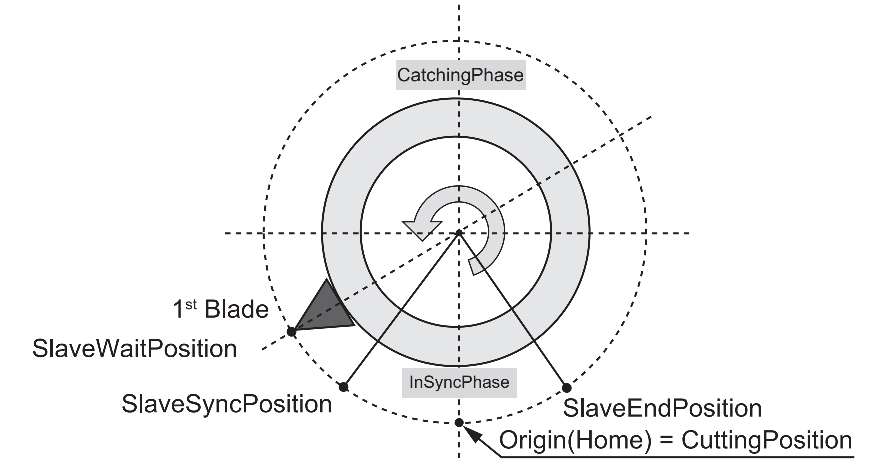
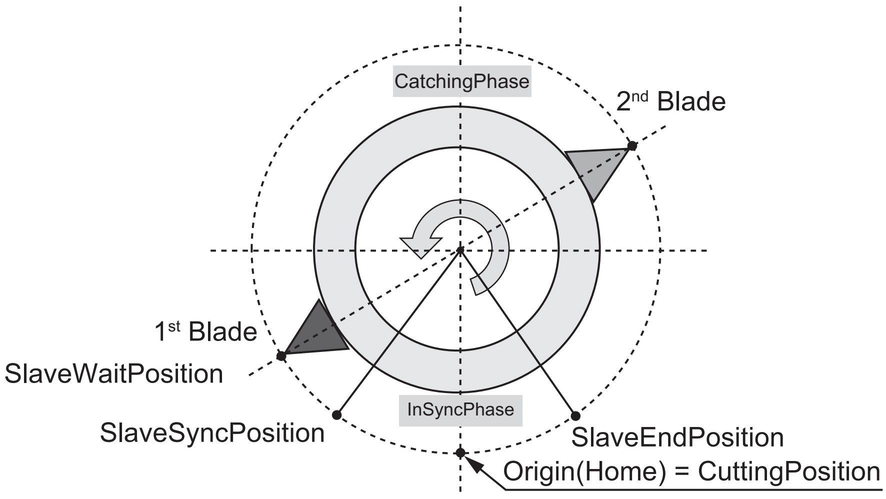
In the modulo maximum position setting value of SlaveAxis, set the circumference (perimeter of the rotary knife) on which the edge of the rotary knife traces. Refer to the following figure. (The dotted line indicates the circumference of the blade edge.)


Modulo maximum position setting value of SlaveAxis
SlaveSyncPosition
SlaveEndPosition
Origin(Home) = CuttingPosition

● Setting SlaveWaitPosition Input Variable

How to set SlaveWaitPosition of SlaveAxis:
The SlaveWaitPosition is input into *SlaveCfg.WaitPos*. When *Enable* is TRUE or the cycle has stopped, SlaveWaitPosition is a waiting position of the blade.
Set the value for the SlaveWaitPosition so that the following conditions are met.

- The set value must be within the CatchingPhase (between SlaveEndPosition and SlaveStartPosition).
- There must be a sufficient distance from the SlaveEndPosition to decelerate to a stop at a specified deceleration.

Figure	Description
 CatchingPhase 1st Blade SlaveWaitPosition SlaveSyncPosition InSyncPhase SlaveEndPosition Origin(Home) = CuttingPosition	When the number of blades is 1 In the SlaveWaitPosition, set the distance from the blade as shown in the figure on the left.
 CatchingPhase 2nd Blade 1st Blade SlaveWaitPosition SlaveSyncPosition InSyncPhase SlaveEndPosition Origin(Home) = CuttingPosition	When the number of blades is more than one. (The figure at left has two blades.) In the same way as when the number of blades is 1, set the wait position of the 1st blade in the SlaveWaitPosition as shown in the figure on the left. When this FB is executed, the blade nearest to the SlaveWaitPosition moves to the wait position.

● Setting TriggerInput

This variable is used only when *CuttingCfg.mode*=1(MarkToMark).
The following table shows the member variables of *_sTRIGGER_REF* structure variable.

Name	Meaning	Data type	Valld range	Function
Mode	Mode	eMC_TRIGGER_MODE	0: _mcDrive 1: _mcController	Specify the trigger mode. 0: Drive Mode 1: Controller Mode
LatchID	Latch ID Selection	eMC_TRIGGER_LATCH_ID	0: _mcLatch1 1: _mcLatch2	Specify which of the two latch functions to use. 0: Latch 1 1: Latch 2
InputDrive	Trigger Input Signal	eMC_TRIGGER_INPUT_DRIVE	0: _mcEncoderMark 1: _mcEXT	Specify the Servo Drive trigger signal to use in Drive Mode. 0: Z-phase signal 1: External input

When using this FB, specify a value for each member as follows:

Name	Set value	Description
Mode	0: _mcDrive 1: _mcController	Set the mode according to trigger mode.
LatchID	0: _mcLatch1 1: _mcLatch2	Specify a latch ID to use as a trigger when *Mode* is set to 0:_mcDrive.
InputDrive	1: _mcEXT	Set an external input.

For details, refer to the motion control instructions reference manual.

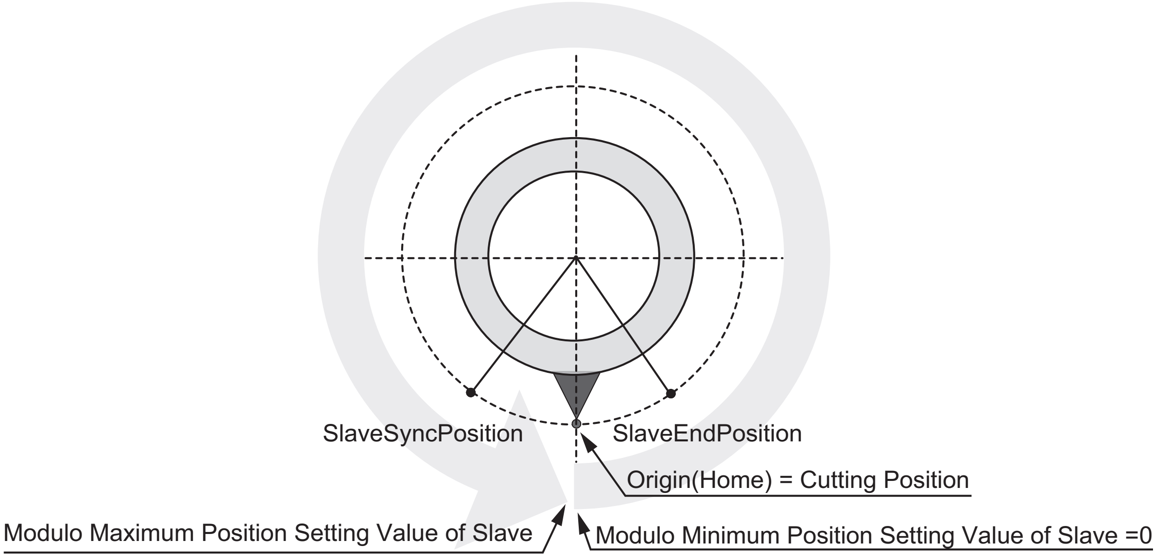
● Setting SlaveSyncPosition and SlaveEndPosition

This section describes the set values of the SlaveSyncPosition and SlaveEndPosition input variables.

Input SlaveSyncPosition for *SlaveCfg.SyncPos*, and input SlaveEndPosition for *SlaveCfg.EndPos* respectively.

The following relationship must be established when you set the SlaveSyncPosition, SlaveEndPosition, modulo maximum position setting value of SlaveAxis and modulo minimum position setting value of SlaveAxis.

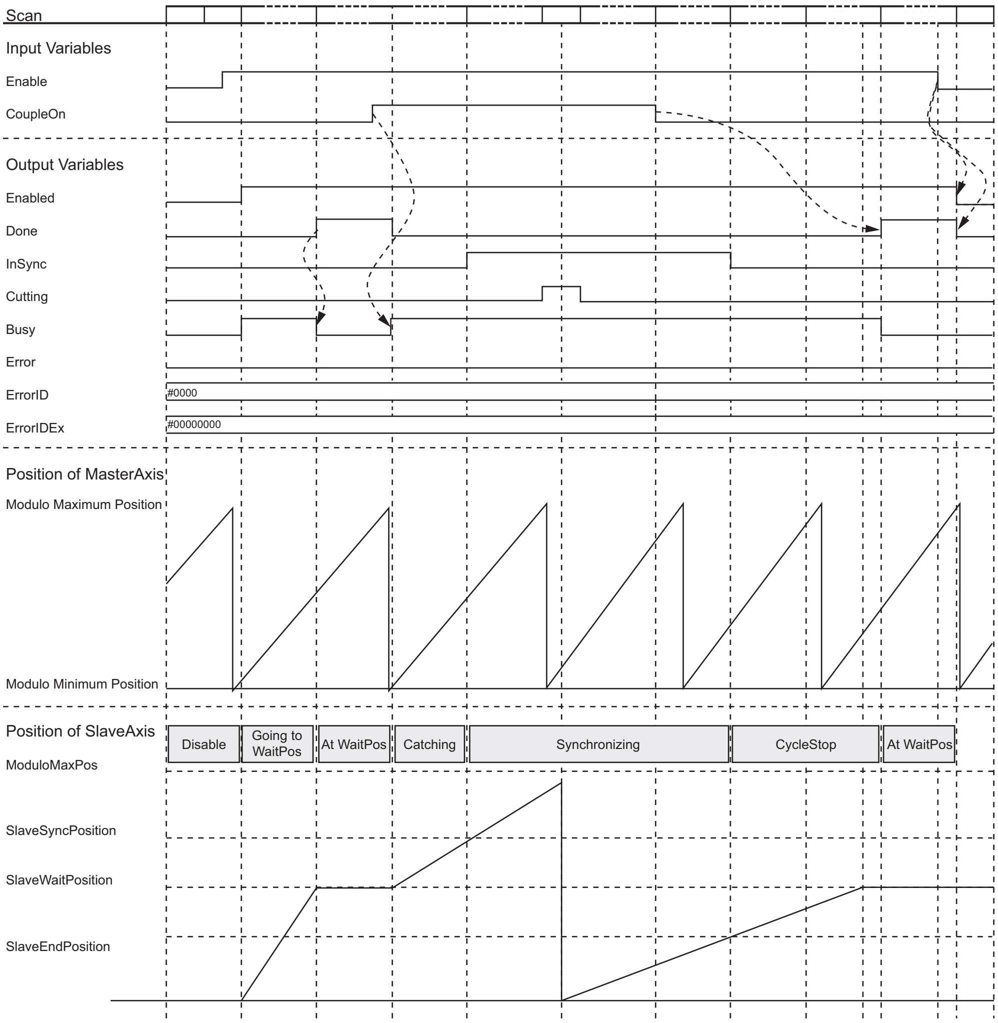
Modulo minimum position setting value of SlaveAxis (=0) < SlaveEndPosition < SlaveSyncPosition < Modulo maximum position setting value of SlaveAxis If the relationship above is not established, an error will occur.


SlaveSyncPosition
SlaveEndPosition
Origin(Home) = Cutting Position
Modulo Maximum Position Setting Value of Slave
Modulo Minimum Position Setting Value of Slave =0

Timing Charts

This section provides timing charts.

● Normal end

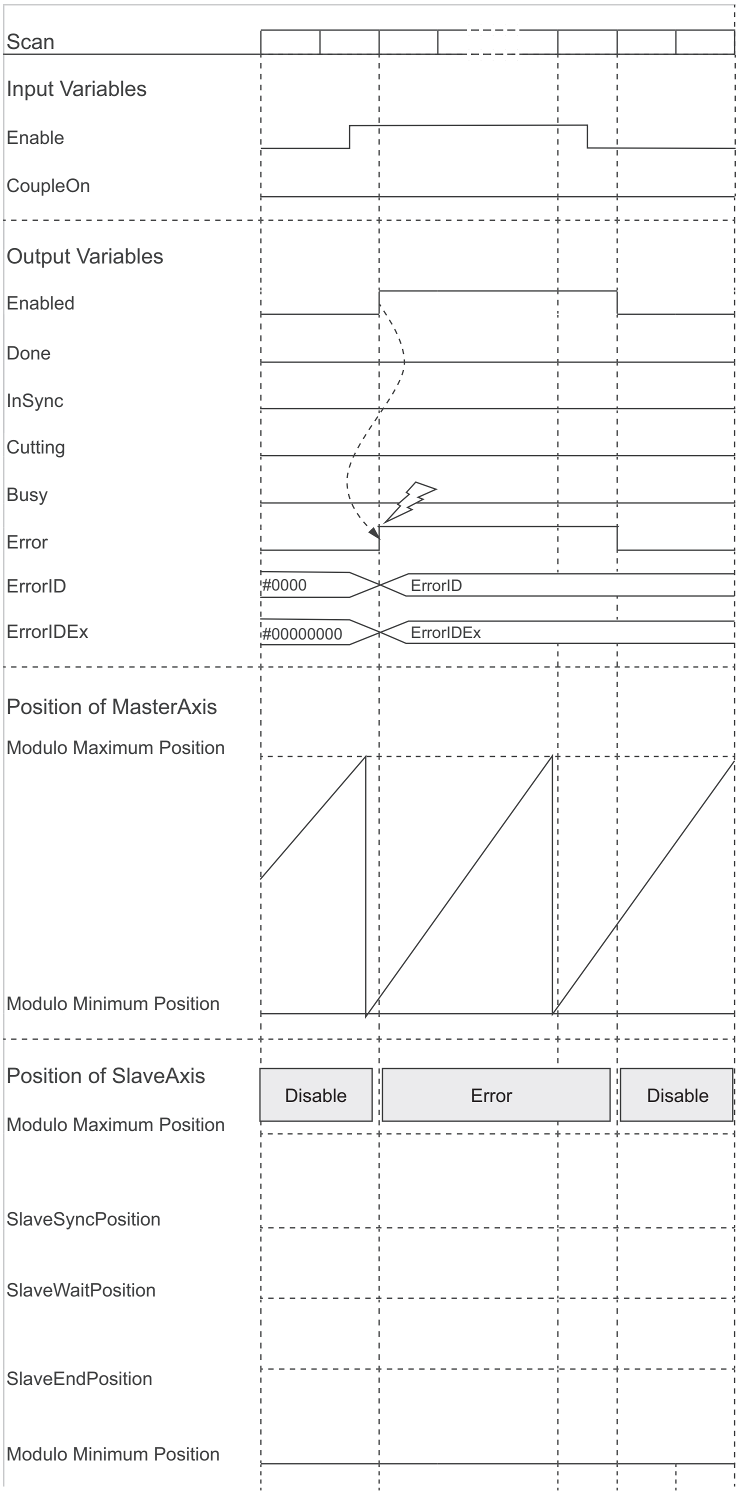


- When *Enable* changes to TRUE, the blade moves to the SlaveWaitPosition.
- When moving to the SlaveWaitPosition is completed, *Done* changes to TRUE.
- When *CoupleOn* changes to TRUE, the SlaveAxis starts synchronization with the MasterAxis.
- While operating in the InSyncPhase, *InSync* is TRUE.
- If the blade of the rotary knife passes the cutting position, *Cutting* is TRUE for one task period.
- When *CoupleOn* changes to FALSE, the blade performs a CycleStop considering the SlaveWait-Position as the stop position.
- After CycleStop, *Done* changes to TRUE.
- After CycleStop, *Done* keeps TRUE while *Enable* is TRUE.
- When you terminate operation of this FB, after *Done* changes to TRUE by CycleStop, change *Enable* from TRUE to FALSE.
- For re-starting FB, confirm *Enable* is FALSE and execute FB.
- The dotted lines of the Scan shown in the figure indicate multiple task periods.
- Refer to *Skip Function* on page 67 for the information of the condition and timing for *Skip* of output variable.
- Refer to *Buffer Function* on page 73 for the information of the condition and timing for *Skip* of output *BufferEmpty.*

For CycleStop, refer to *CycleStop Function* on page 71.

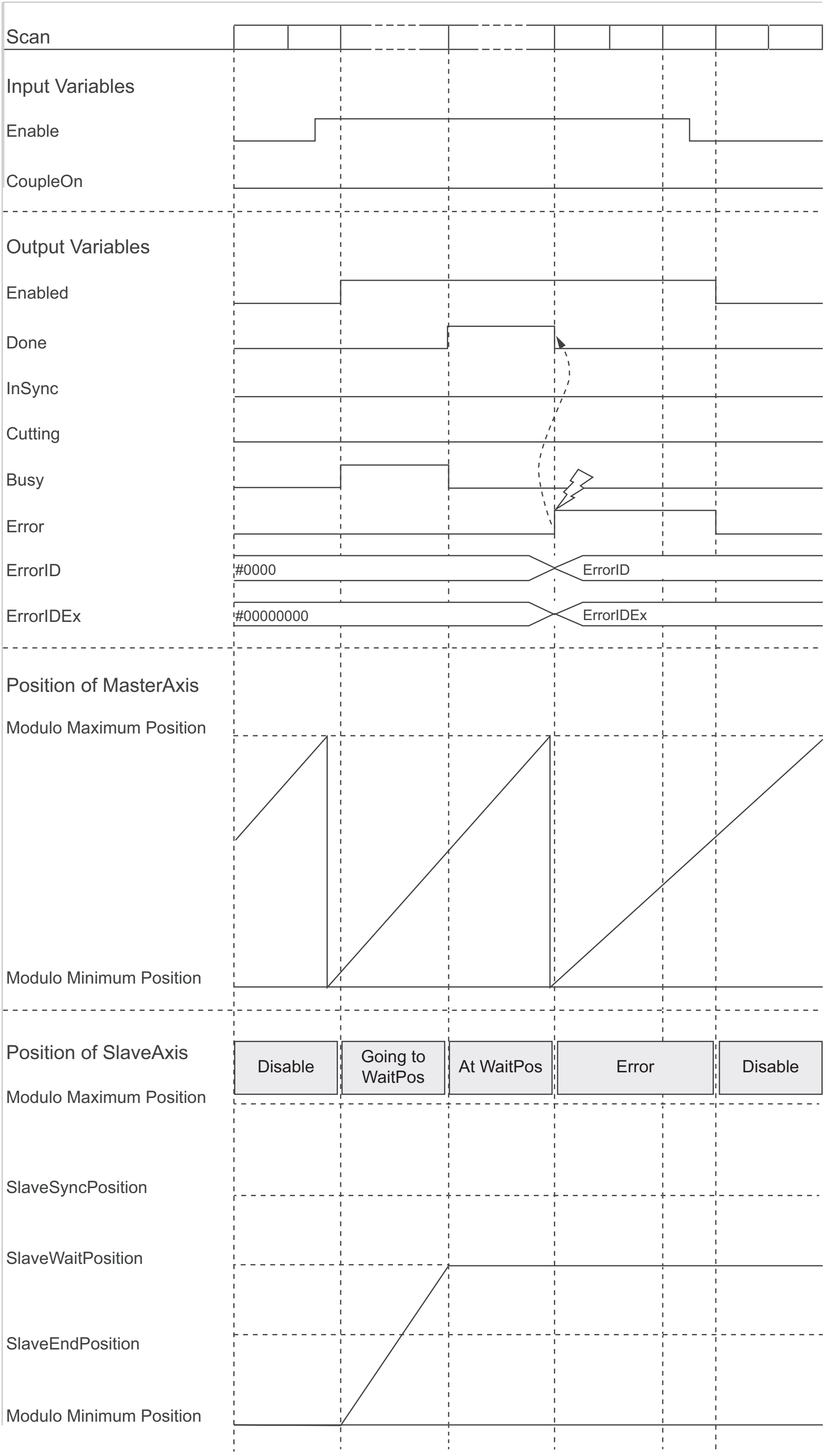
⚠ Caution	
Make sure that the SlaveAxis stopped before *Enable* changes to FALSE. When *Enable* changes to FALSE while this function block is operating or SlaveAxis is moving, it may stop immediately and cause property damage.	⚠

- **When an error occurs a function block an execution because the execution condition is not met**



- When the execution conditions are not met, this FB is not executed.
- While *Enable* of input variable is TRUE, *Enabled* of output variable is TRUE.
- The case that execution condition is not met is that it does not meet the conditions of *Execution Conditions of the FB* on page 44 and/or the value(s) of input variable is invalid.

● When SlaveAxis is at SlaveWaitPosition and *Done*=TRUE, error occurs

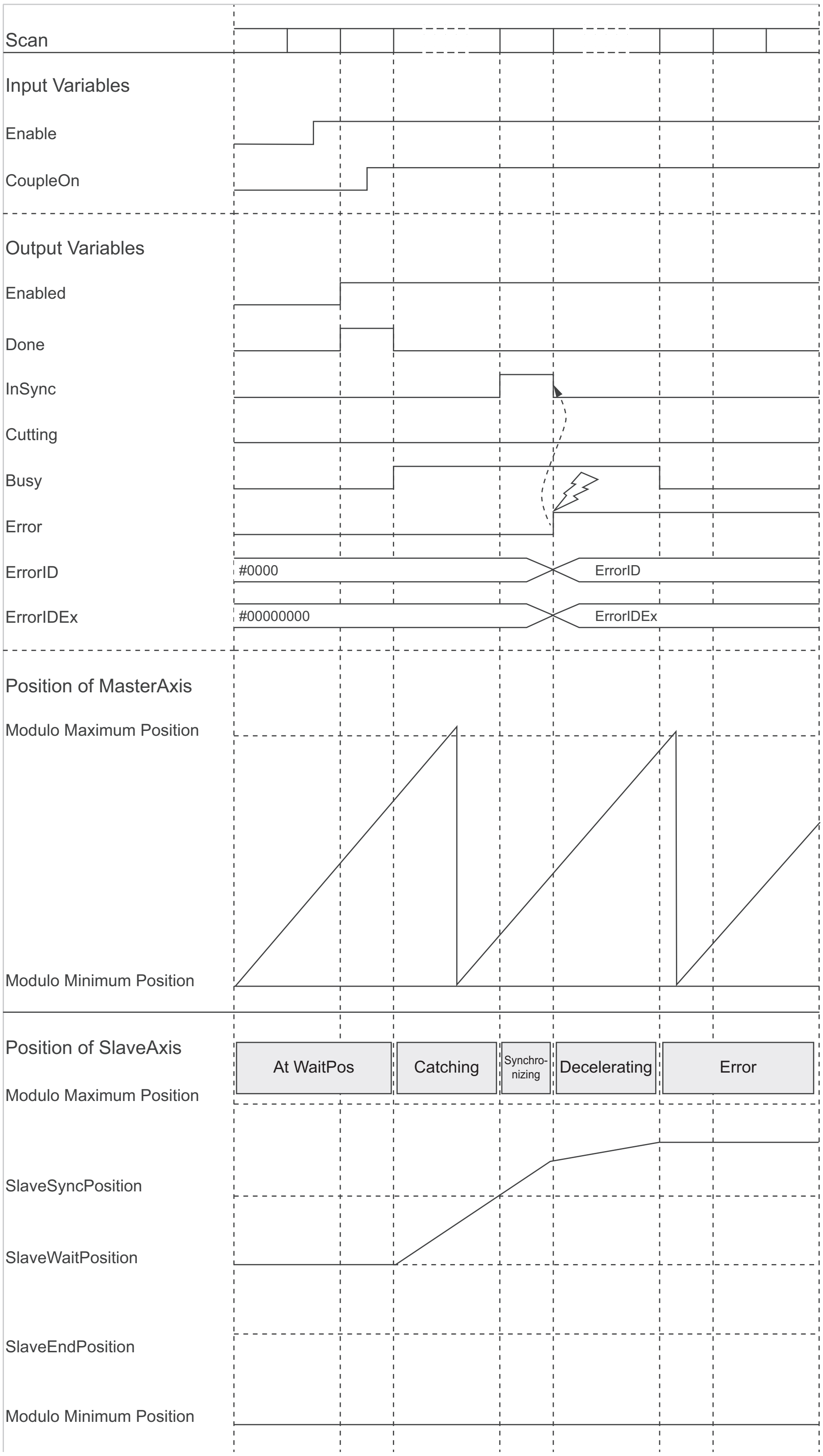


- When Blade is at WaitPsos and *Done*=TRUE, if *Error* changes to TRUE, *Done* changes to FALSE.
- When *Error* changes to TRUE during execution, the error codes are stored in *ErrorID* and *ErrorIDEx*.
- When *Error* changes to TRUE, the operation of FB ends.
- After clearing Error, *Enable* changes TRUE->FALSE->TRUE, the error code is cleared to zero.

- At Error, *Error* keeps TRUE, while *Enable* is TRUE.

The dotted lines of the Scan shown in the figure indicate multiple task periods.

● Under *Busy*=TRUE, error occurs

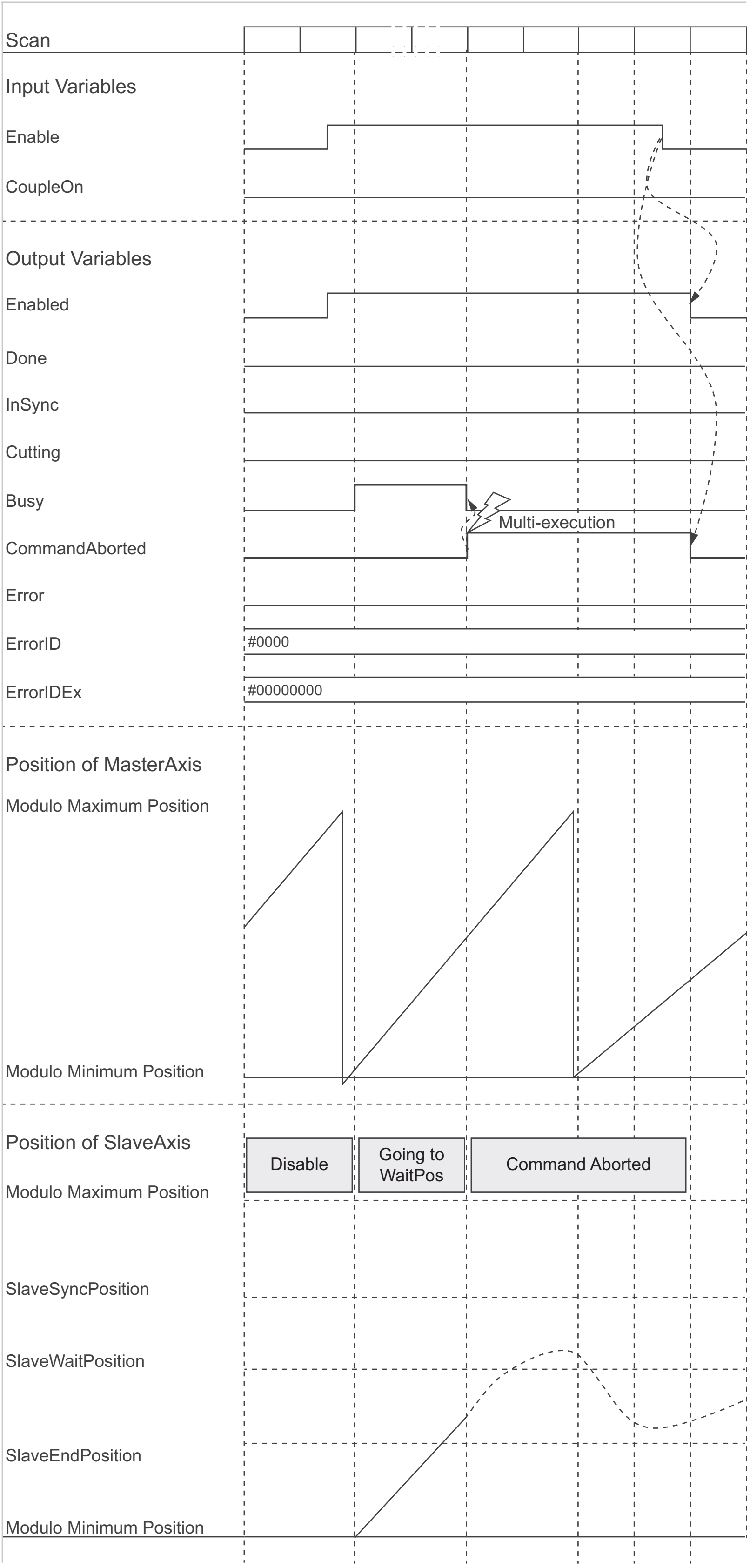


- If this FB occurs error when the SlaveAxis is moving of *Busy*=TRUE, the SlaveAxis decelerates to stop at the deceleration specified with *SlaveCfg.Deceleration*. After stopping, *Error* changes to TRUE.

- If Axis error occurs, the SlaveAxis will stop immediately or decelerate to a stop depending on the error.
- When *Error* changes to TRUE during execution, the error code(s) is stored in *ErrorID* and *ErrorIDEx*.
- After this FB is aborted, *Enable* internally changes to FALSE and the operation ends.
- When *Enable* changes TRUE->FALSE->TRUE after resetting an error, the error code is cleared to zero.
- At error, *Error* keeps TRUE while *Enable* is TRUE.
- The dotted lines of the *Scan* shown in the figure indicate multiple scans/task periods.

For the information of error code and cause of error, refer to *Troubleshooting* on page 111.

● While SlaveAxis is moving, other motion FB is executed

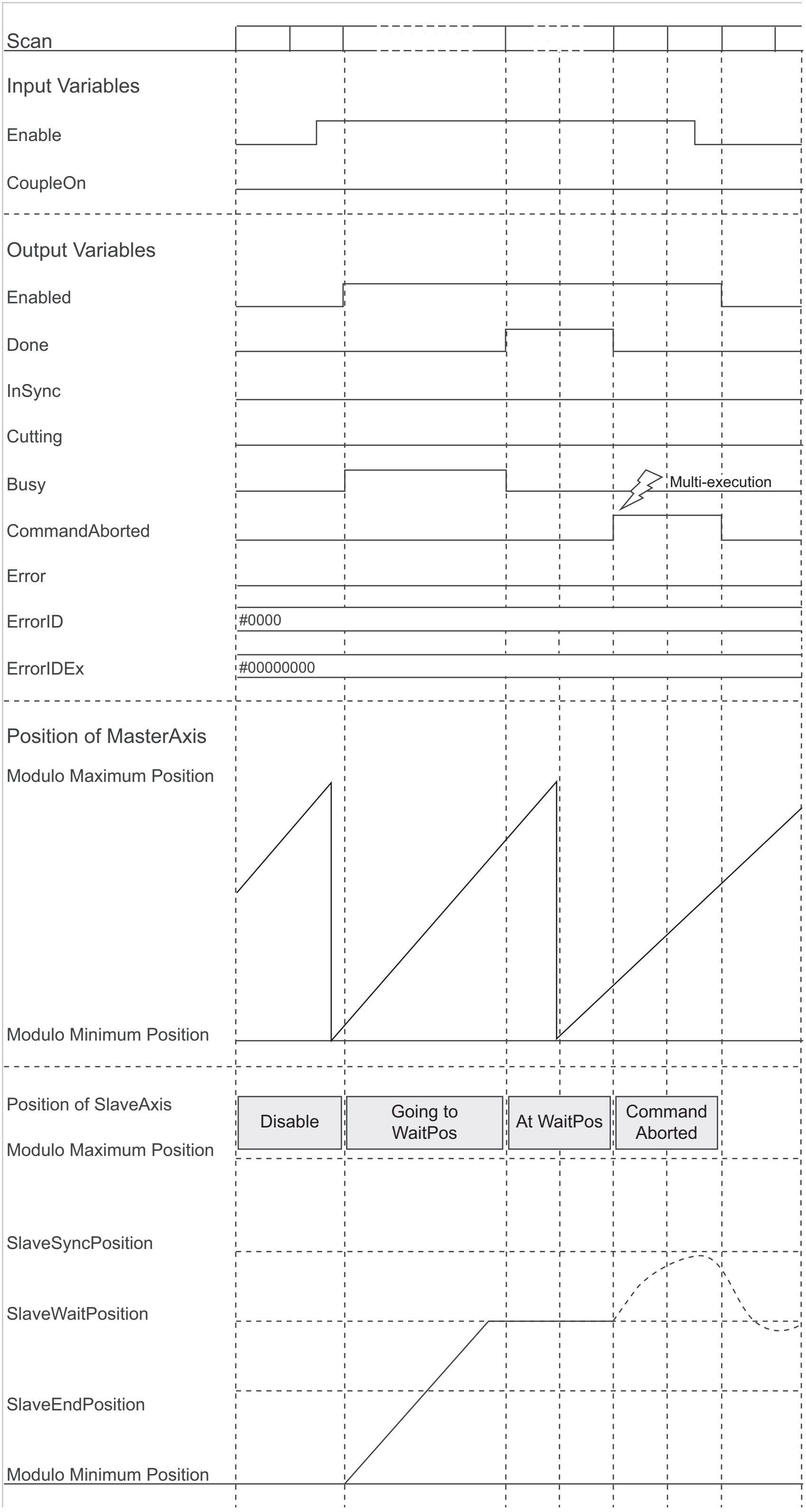


- When a multi-execution of another motion control FB is executed on the SlaveAxis during an execution of this FB, the execution of this FB is aborted.
- Multi-executed FB is executed.

- When the execution is aborted, *CommandAborted* changes to TRUE. This TRUE is kept while *Enable* is TRUE.
- After this FB is aborted, *Enable* internally changes to FALSE and the operation ends. Therefore, even if you keep *Enable* TRUE, the operation does not continue.
- The dotted lines of the Scan shown in the figure indicate multiple task periods.

⚠ **Caution**	
Do not execute other FB(s) for the SlaveAxis activated by this FB. Doing so may change the command to the SlaveAxis suddenly, and give excessive load to the device.	⚠

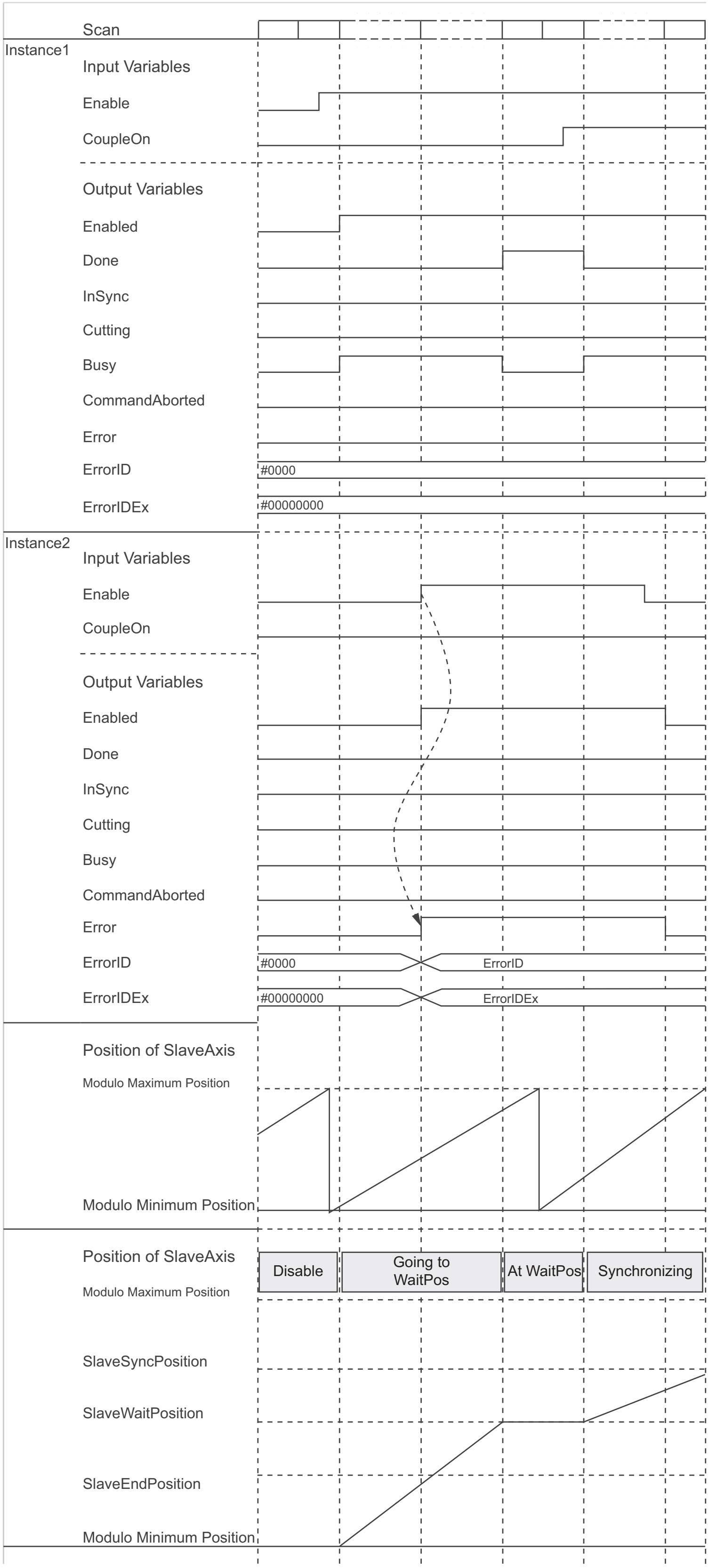
● While SlaveAxis is at SlaveWaitPosition (of *Done*=TRUE), other motion FB is executed



- When a multi-execution of other motion control FB is executed on the SlaveAxis during executing this FB of *Busy* being TRUE, the execution of this FB is aborted.
- Multi-executed FB is executed.
- After this FB is aborted, the operation ends.
  Therefore, even if you keep *Enable* TRUE, the operation does not continue.
- The dotted lines of the Scan shown in the figure indicate multiple task periods.

⚠ Caution	
Do not execute other FB(s) for the SlaveAxis activated by this FB. Doing so may change the command to the SlaveAxis suddenly, and give excessive load to the device.	⚠

- **While this FB executes and *Enable* is TRUE, another Instance of this FB is executed for the SlaveAxis**

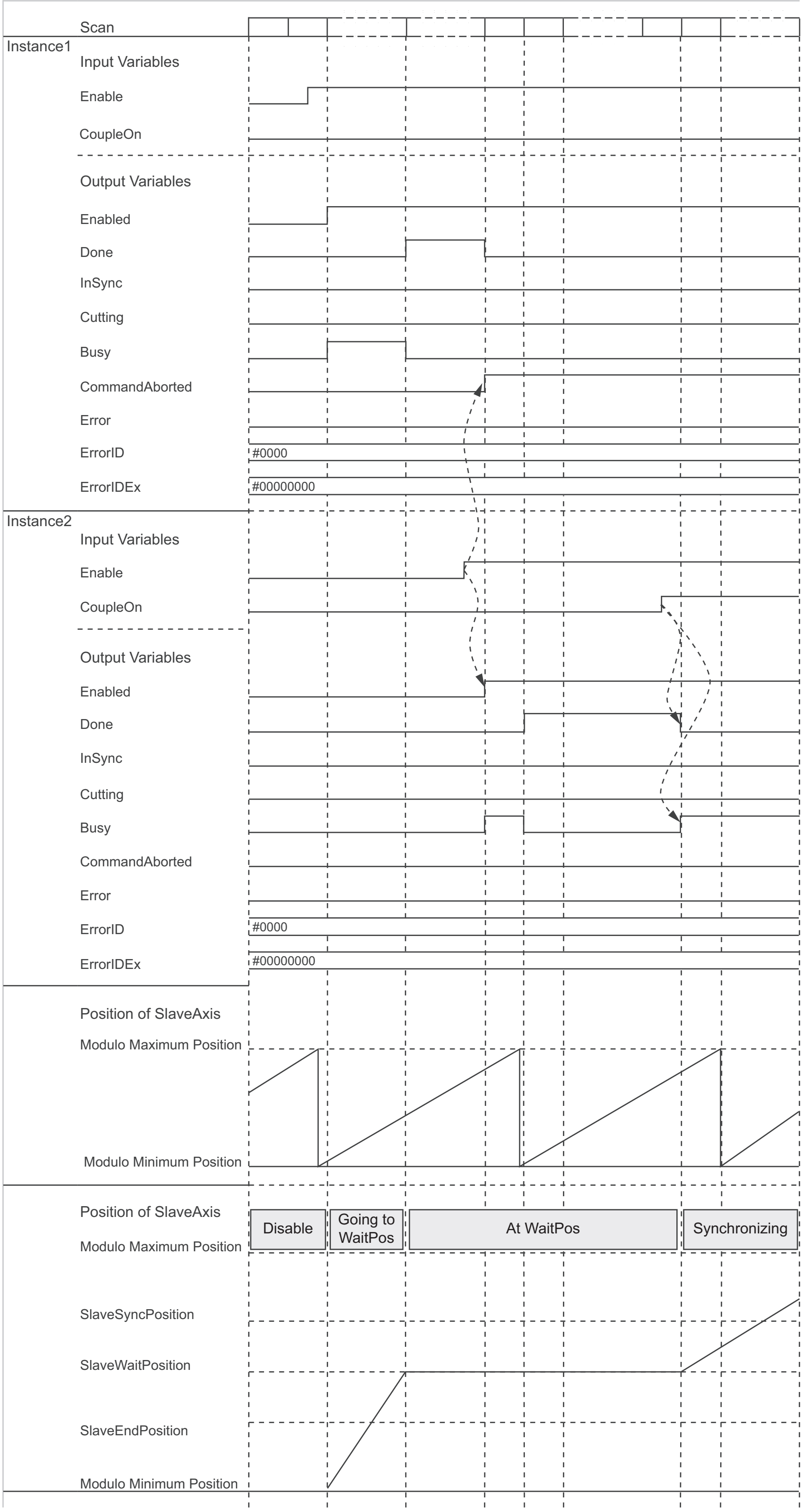


- Figure above shows the timing chart when RotaryKnife_Instance2 is multi-executed while RotaryKnife_Instance1 is executing. In this case both instances have same setting of SlaveWaitPosition.

- When this FB is multi-executed while the SlaveAxis is under operation of *Busy*=TRUE, executing FB Instance; in this case, Instance1, keep operation.
- Multi-executed FB Instance; in this case, Instance2, stops error of *Error*=TRUE, therefore the executing condition is not met at multi-executed.
- The dotted lines of the *Scan* shown in the figure indicate multiple task periods.

⚠ Caution	
Do not execute another instance of this FB (change *Enable* from FALSE to TRUE) for same axis when plural instances of this FB are placed in a program and one of them is being executed. Velocity of SlaveAxis may largely increase and result in property damage.	⚠

- **While SlaveAxis is at SlaveWaitPosition, another Instance of this FB is executed.**



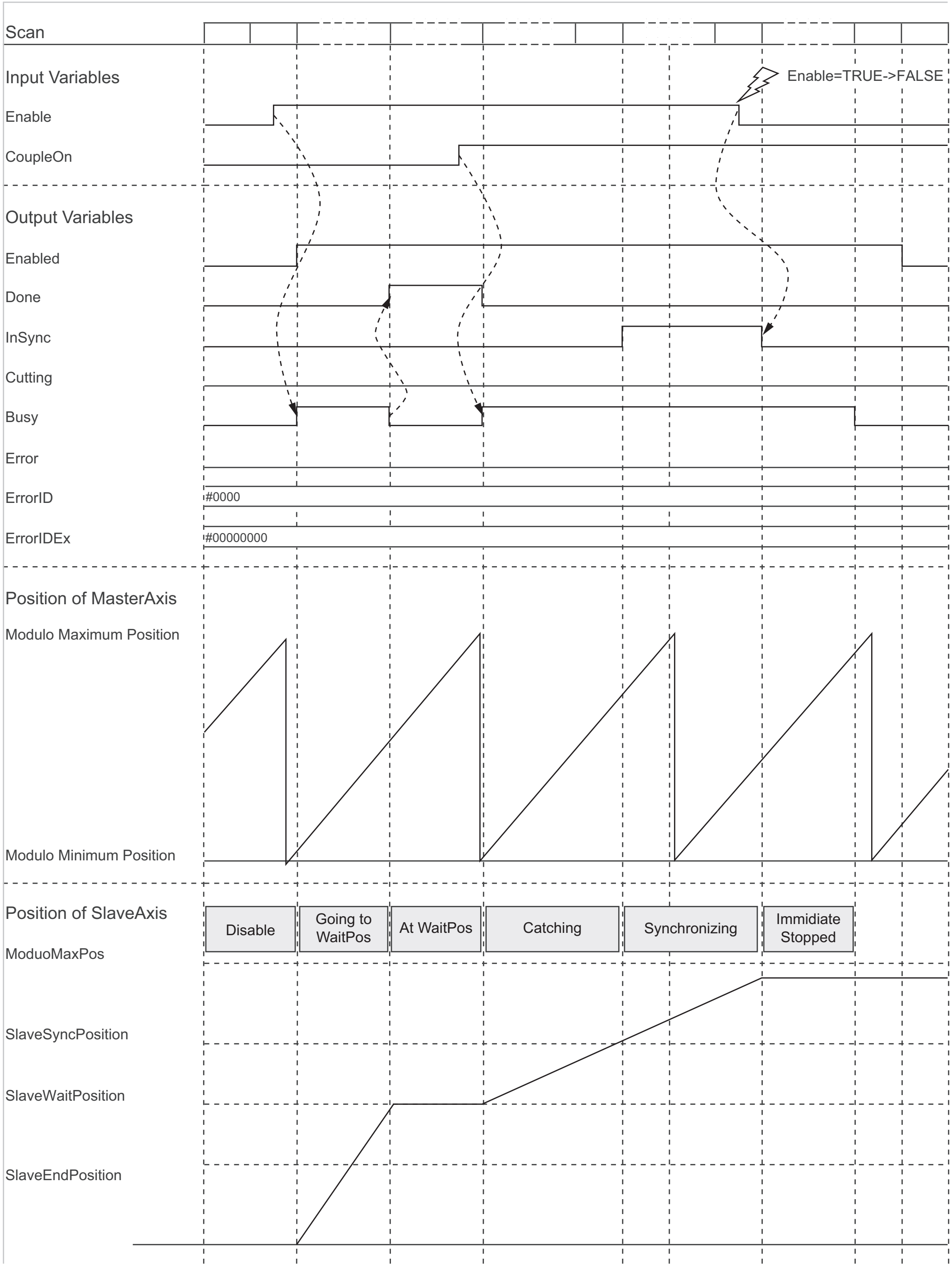
- Figure above shows the timing chart that RotaryKnife_Instance2 is multi-executed while RotaryKnife_Instance1 is executing. In this case both instances have same setting of SlaveWaitPosition.
- When this FB is multi-executed while the SlaveAxis is at SlaveWaitPosition of *Done* =TRUE, the executing FB Instance; in this case, Instance1, ends.
- Multi-executed FB Instance; in this case, Instance2, is executed.

- The dotted lines of the Scan shown in the figure indicate multiple scans/task periods.

⚠ Caution	
Do not execute another instance of this FB (change *Enable* from FALSE to TRUE) for same axis when plural instances of this FB are placed in a program and one of them is being executed. Velocity of SlaveAxis may largely increase and result in property damage.	⚠

● While SlaveAxis *Busy* =TRUE, *Enable* changes from TRUE to FALSE

When *Enable* changes to FALSE while *Busy* is TRUE, SlaveAxis stops immediately.

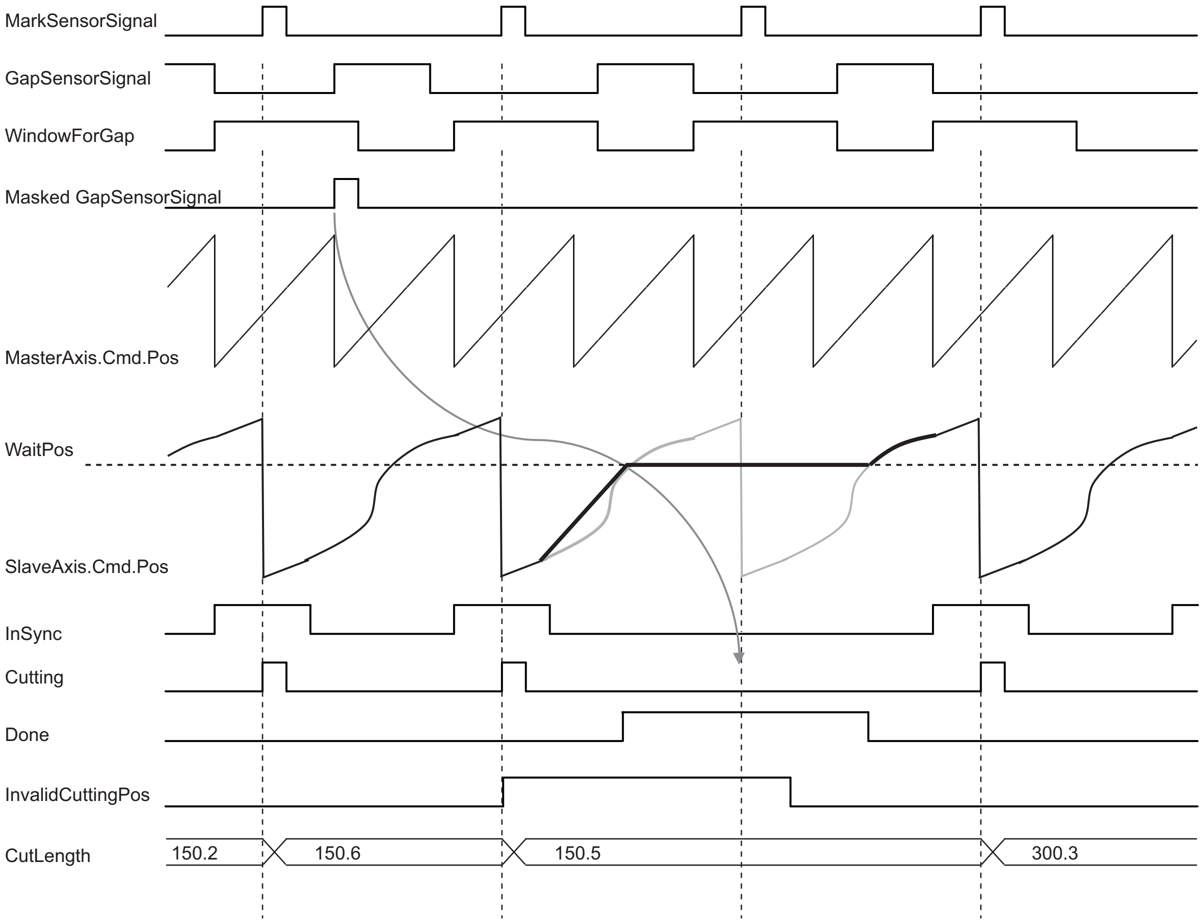


- When *Enable* changes from TRUE to FALSE while the SlaveAxis is under operation, the output variables excluding *Busy* and *Enabled*, change to FALSE or 0 (the same outputs as for non-execution state).
- The SlaveAxis stops immediately at the deceleration rate of 0.
- After stopping, *Busy* changes to FALSE.
- After one task period when *Busy* changes to FALSE, *Enabled* changes to FALSE.
- The dotted lines of the Scan shown in the figure indicate multiple task periods.

⚠ Caution

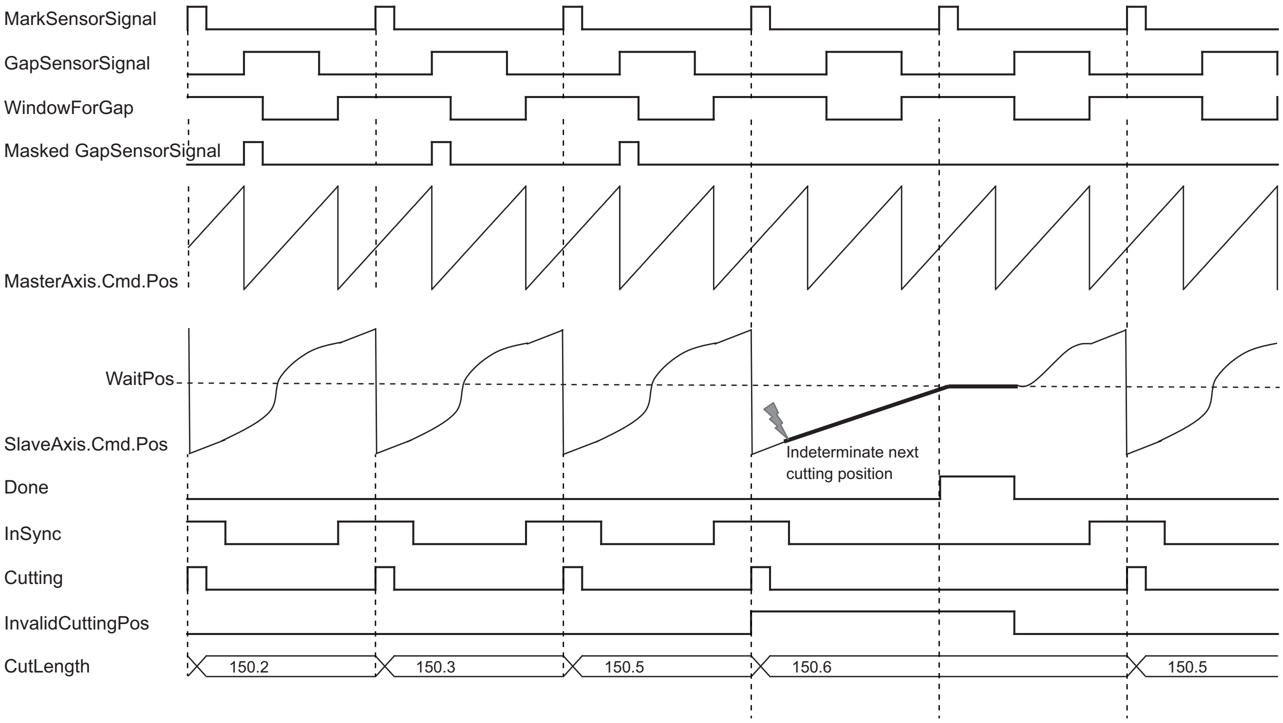
Make sure that the SlaveAxis stopped before *Enable* changes to FALSE. When *Enable* changes to FALSE while SlaveAxis is moving, it stops immediately.That may result in property damage.	⚠

● When the Skip behavior of NoGapNoSeal function is executed



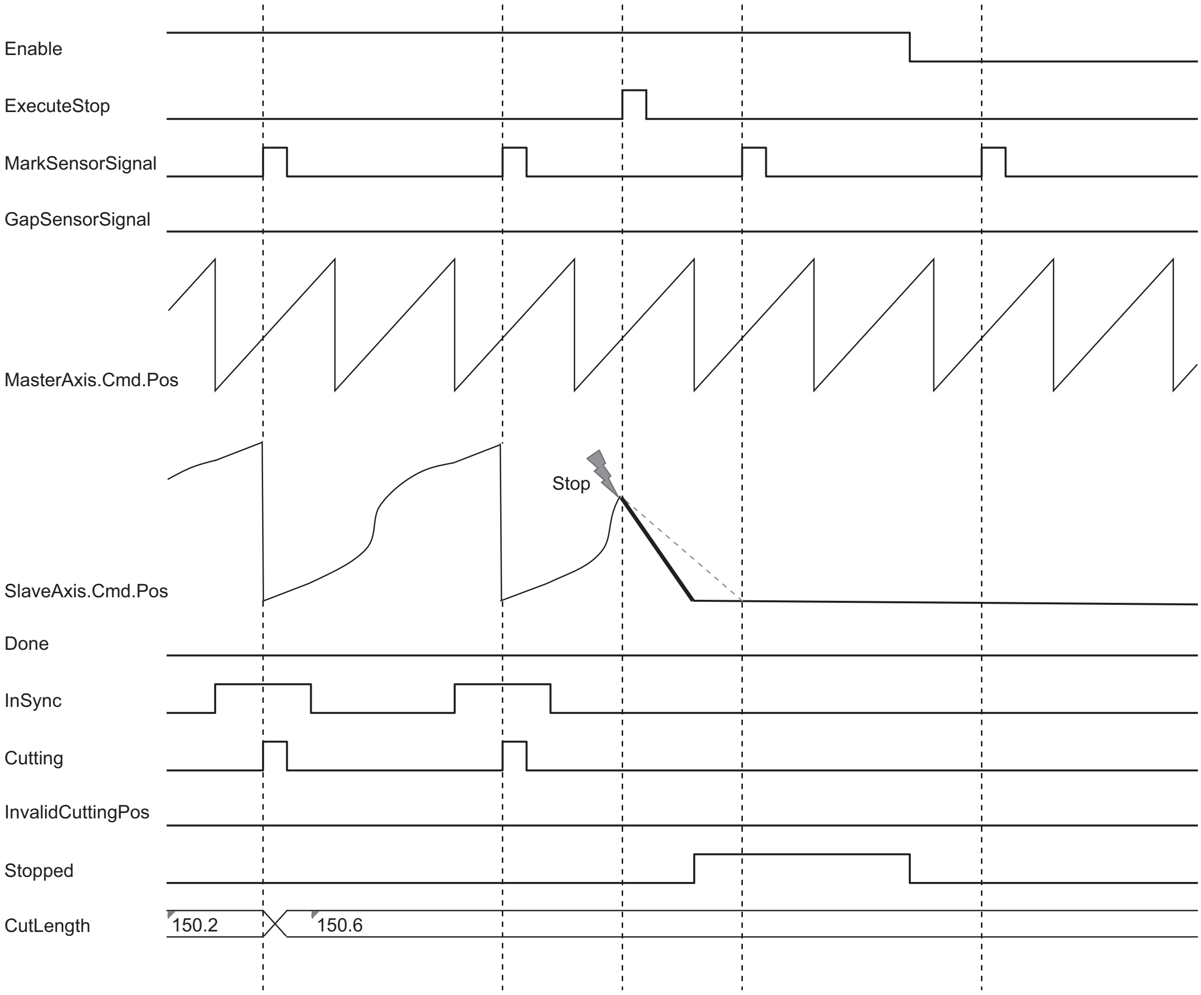
● When all the cutting position is judged that cutting is not possible

When NoGapNoSeal function is valid, if the all cutting positions are judged that cutting is not possible, move to the slave wait position (WaitPos).



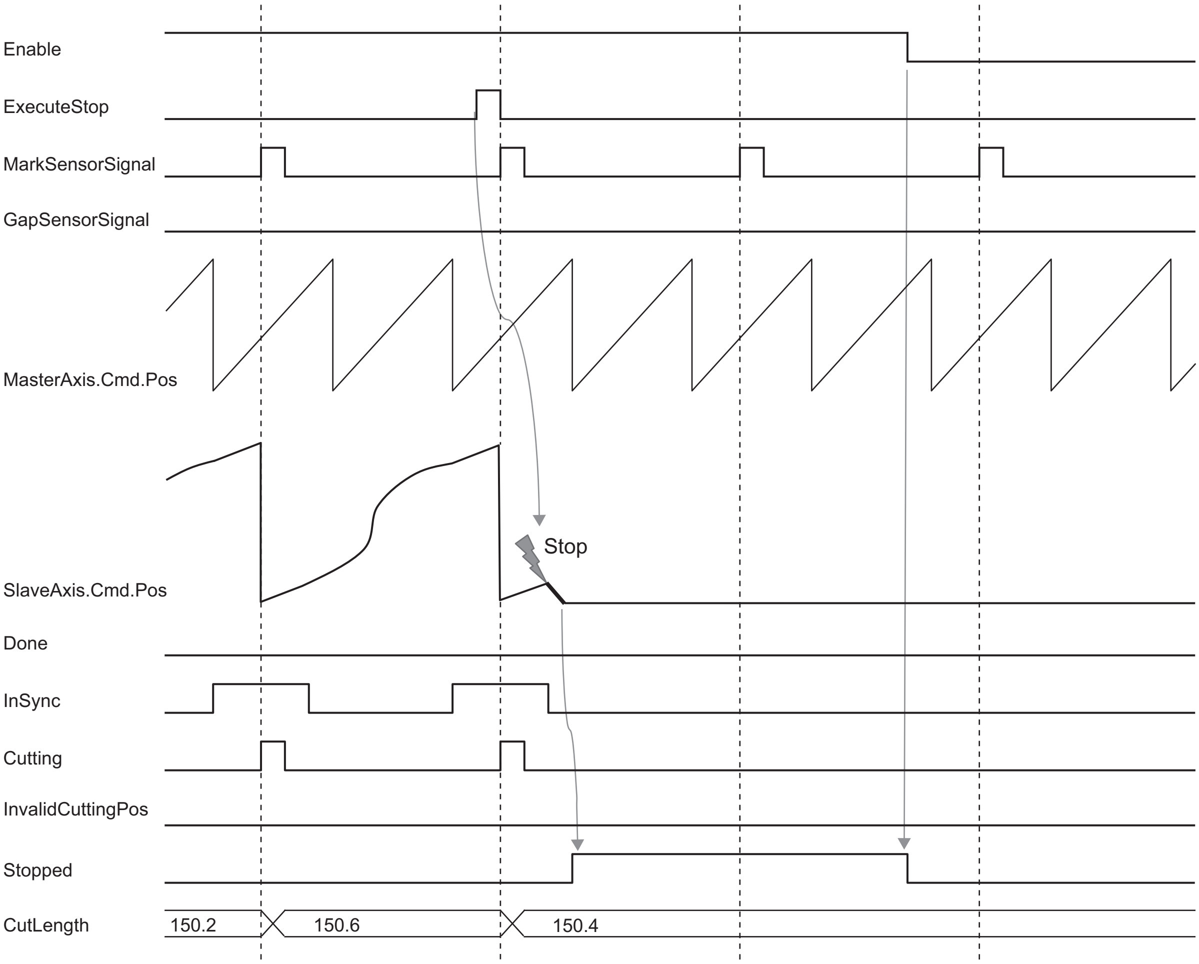
● When deceleration stop is instructed and it decelerates in a specified deceleration, it enters the InSyncPhase

If it enters the InSyncPhase when deceleration stop is executed in a specified deceleration, calculate the deceleration so that it does not enter the InSyncPhase.



● When deceleration stop is instructed during InSyncPhase

When deceleration stop is performed during InSyncPhase, after it passes through the InSyncPhase it starts deceleration stop.



Troubleshooting

ErrorID

The following table lists the error codes that are stored in *ErrorID*.

Error code	Status	Description	Correction
#0000	Normal end (Normal)	-	-
#3C42	Error end (Error)	An error occurred in the standard instruction that is used in the FB.	Check the last 4 digits of the error code that is stored in *ErrorIDEx* and follow the instructions in the manuals to take measures.[*1]
#3C43	Error end (Error)	An error occurred in the FB.	Check the error code stored in *ErrorIDEx*.

[*1] Refer to the motion control instructions reference manual and the instructions reference manual.

ErrorIDEx

The following table lists the error codes that are stored in *ErrorIDEX* when #3C43 is stored in the *ErrorID* output variable.

Expansion error code	Status	Description	Correction
16#00010001	Invalid cam profile type	The cam profile type is invalid.	Check the cam profile type.
16#00010002	Invalid allowable skip count	The value of the allowable skip count is invalid.	Check the value of the allowable skip count.
16#00010003	MasterAxis error	An axis minor fault occurred in the MasterAxis.	Clear the axis minor fault in the MasterAxis.
16#00010004	Skip count error	The skip count reached the *SkipCountLimit* of input variable.	Set the cut length to a greater value. Lower the synchronous velocity.
16#00010005	Axis setting error	The axis number is invalid.	Check the axis number.
16#00020001	Execution condition error	The execution conditions of this FB are not met.	Check the execution conditions described in *Execution Conditions of the FB* on page 44.
16#00020002	Invalid input value	The input value to this FB is outside of the specification range.	Check the valid range specified in *Input Variables* on page 48.
16#00020003	Conditions for continuous execution error	The conditions to continue the execution of this FB are not met.	The home position of SlaveAxis is not defined. Define the home position.
16#00020004	Invalid distance	The command value to the SlaveAxis was a negative value during an execution of this FB.	Check the input variable settings or axis configurations.
16#00020005	Invalid re-execution under stopping process	After *Enable* changes to FALSE and starts stopping process, again *Enable* changes TRUE.	Program should be considered as that *Enable* changes from FALSE to TRUE again, on the condition that all BOOL output variables are FALSE after two scans/task periods.
16#00030001	Invalid input variable	The input variable is invalid. • *MasterAxisNo* is out of range.	Correct the input variable.

Expansion error code	Status	Description	Correction
16#00030002	Invalid MasterAxis configuration	The MasterAxis configuration is invalid. • Modulo Maximum Position Setting Value is 0 or less • Modulo Minimum Position Setting Value is other than 0.	Check the MasterAxis configuration.
16#00030003	Invalid SlaveAxis configuration	The SlaveAxis configuration is invalid. • SyncPos is out of range.*1 • VelFactor is out of range.*1	Check the SlaveAxis configuration.
16#00030004	Invalid cutting operation setting	The cutting operation setting is invalid. • Mode is out of range. • Cut length is out of range.*1 • *MasterSyncPos* is less than *MasterAxis.Act.Pos*.*1 • *MasterSyncPos* is out of range.*1 • DistanceToMarkSensor is out of range.*2 • The set value of *CuttingCfg.MaskLength* is out of range.*2 • The set value of *CuttingCfg.MaskLength* is larger than *CuttingCfg.CutLength*.*2	Check the cutting operation setting.
16#00030005	Buffer full error	64 marks were detected between the rotary knife and the mark sensor.	Correct the position of the mark sensor.
16#00030006	Invalid SlaveAxis count	• *MasterAxis.Act.Pos* and latched position are outside the range of the MasterAxis' modulo configuration. • The MasterAxis performs forward operation exceeding the next cut length.	Correct the MasterAxis configuration.
16#00030007	Latch instruction aborted	The MC_TouchProbe instruction has output *CommandAborted*.	Check if there is any logic that externally stops the latch instruction.
16#00030008	Invalid offset	After offset compensation, a negative value is used as the cut length.	Check the offset value.
16#00030009	MasterAxis reverse operation error	The MasterAxis performs reverse operation exceeding the previous cut length.	Check the operation of the MasterAxis.
16#0003000A	MasterAxis error detected	An error is detected in the MasterAxis during operation.	Check the operation of the MasterAxis.

Expansion error code	Status	Description	Correction
16#0003000B	NoGapNoSeal setting invalid	When *EnableNGNS* is TRUE The value of • *DistanceToGapSensor* is zero or negative value • The value of *WindowForGap* is a negative value • *DistanceToGapSensor* ≥ *DistanceToMarkSensor*	Check the cutting operation setting.
16#0003000C	Continuous operation data invalid	Out of range *ContinuousOpe.Data* of specification	Correct the Continuous operation data.
16#00800001 16#00850001[*3]	Set value error	The set value is out of range. The correlation is invalid.	An error occurs if the following correlation is established. • EndPos > SyncPos • EndPos < ModuloMaxPos • SyncPos < ModuloMaxPos
16#00800002 16#00850002[*3]	Synchronous velocity error	The velocity of the MasterAxis is 0 or less.	Check the velocity of the MasterAxis.
16#00800003 16#00850003[*3]	Insufficient travel distance at first execution	The travel distance is insufficient at the first catching operation.	When moving at the acceleration of the SlaveAxis, the *MasterSyncPos* is passed.
16#00800004 16#00850004[*3]	Too long travel distance at first execution	The travel distance is exceeded at the first catching operation.	It is not possible to catch up at the *MasterSyncPos*.
16#00800005 16#00850005[*3]	Maximum velocity exceeded	The catching velocity is over the maximum velocity.	Reduce the velocity of the SlaveAxis or, increase the MaxVelocity.
16#00800006	Acceleration/deceleration setting error	When *CuttingCfg.ProfileType* is set to 0 (Trapezoid), the velocity during the CatchingPhase was a negative value.	Increase the acceleration and deceleration value.
16#00800007 16#00850007[*3]	Too fast MasterAxis velocity	With the velocity of the MasterAxis, the MasterAxis passes the InSyncPhase: synchronizing section in one period.	Slower the velocity of the MasterAxis.

[*1] The check is executed at *Continuous* Mode.

[*2] The check is executed at *MarkToMark* Mode and *ExternalMarkToMark* Mode.

[*3] The Expansion Error Codes vary depending on the set value in *SlaveCfg.ProfileType* as follows.
SlaveCfg.ProfileType =0 (Trapezoidal): 16#0080XXXX
SlaveCfg.ProfileType =5 (5-order polynomial): 16#0085XXXX

Additional Information

Multiple exit codes listed above may occur at the same time. In this case, the youngest exit code will be output. Therefore, deal with the output exit error codes in order.

Countermeasure for BufferEmpty

The causes and their solutions are described to troubleshoot occurrence of “BufferEmpty”.

The BufferEmpty occurs when the value of *CuttingCfg.Mode* is 1 (Mark To Mark Mode) or 2 (External Mark To Mark Mode). The causes and their solutions are as follows.

Cause	Solution/Correction
The distance between marks is larger than the input variable *CuttingCfg.DistanceToMarkSensor*.	Adjust the position of the mark sensor so that the distance between marks is smaller than *CuttingCfg.DistanceToMarkSensor*. There should be one or more marks between mark sensor and blade.
The actual distance between marks differs from the value of the input variable *CuttingCfg.CutLength*.	Set the value of *CuttingCfg.CutLength* to the actual distance between marks.
The value of the input variable *CuttingCfg.Tolerance* is too small.	Adjust the value of *CuttingCfg.Tolerance* according to the product to cut.
There is malfunction on the mark sensor.	Replace or clean the mark sensor. Or, check whether the mark sensor is normal or not.

Additional Information

- For BufferEmpty, refer to *Buffer Function* on page 73.
- For DistanceToMarkSensor, refer to *sCUTTING_CFG* on page 52.
- For *CuttingCfg.Cutlength*, refer to *CutLength Function* on page 75
- For *CuttingCfg.Tolerance*, refer to *Tolerance Function* on page 76.

Countermeasures for Skip

This section describes the causes and countermeasures for Skip. The contents differ depending on the values of *CuttingCfg.Mode*. The following table shows the causes and measures.

● *CuttingCfg.Mode* is 0 (the Continuous mode):

Cause	Solution/Correction
The value of the input variable *CuttingCfg.Cutlength* is smaller than the range of InSyncPhase.	Set the value of *CuttingCfg.Cutlength* larger than the range of InSyncPhase
The value of the input variable *CuttingCfg.Cutlength* is too small and the velocity of the product is too fast.	Adjust the set value of SlaveSyncPos and SlaveEndPos to set the range of InSyncPhase wider. Slow down the velocity of the product.
InSync area is too wide.	Set the InSync area narrower. Adjust the set value of SlaveSyncPos and SlaveEndPos.
Slave velocity reaches at its limitation.	Check the set value of *SlaveCfg.MaxVelocity*, *SlaveCfg.Acceleration* and *SlaveCfg.Deceleration*.

● *CuttingCfg.Mode* is 1 (the Mark To Mark mode) or 2 (the External Mark To Mark mode):

Cause	Solution/Correction
The value of the actual cut length (distance between marks) is smaller than the range of InSyncPhase.	Set the range of InSyncPhase smaller than actual cut length (DistanceToMarkSensor).
The value of the input variable *CuttingCfg.Cutlength* is too small and the velocity of product is so fast.	Adjust the set value of SlaveSyncPos and SlaveEndPos to set the range of InSyncPhase wider. Slow down the velocity of the product.
InSync area is too wide.	Set the InSync area narrower. Adjust the set value of SlaveSyncPos and SlaveEndPos.
Slave velocity reaches at its limitation.	Check the set value of *SlaveCfg.MaxVelocity*, *SlaveCfg.Acceleration* and *SlaveCfg.Deceleration*.

Additional Information

For Skip, refer to *Skip Function* on page 67.

Sample Programming1

Using Function Block RotaryKnife in a sample program.

This chapter describes the system that consists of one MasterAxis and one SlaveAxis and they are both servo motors.

When using this FB in the system similar to this one, select Mark To Mark Mode for the operation mode.

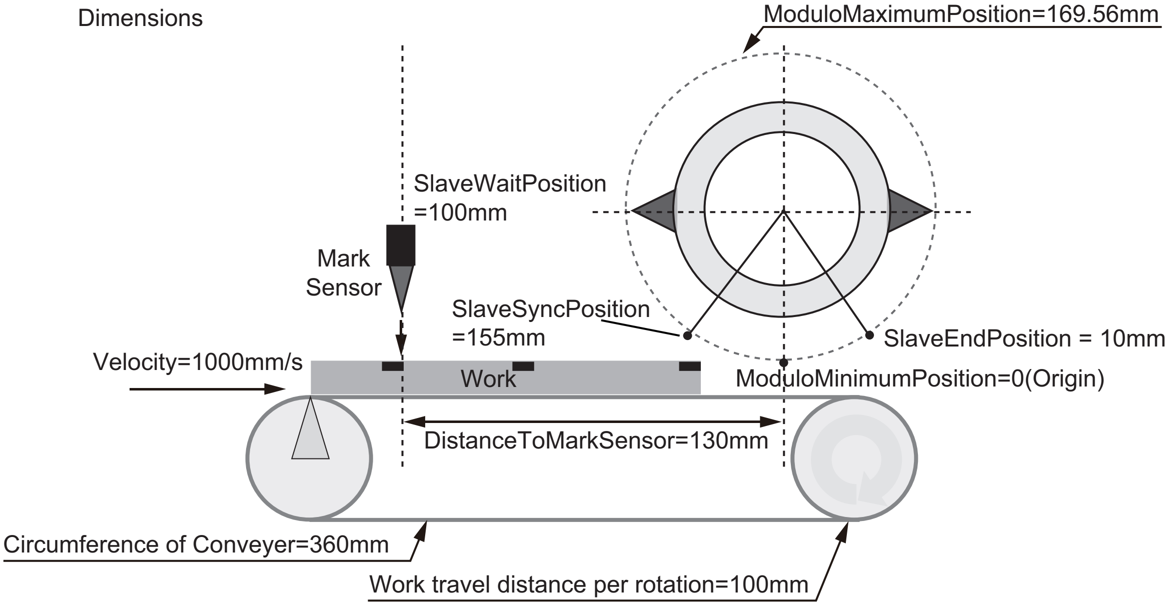
Machine Sample

This section describes machine sample when using FB, overview and dimensions, and hardware configuration.

● Overview and Dimensions

This sample system cuts the product at mark on product detected by mark sensor (this is Mark-ToMark mode).

Each dimension is shown in the following figure.


Dimensions
ModuloMaximumPosition=169.56mm
SlaveWaitPosition
=100mm
Mark
Sensor
SlaveSyncPosition
=155mm
SlaveEndPosition = 10mm
Velocity=1000mm/s
Work
ModuloMinimumPosition=0(Origin)
DistanceToMarkSensor=130mm
Circumference of Conveyer=360mm
Work travel distance per rotation=100mm

● Hardware Configuration

The hardware configuration of above sample system is as follows.

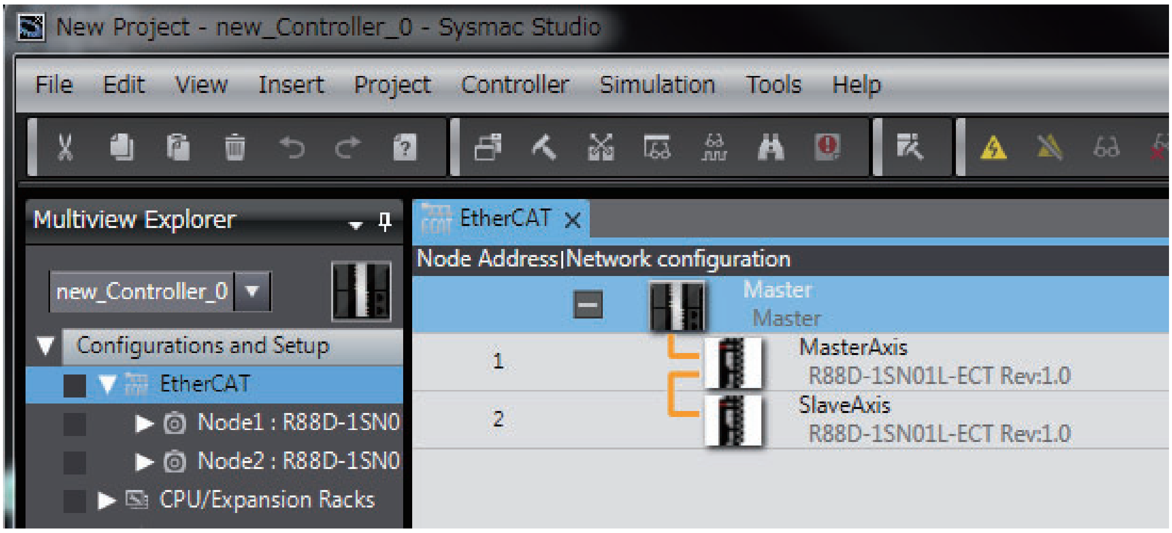
Name	Type	Version	Number
MachineAutomationController	NJ501-1500	Ver. 1.10	1
AC Servo Drive(1S Series)	R88D-1SN01L-ECT	Rev.1.0	2
AC Servo Motor(1S Series)	R88M-1□	-	2
Mark Sensor	- not specified	-	1

Project Sample

This section describes project sample using FB, EherCAT configuration, task settings, motion control setup, variables, and program sample.

● EtherCAT Configuration

The EtherCAT network configuration is as follows.



new_Controller_0

- Configurations and Setup
  - EtherCAT
    - Node1 : R88D-1SN0
    - Node2 : R88D-1SN0
  - CPU/Expansion Racks

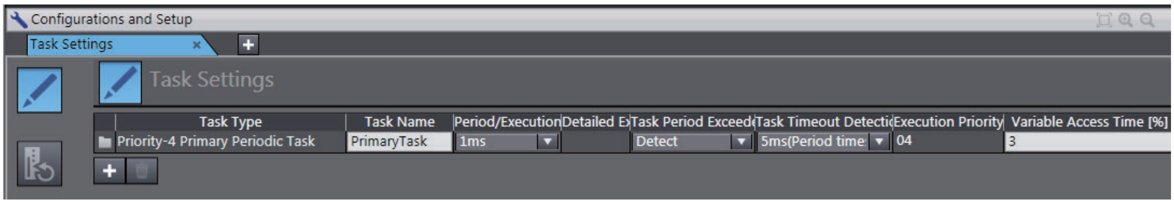
Node Address	Network configuration
	Master Master
1	MasterAxis R88D-1SN01L-ECT Rev:1.0
2	SlaveAxis R88D-1SN01L-ECT Rev:1.0

● Task Settings

The task settings are as follows.

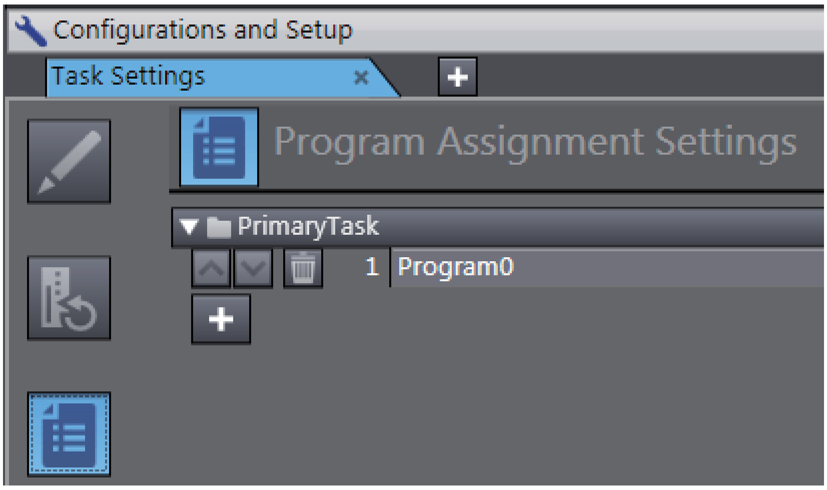
Item	Settings
Task Type	Primary Periodic Task
Task Period	1ms
Assigned Program to the Task	Program0

Task Settings



Task Type	Task Name	Period/Execution	Detailed E:	Task Period Exceed	Task Timeout Detecti	Execution Priority	Variable Access Time [%]
Priority-4 Primary Periodic Task	PrimaryTask	1ms		Detect	5ms(Period time	04	3

Program assignment Settings



Configurations and Setup

Task Settings

Program Assignment Settings

PrimaryTask

1 Program0

● Motion Control Setup

The followings are a sample of Motion control setup.

Settings below are changed and others which are not shown keep initial values.

Axis Basic Settings

Axis Name	Axis No	Axis Type	Output device
MasterAxis	0	ServoAxis	Node:1 Device: R88D-1SN01L-ECT
SlaveAxis	1	ServoAxis	Node:2 Device: R88D-1SN01L-ECT

Unit Conversion Settings

Axis Name	Unit of Display	Command pulse count per motor rotation	Work travel distance per motor rotation
MasterAxis	mm	131072	100
SlaveAxis	mm	131072	169.56

Position Count Settings

Axis Name	Count Mode	Modulo maximum position setting value	Modulo minimum position setting value
MasterAxis	Rotary mode	360	0
SlaveAxis	Rotary mode	169.56	0

Variables

- **Internals**

Name	Data type	Initial Value	Comment
PowerOn	BOOL		
Power_MasterAxis	MC_Power		
Power_SlaveAxis	MC_Power		
Homing	BOOL		
Home_MasterAxis	MC_Home		
Home_SlaveAxis	MC_Home		
RunCondition_MasterAxis	BOOL		
MoveVel_MasterAxis	MC_MoveVelocity		
Master_Velocity	LREAL	1000	
Master_Acceleration	LREAL	0	
Master_Deceleration	LREAL	0	
Stop_MasterAxis	MC_Stop		
Stop	BOOL		
Run	BOOL		
Pwr1_Status	BOOL		
Pwr2_Status	BOOL		
RotaryKnife_Instance	OmronLib\PKG_RK\RotaryKnife		
ContinuousOpe	OmronLib\PKG_RK\sCONTINUOUS_OPE		
Enable	BOOL		
CoupleOn	BOOL		
ExecuteStop	BOOL		
GapSensorSignal	BOOL		
TriggerVariable	BOOL		
RecordedPosition	LREAL		
MasterCfg	OmronLib\PKG_RK\sMASTER_CFG		
SlaveCfg	OmronLib\PKG_RK\sSLAVE_CFG		
CuttingCfg	OmronLib\PKG_RK\sCUTTING_CFG		
Enabled	BOOL		
Done	BOOL		
InSync	BOOL		
Cutting	BOOL		
CutLengthOut	LREAL		
Skip	BOOL		
BufferEmpty	BOOL		
MarkNotAccepted	BOOL		
Stopped	BOOL		
InvalidCuttingPos	BOOL		
Busy	BOOL		
CommandAborted	BOOL		
Error	BOOL		
ErrorID	WORD		
ErrorIDEx	DWORD		

- **Externals**

Name	Data type	Constant	Comment
MasterAxis	_sAXIS_REF	TRUE	
SlaveAxis	_sAXIS_REF	TRUE	
_EC_PDSlavTbl	ARRAY[1..192] OF BOOL	TRUE	
_EC_CommErrTbl	ARRAY[1..192] OF BOOL	TRUE	

- **Global Variables**

Name	Data type	Initial Value	AT	Constant	Comment
MasterAxis	_sAXIS_REF		MC://_MC_AX[0]	TRUE	
SlaveAxis	_sAXIS_REF		MC://_MC_AX[1]	TRUE	

Set Value

- **Setting *MasterCfg* (Data Type: sMASTER_CFG)**

Name	Data type	Setting Value	Comment
TriggerInput	_sTRIGGER_REF	Refer to *Setting TriggerInput* on page 90.	
CmdPosMode	BOOL	FALSE	The type, which is referred to as the master axis, is specified as the feedback value.
EnableAdvanceAngleCorr	BOOL	FALSE	The Advance Angle Compensation function is disabled.
AdvanceAngleCorrValue	LREAL	0.0	-

- **Setting *MasterCfg.TriggerInput* (Data Type:_sTRIGGER_REF)**

Name	Data type	Setting Value	Comment
Mode	ENUM	_mcDrive	Trigger Mode
LatchID	ENUM	_mcLatch1	Latch ID
InputDrive	ENUM	_mcEXT	External Input

- **Setting *SlaveCfg* (Data Type: sSLAVE_CFG)**

Name	Data type	Setting Value	Comment
SyncPos	LREAL	155.0	SlaveSyncPos
EndPos	LREAL	10.0	SlaveEndPos
WaitPos	LREAL	100.0	SlaveWaitPos
MaxVelocity	LREAL	100000.0	MaxVelocity
Acceleration	LREAL	100000.0	SlaveAcceleration
Deceleration	LREAL	100000.0	SlaveDeceleration
BladeNum	UDINT	2	Number of Blade
BladeMountErrorComp	ARRAY[1..4] OF LREAL	4(0.0)	-
ProfileType	UDINT	0	Cam Profile Type
VelocityToWaitPos1	LREAL	360.0	Velocity to SlaveWaitPos
VelocityToWaitPos2	LREAL	360.0	
VelFactor	UDINT	100	Override Velocity Factor
StopMode	_eMC_STOP_MODE	_emc_STOP _MODE# _mcDeccelerationStop	

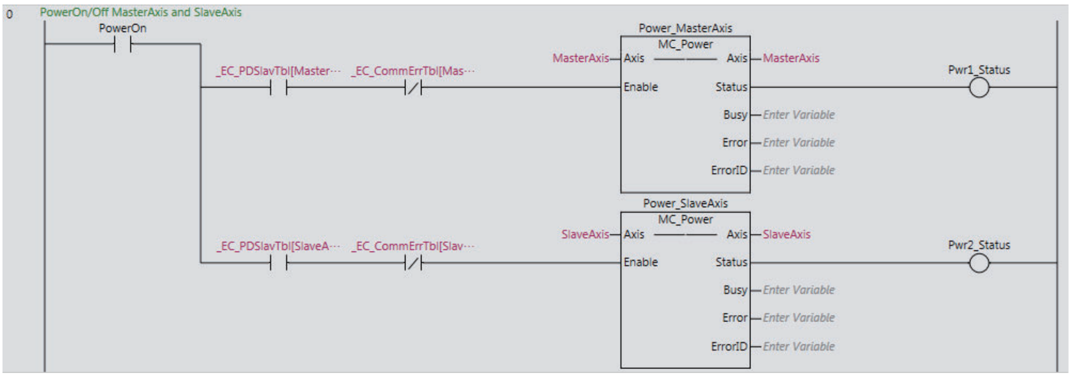
- **Setting *CuttingCfg* (Data Type: sCUTTING_CFG)**

Name	Data type	Setting Value	Comment
Mode	UDINT	1	MarkToMark
CutLength	LREAL	50.0	Ideal mark pitch
ToleranceForMark	LREAL	0.0	
WindowForGap	LREAL	0.0	
MasterSyncPos	LREAL	0.0	Use only continuous mode
DistanceToMarkSensor	LREAL	130.0	Distance between the mark sensor and the blade.
EnableNGNS	BOOL	FALSE	
DistanceToGapSensor	LREAL	0.0	
OffsetFromMark	LREAL	0.0	
SkipCountLimit	UDINT	0	

Ladder Diagram

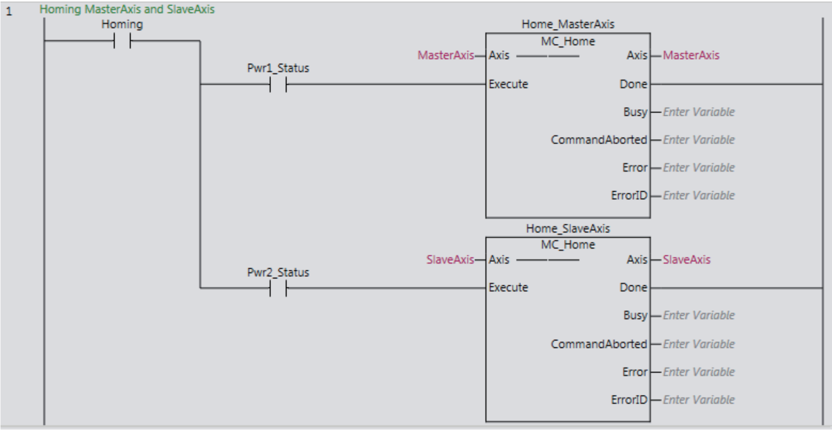
A sample program (Program0) with RotaryKnife.

1



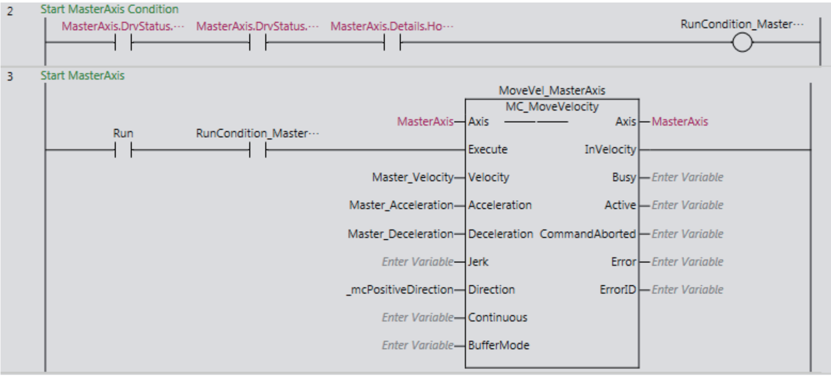
Power on the master axis and slave axis.

2



Execute homing to origin position.

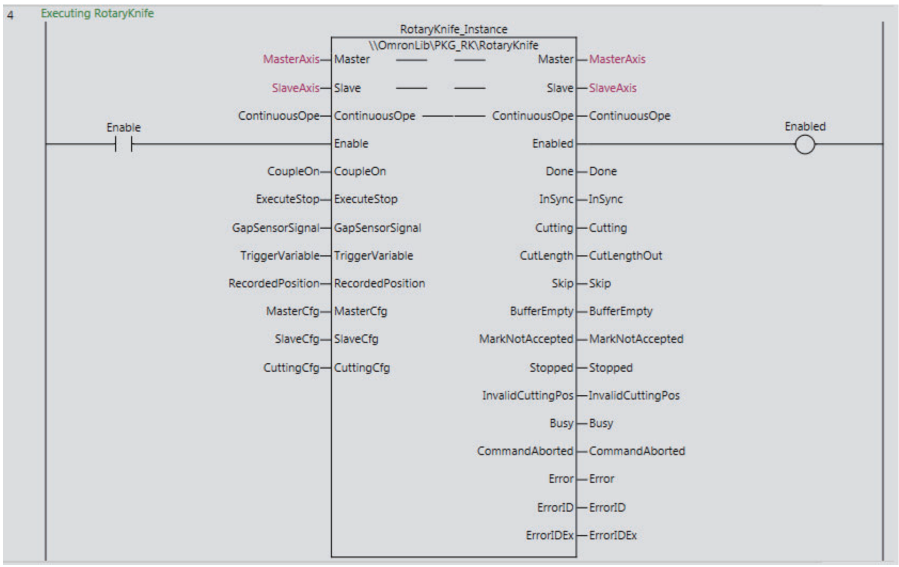
3



Line2
Confirm the condition for starting master axis.

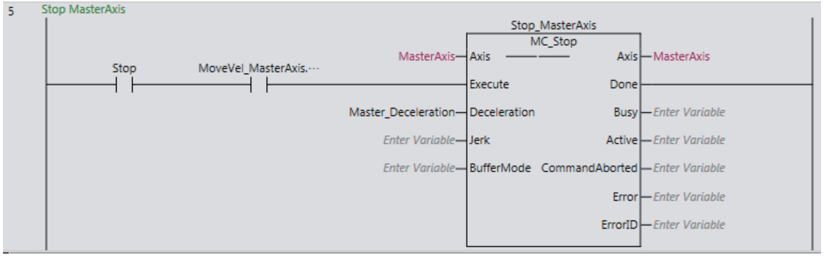
Line3
After then, start Master Axis.
Run is manual bit to execute MC_MoveVelocity.

4



Allocate RotaryKnife instance and its input/outputs.

5



The value of the [Stop] to TRUE, to stop the main axis.

⚠ WARNING

Using this FB in a device, confirm that the program and FB operates properly. Design a program so that safety measures such as fail-safe circuits are implemented outside of the FB.

Operation procedure

The operation procedure to execute RotaryKnife.

● From Power off To Running (Synchronizing)

1	Power on the system	Turn the system main power on.
	▼	
2	Servo on Master Axis and Slave Axis	After starting up and confirming the safety, servo on Master and Slave Axis.
	▼	
3	Homing Master Axis and Slave Axis	Confirm the safety and start homing Master and SlaveAxis.
	▼	
4	Setting product to be cut.	Locate the product to be cut in the machine.
	▼	
5	Start moving to SlaveWaitPosition	Set *Enable* to TRUE, then SlaveAxis goes to SlaveWaitPosition.
	▼	
6	Arriving at SlaveWaitPos	After SlaveAxis moves to SlaveWaitPos, *Done* is set TRUE.
	▼	
7	To Start MasterAxis, set *RUN* TRUE	*RUN* is in the 3rd line of *Ladder Diagram* on page 121. Then, set *CoupleOn* to TRUE.
	▼	
8	Synchronizing	Master and SlaveAxis start Synchronizing. Then products are cut continuously.

From Running (Synchronizing) to CycleStop

1	Synchronizing stop	When you want to stop synchronizing, set *CoupleOn* to FALSE. CycleStop will start.
	▼	
2	CycleStop	SlaveAxis terminates synchronizing. Then, SlaveAxis goes to SlaveWaitPosition.
	▼	
3	CycleStop completes	After stopping at SlaveWaitPosition, *Done* is set TRUE. SlaveAxis waits for *CoupleOn* being set TRUE again.

From CycleStop to Re-Synchronizing

1	Synchronizing start	When *CoupleOn* value is set to TRUE, synchronizing restarts.
	▼	
2	Cutting operation will be restarted.	Synchronizing and Cutting operation will be restarted.

From Running (Synchronizing) to Stop RotaryKnife

1	Syncronizing stops	When you do not want to continue cutting any more, set *CoupleOn* to FALSE after CycleStop.
	▼	
2	Cyclestop	After CycleStop, set *Enable* to FALSE.
	▼	
3	Rotary Knife stops	Cutting Operation is finished.

From Error during Synchronizing to RotaryKnife Restart

1	An Error occurs	When an error(s) occurs in RotaryKnife(FB), the RotaryKnife stops (Error Stop). SlaveAxis will decelerate to stop. (Error End).
	▼	
2	Reset the error	Check cause of the error(s). After checking, confirm the safety and reset the error. For troubleshooting, refer to the user's manual.
	▼	
3	Procedures after the error is reset.	Refer to *From Emergency/Error stop to RotaryKnife Restart* on page 125.

From Emergency Stop during Synchronizing to RotaryKnife Restart

1	Emergency stop occurs	When an emergency(s) stop or a fatal error(s) occurs in the machine, the cutting operation stops immediately (Emergency Stop).
	▼	
2	Reset the error	Check cause of the error. After checking, confirm the safety and reset the error. For troubleshooting, refer to the user's manual.
	▼	
3	The procedure after the error is reset.	Refer to *From Emergency/Error stop to RotaryKnife Restart* on page 125.

● From Emergency/Error stop to RotaryKnife Restart

1	Reset the error	Reset the error.
	▼	
2	Servo on Master Axis and Slave Axis	After resetting the error, confirm the safety and servo on Master and Slave Axis.
	▼	
3	Homing Master Axis and Slave Axis	Confirm the safety and start homing Master and SlaveAxis.
	▼	
4	Set the product to be cut.	Locate the product to be cut in the machine.
	▼	
5	Start moving to the wait position	Set *Enable* to TRUE, then SlaveAxis moves to SlaveWaitPosition.
	▼	
6	Reaches wait position	After SlaveAxis moves to SlaveWaitPos, *Done* is set TRUE.
	▼	
7	Start MasterAxis	*RUN* is in the 3rd line of *Ladder Diagram* on page 121. Then, set *CoupleOn* to TRUE.
	▼	
8	Synchronizing	MasterAxis and SlaveAxis start synchronization and cut products continuously.

Sample Programming 2

Using Function Block RotaryKnife in a sample program.

This chapter describes the system consists of one master axis and two slave axes.

To use the FB in such a system, user should select the “ExternalMarkToMark” mode of the FB.

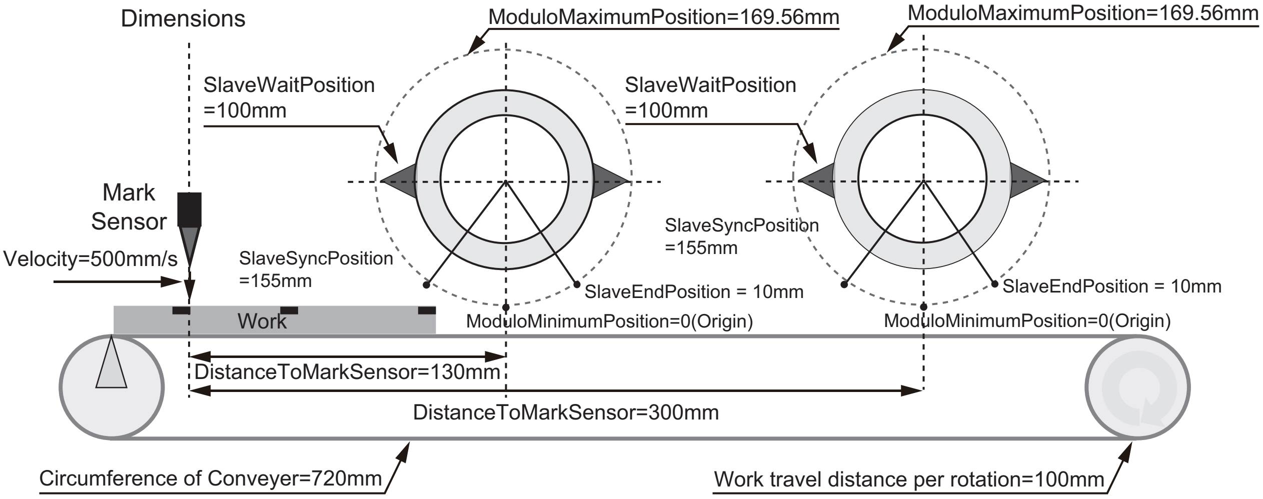
Machine sample

This section describes machine sample when using FB, overview and dimensions, and hardware configuration of the machine.

● Overview and Dimensions

This sample system cuts the product at mark on product detected by mark sensor.

There are two rotary knives to cut the product.


Dimensions
ModuloMaximumPosition=169.56mm
ModuloMaximumPosition=169.56mm
SlaveWaitPosition =100mm
SlaveWaitPosition =100mm
Mark Sensor
Velocity=500mm/s
SlaveSyncPosition =155mm
SlaveSyncPosition =155mm
SlaveEndPosition = 10mm
SlaveEndPosition = 10mm
Work
ModuloMinimumPosition=0(Origin)
ModuloMinimumPosition=0(Origin)
DistanceToMarkSensor=130mm
DistanceToMarkSensor=300mm
Circumference of Conveyer=720mm
Work travel distance per rotation=100mm

● Hardware Configuration

The hardware configuration of above sample system is as follows.

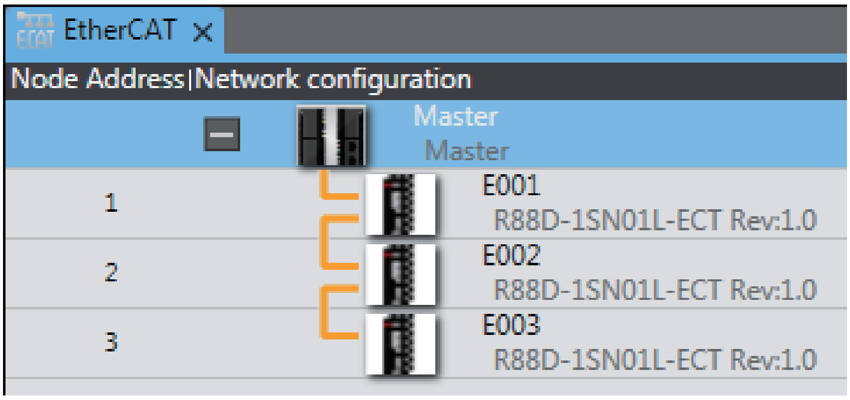
Name	Type	Version	Number
MachineAutomationController	NJ501-1500	Ver.1.10	1
AC Servo Drive(1S Series)	R88D-1SN01L-ECT	Rev.1.0	3
AC Servo Motor(1S Series)	R88M-1□	-	3
Mark Sensor	- not specified	-	1

Project Sample

This section describes project sample using FB, EherCAT configuration, task settings, motion control setup, variables, and program sample.

● EtherCAT Network Configuration

The EtherCAT network configuration is as follows.



EtherCAT

Node Address	Network configuration
	Master Master
1	E001 R88D-1SN01L-ECT Rev:1.0
2	E002 R88D-1SN01L-ECT Rev:1.0
3	E003 R88D-1SN01L-ECT Rev:1.0

● Task Setting

Refer to *Task Settings* on page 117.

● Motion Control Setting

The followings are samples of Motion control setting.

Settings below are changed and others which are not shown keep initial values.

Axis Basic Settings

Axis Name	Axis No	Axis Type	Output device
MasterAxis	0	ServoAxis	Node:1 Device: R88D-1SN01L-ECT
SlaveAxis	1	ServoAxis	Node:2 Device: R88D-1SN01L-ECT
SlaveAxis2	2	ServoAxis	Node:3 Device: R88D-1SN01L-ECT

Unit Conversion Settings

Axis Name	Unit of Display	Command pulse count per motor rotation	Work travel distance per motor rotation
MasterAxis	mm	131072	100
SlaveAxis	mm	131072	169.56
SlaveAxis2	mm	131072	169.56

Position Count Settings

Axis Name	Count Mode	Modulo maximum position setting value	Modulo minimum position setting value
MasterAxis	Rotary mode	720	0
SlaveAxis	Rotary mode	169.56	0
SlaveAxis2	Rotary mode	169.56	0

Variables

● Internals

Name	Data type	Initial Value	Comment
PowerOn	BOOL		
Power_MasterAxis	MC_Power		
Power_SlaveAxis	MC_Power		
Power_SlaveAxis2	MC_Power		
Pwr1_Status	BOOL		
Pwr2_Status	BOOL		
Pwr3_Status	BOOL		
Homing	BOOL		
Home_MasterAxis	MC_Home		
Home_SlaveAxis	MC_Home		
Home_SlaveAxis2	MC_Home		
RunCondition_MasterAxis	BOOL		
Run	BOOL		
MoveVel_MasterAxis	MC_MoveVelocity		
Master_Velocity	LREAL	500	
Master_Acceleration	LREAL	500	
Master_Deceleration	LREAL	500	
MC_TouchProbe_Instance	MC_TouchProbe		
TriggerVariable	BOOL		
TouchProbeDone	BOOL		
RecordedPosition	LREAL		
RotaryKnife_Instance	OmronLib\PKG_RK\RotaryKnife		
ContinuousOpe	OmronLib\PKG_RK\sCONTINUOUS_OPE		
Enable	BOOL		
CoupleOn	BOOL		
ExecuteStop	BOOL		
GapSensorSignal	BOOL		
MasterCfg	OmronLib\PKG_RK\sMASTER_CFG		
SlaveCfg	OmronLib\PKG_RK\sSLAVE_CFG		
CuttingCfg	OmronLib\PKG_RK\sCUTTING_CFG		
Enabled	BOOL		
Done	BOOL		
InSync	BOOL		
Cutting	BOOL		
CutLengthOut	LREAL		
Skip	BOOL		
BufferEmpty	BOOL		
MarkNotAccepted	BOOL		
Stopped	BOOL		
InvalidCuttingPos	BOOL		
Busy	BOOL		
CommandAborted	BOOL		
Error	BOOL		
ErrorID	WORD		
ErrorIDEx	DWORD		
RotaryKnife_Instance2	OmronLib\PKG_RK\RotaryKnife		
ContinuousOpe2	OmronLib\PKG_RK\sCONTINUOUS_OPE		
CuttingCfg2	OmronLib\PKG_RK\sCUTTING_CFG		

Name	Data type	Initial Value	Comment
Enabled2	BOOL		
Done2	BOOL		
InSync2	BOOL		
Cutting2	BOOL		
CutLength2	LREAL		
Skip2	BOOL		
BufferEmpty2	BOOL		
MarkNotAccepted2	BOOL		
Stopped2	BOOL		
InvalidCuttingPos2	BOOL		
Busy2	BOOL		
CommandAborted2	BOOL		
Error2	BOOL		
ErrorID2	WORD		
ErrorIDEx2	DWORD		
Stop	BOOL		
Stop_MasterAxis	MC_Stop		

- **Externals**

Name	Data type	Constant	Comment
MasterAxis	_sAXIS_REF	TRUE	
SlaveAxis	_sAXIS_REF	TRUE	
_EC_PDSlavTbl	ARRAY[1.192] OF BOOL	TRUE	
_EC_CommErrTbl	ARRAY[1.192] OF BOOL	TRUE	
SlaveAxis2	_sAXIS_REF	TRUE	

- **Global Variables**

Name	Data type	Initial Value	AT	Constant	Comment
MasterAxis	_sAXIS_REF		MC://_MC_AX[0]	TRUE	
SlaveAxis	_sAXIS_REF		MC://_MC_AX[1]	TRUE	
SlaveAxis2	_sAXIS_REF		MC://_MC_AX[2]	TRUE	

Set Value

- **Setting *MasterCfg* (Data Type: sMASTER_CFG)**

Name	Data Type	Setting Value	Comment
TriggerInput	_sTRIGGER_REF	Refer to *Setting TriggerInput* on page 90	-
CmdPosMode	BOOL	FALSE	The type, which is referred to as the master axis, is specified as the feedback value.
EnableAdvanceAngleCorr	BOOL	FALSE	The Advance Angle Compensation function is disabled.
AdvanceAngleCorrValue	LREAL	0.0	-

- **Setting *MasterCfg.TriggerInput* (Data Type:_sTRIGGER_REF)**

Name	Data Type	Setting Value	Comment
Mode	ENUM	_mcDrive	Trigger Mode
LatchID	ENUM	_mcLatch1	Latch ID
InputDrive	ENUM	_mcEXT	External Input

- **Setting *SlaveCfg* (Data Type: sSLAVE_CFG)**

Name	Data Type	Setting Value	Comment
SyncPos	LREAL	155.0	SlaveSyncPos
EndPos	LREAL	10.0	SlaveEndPos
WaitPos	LREAL	100.0	SlaveWaitPos
MaxVelocity	LREAL	100000.0	MaxVelocity
Acceleration	LREAL	100000.0	SlaveAcceleration
Deceleration	LREAL	100000.0	SlaveDeceleration
BladeNum	UDINT	2	Number of Blade
BladeMountErrorComp	ARRAY[1..4] OF LREAL	4(0.0)	-
ProfileType	UDINT	0	Cam Profile Type
VelocityToWaitPos1	LREAL	360.0	Velocity to SlaveWaitPos
VelocityToWaitPos2	LREAL	360.0	
VelFactor	UDINT	100	Override Velocity Factor
StopMode	_eMC_STOP_MODE	_emc_STOP _MODE# _mcDecceleration Stop	

- **Setting *CuttingCfg* (Data Type: sCUTTING_CFG)**

Name	Data Type	Setting Value	Comment
Mode	UDINT	2	Operation Mode
CutLength	LREAL	50.0	Ideal mark pitch
ToleranceForMark	LREAL	0.0	Tolerance for mark detection
WindowForGap	LREAL	0.0	
MasterSyncPos	LREAL	0.0	Use only in continuous mode
DistanceToMarkSensor	LREAL	130.0	Distance between the mark sensor and the blade
EnableNGNS	BOOL	FALSE	
DistanceToGapSensor	LREAL	0.0	
OffsetFromMark	LREAL	0.0	
SkipCountLimit	UDINT	0	

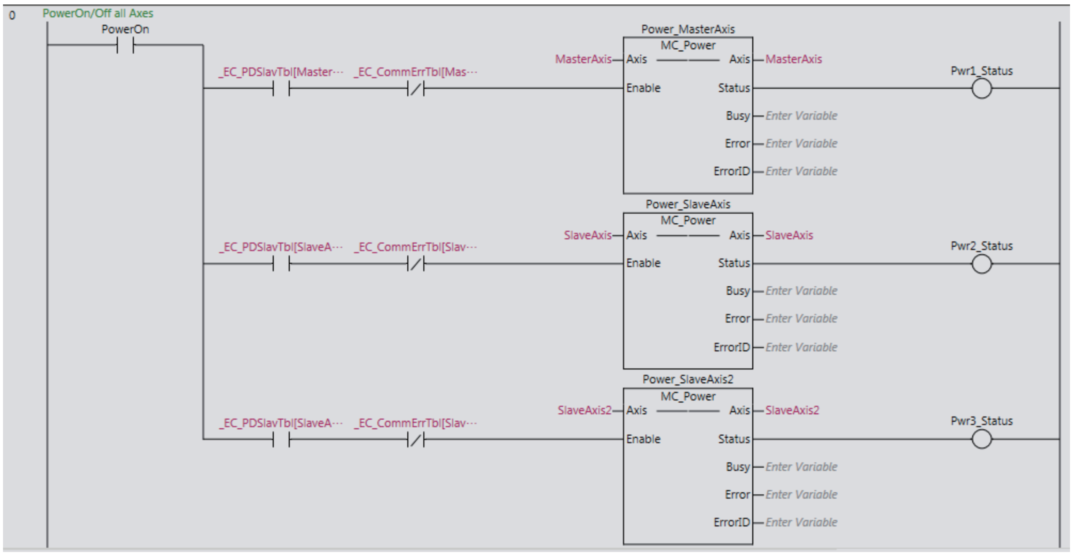
- **Setting *CuttingCfg2* (Data Type: sCUTTING_CFG)**

Name	Data Type	Setting Value	Comment
Mode	UDINT	2	Operation Mode
CutLength	LREAL	50.0	Ideal mark pitch
ToleranceForMark	LREAL	0.0	Tolerance for mark detection
WindowForGap	LREAL	0.0	
MasterSyncPos	LREAL	0.0	Use only in continuous mode
DistanceToMarkSensor	LREAL	300.0	Distance between the mark sensor and the blade
EnableNGNS	BOOL	FALSE	
DistanceToGapSensor	LREAL	0.0	
OffsetFromMark	LREAL	0.0	
SkipCountLimit	UDINT	0	

Ladder Diagram

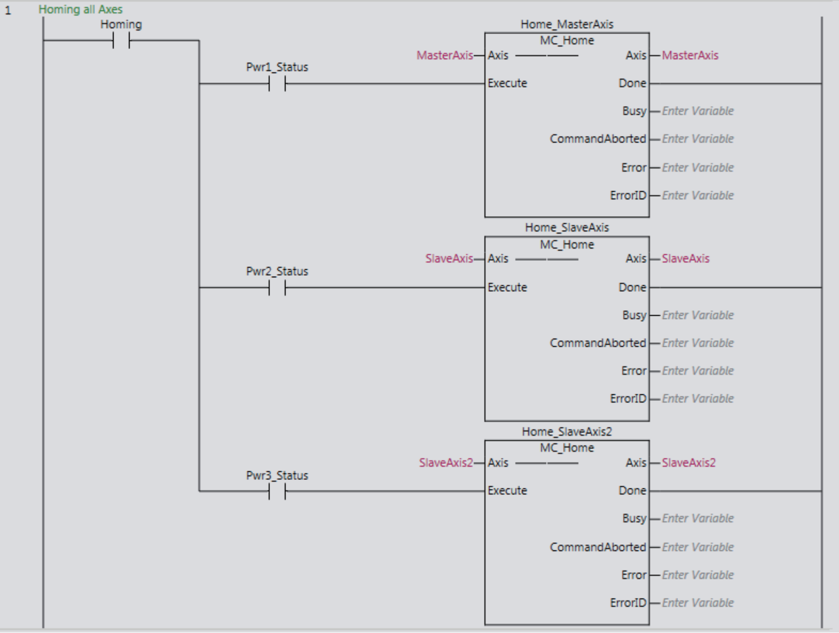
A sample program (Program0) with RotaryKnife.

1



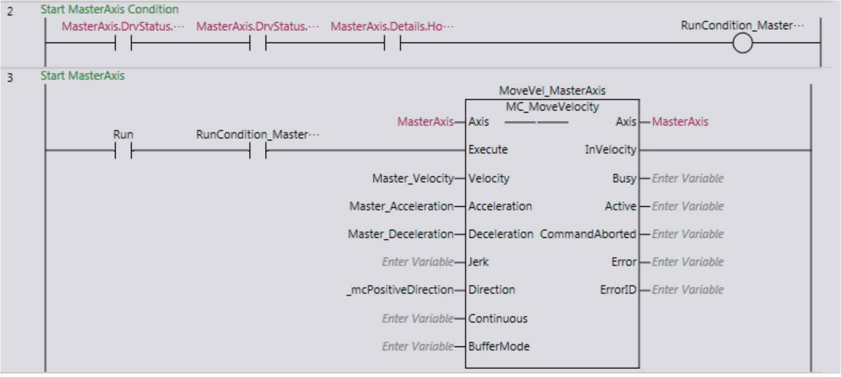
Servo on MasterAxis and SlaveAxis.

2



Execute homing of MasterAxis and SlaveAxis.

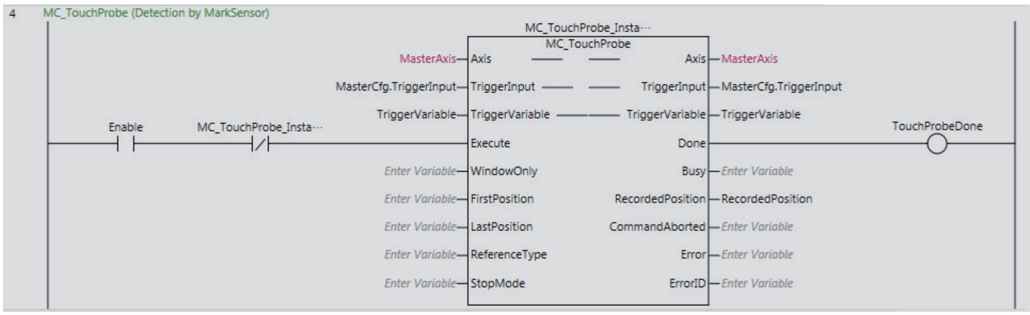
3



Line2
Confirm the condition for starting master axis.

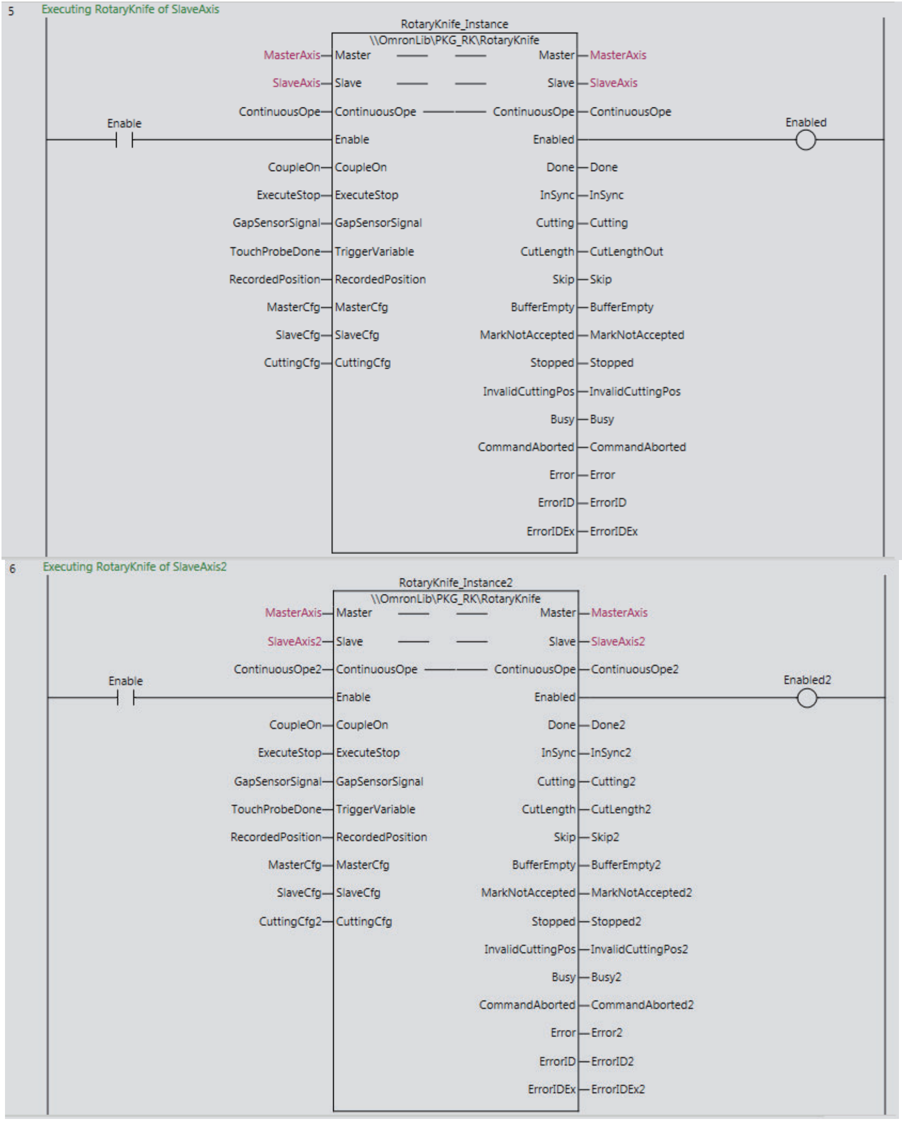
Line3
Start MasterAxis. Run MasterAxis with MC_MoveVelocity function block.

4



Line4
Allocate MC_TouchProbe function block.
This FB is used to obtain a position of the MasterAxis as a signal from the mark sensor is set to trigger, to obtain the position of the MasterAxis.
Input the obtained position of the MasterAxis in the *RecordedPosition* of the RotaryKnife.

5

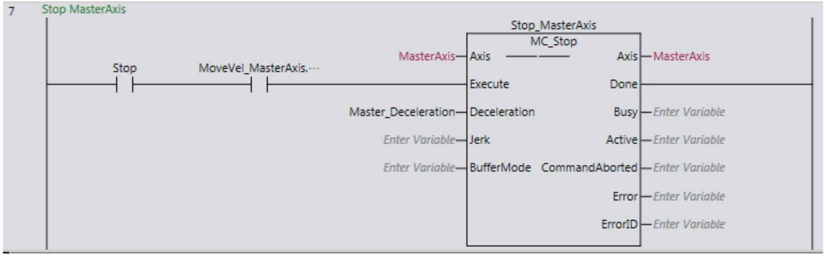

5 Executing RotaryKnife of SlaveAxis
RotaryKnife_Instance
\\OmronLib\PKG_RK\RotaryKnife
MasterAxis—Master — Master—MasterAxis
SlaveAxis—Slave — Slave—SlaveAxis
ContinuousOpe—ContinuousOpe — ContinuousOpe—ContinuousOpe
Enable
Enable
Enabled
Enabled
CoupleOn—CoupleOn
Done—Done
ExecuteStop—ExecuteStop
InSync—InSync
GapSensorSignal—GapSensorSignal
Cutting—Cutting
TouchProbeDone—TriggerVariable
CutLength—CutLengthOut
RecordedPosition—RecordedPosition
Skip—Skip
MasterCfg—MasterCfg
BufferEmpty—BufferEmpty
SlaveCfg—SlaveCfg
MarkNotAccepted—MarkNotAccepted
CuttingCfg—CuttingCfg
Stopped—Stopped
InvalidCuttingPos—InvalidCuttingPos
Busy—Busy
CommandAborted—CommandAborted
Error—Error
ErrorID—ErrorID
ErrorIDEx—ErrorIDEx
6 Executing RotaryKnife of SlaveAxis2
RotaryKnife_Instance2
\\OmronLib\PKG_RK\RotaryKnife
MasterAxis—Master — Master—MasterAxis
SlaveAxis2—Slave — Slave—SlaveAxis2
ContinuousOpe2—ContinuousOpe — ContinuousOpe—ContinuousOpe2
Enable
Enable
Enabled2
Enabled
CoupleOn—CoupleOn
Done—Done2
ExecuteStop—ExecuteStop
InSync—InSync2
GapSensorSignal—GapSensorSignal
Cutting—Cutting2
TouchProbeDone—TriggerVariable
CutLength—CutLength2
RecordedPosition—RecordedPosition
Skip—Skip2
MasterCfg—MasterCfg
BufferEmpty—BufferEmpty2
SlaveCfg—SlaveCfg
MarkNotAccepted—MarkNotAccepted2
CuttingCfg2—CuttingCfg
Stopped—Stopped2
InvalidCuttingPos—InvalidCuttingPos2
Busy—Busy2
CommandAborted—CommandAborted2
Error—Error2
ErrorID—ErrorID2
ErrorIDEx—ErrorIDEx2

Line5, 6

Allocate instances of RotaryKnife for both SlaveAxes.

Input position of MasterAxes obtained with MC_TouchProbe function block in the *RecordedPosition* of each instance.

6



The value of the [Stop] to TRUE, to stop the main axis.

⚠ WARNING

When using this FB in a device, confirm that the program and FB operates properly. Design a program so that safety measures such as fail-safe circuits are implemented outside of the FB.

Operation Procedure

The operation procedure to execute RotaryKnife Library.

Please refer to *Operation procedure* on page 123.

Winder/Unwinder

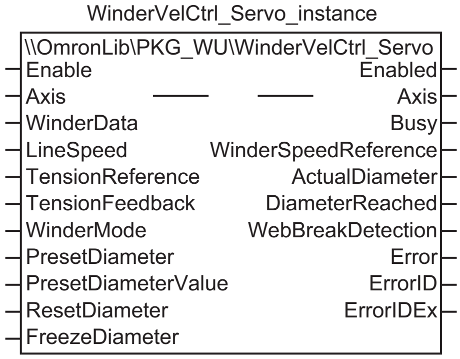
Function block name	Name	Page
WinderVelCtrl_Servo	Velocity Control Winder (for Servo)	P.136
WinderVelCtrl_Inverter	Velocity Control Winder (For Inverter)	P.173

WinderVelCtrl_Servo

WinderVelCtrl_Servo is a function block that controls that controls the tension of a continuous material being wound or unwound with servo motor.

This function block can be used for the following applications.

- Wound or unwound materials from a drum or roll.
- Feeding wrapping materials (unwinder) in packaging application.
- Other sheet transfer applications.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
WinderVelCtrl _Servo	Velocity Control Winder (for Servo)	FB		WinderVelCtrl_Servo_instance(Enable, Axis, WinderData, LineSpeed, TensionReference, WinderMode, PresetDiameter, PresetDiameterValue, ResetDiameter, FreezeDiameter, Enabled, Busy, WinderSpeedReference, ActualDiameter, DiameterReached, WebBreakDetection, Error, ErrorID, ErrorIDEx);

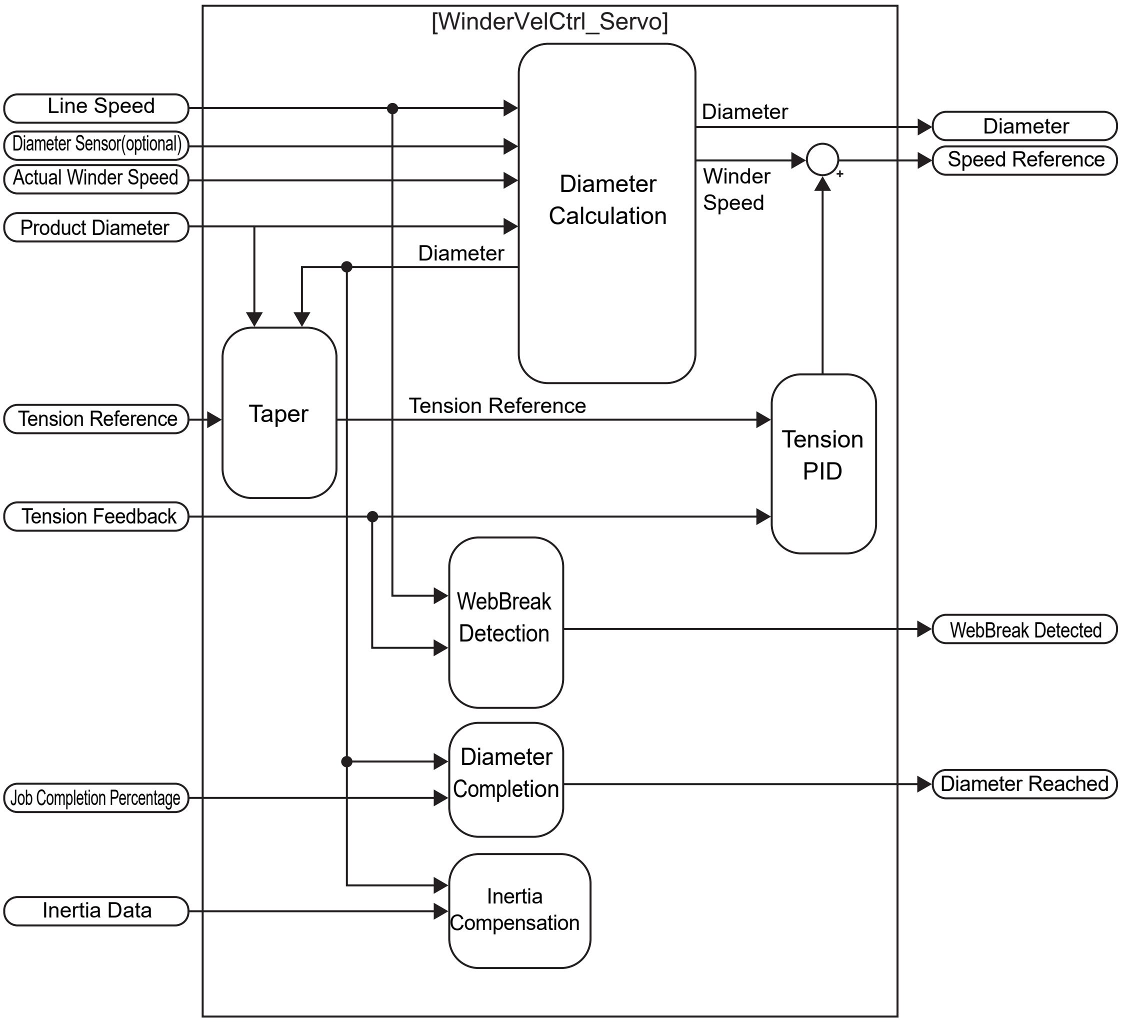
Function Block and Function Information

Item	Description
Library file name	OmronLib_PKG_WU_Vx_x.sir[*1]
Namespace	OmronLib\PKG_WU
Function block and function number	00066
Publish/Do not publish source code	Not Published.
Function block and function version	1.00

[*1] Vx_x indicates version.

Overview of Processing

This section describes the setting values and the internal processing of this function block



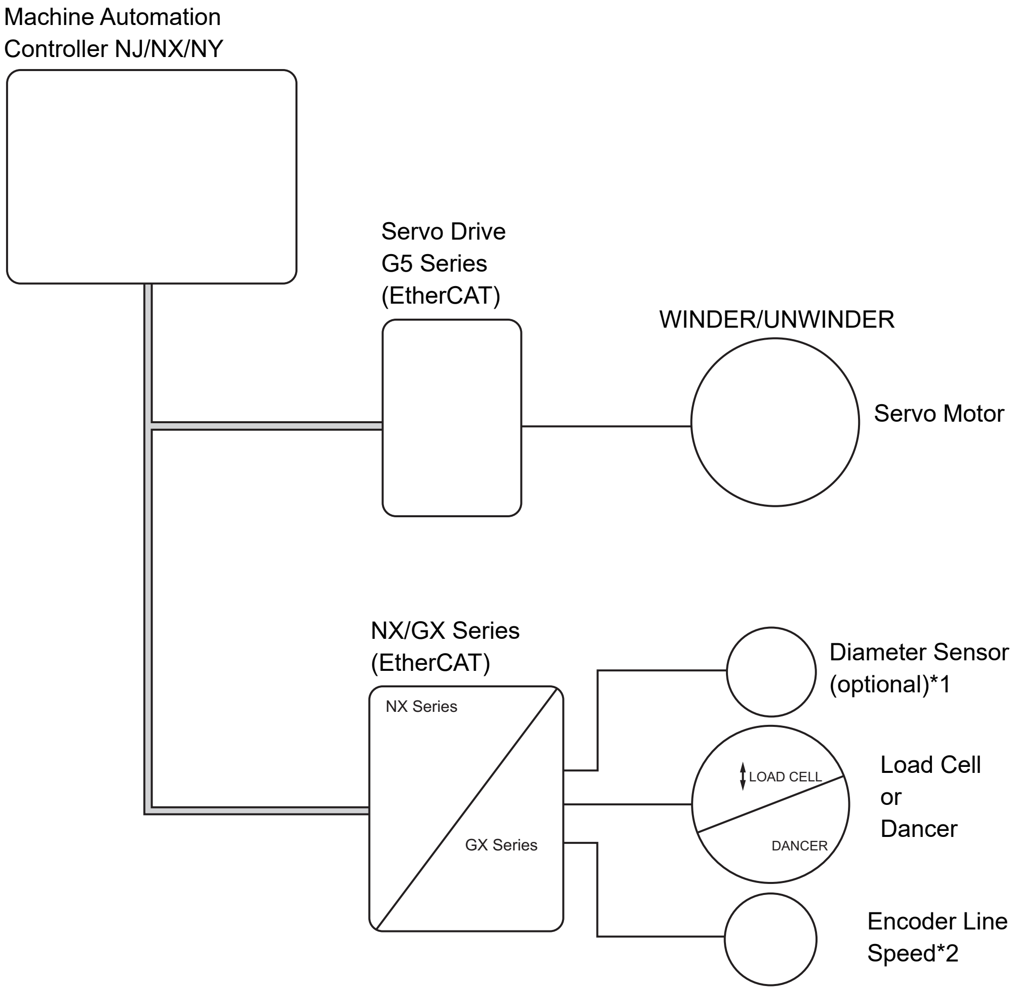
Hardware and Software Configuration

This function block covers the following system configuration.

By G5 Servo Drive

● Hardware Configuration

Winder/Unwinder is driven by G5 servo drive using NX-Series/GX Series I/Os for the diameter sensor, line speed (encoder) and the load cell/dancer.



- The use of a load cell or dancer is necessary for winder speed tension control.

Note 1. Diameter sensor is optional in order to enable the function block to estimate the diameter.

2. Line speed could be provided by analog input or other device trough communications.

- **Software Configuration**

Configure the axis as shown below.

Line Speed Axis Configuration

Classification	Parameter name	Set value
Axis Basic Settings	Axis Type [*1]	1:Encoder axis
Position Count Settings	Count Mode	1: Rotary Mode
	Modulo Maximum Position Setting Value	Negative number, positive number or 0 of LREAL data type[*2]
	Modulo Minimum Position Setting Value	Negative number, positive number or 0 of LREAL data type [*2]

[*1] When using an encoder for line speed.

[*2] Set the value according to the device where this function block is used.

Winder/Unwinder Axis Configuration

Classification	Parameter name	Set value
Axis Basic Settings	Axis Type	0:Servo axis
Position Count Settings	Count Mode	1:Rotary Mode (Infinite length)
	Modulo Maximum Position Setting Value	360 [*1]
	Modulo Minimum Position Setting Value	0 [*1]

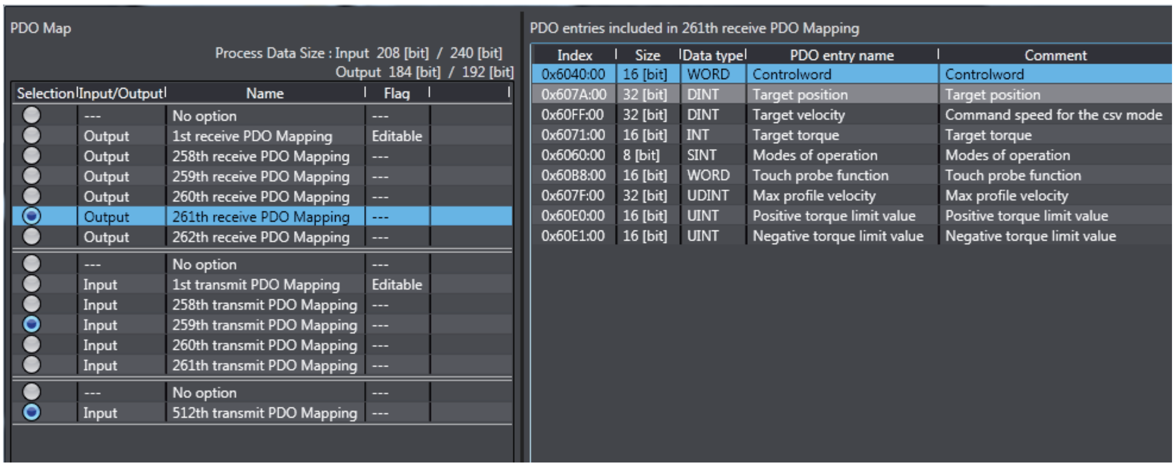
[*1] Set the value according to the device where this function block is used.

⚠ Caution

When using this function block, set the count mode of winding/unwinding axes to rotary mode.
Also, set the rotation direction of the servo drive correctly depending on the machine. If the rotation direction of the motor is not set correctly depending on the machine, unintended operations may occur.

PDO Map settings: use default PDO Map setting.

Output



PDO Map

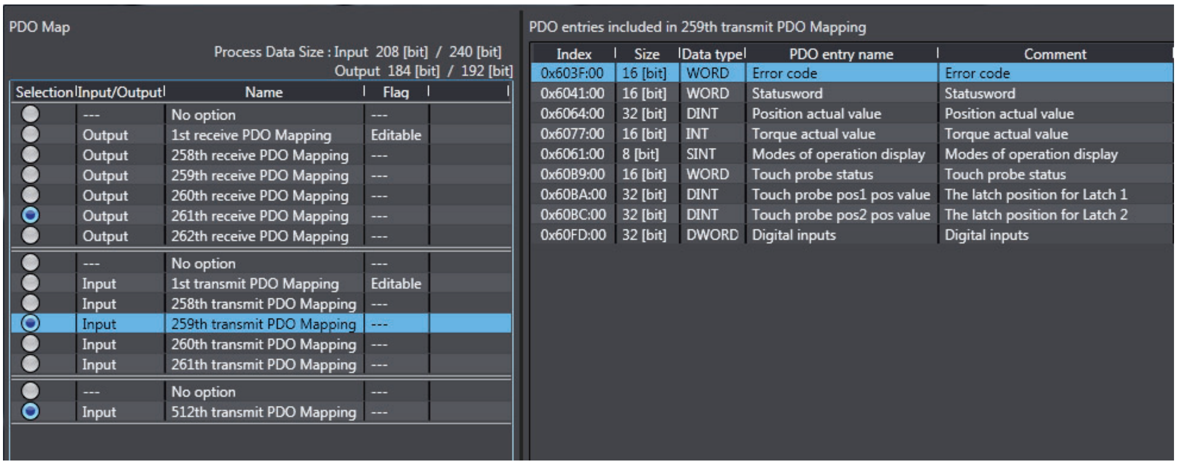
Process Data Size : Input 208 [bit] / 240 [bit]
Output 184 [bit] / 192 [bit]

Selection	Input/Output	Name	Flag
○	---	No option	---
○	Output	1st receive PDO Mapping	Editable
○	Output	258th receive PDO Mapping	---
○	Output	259th receive PDO Mapping	---
○	Output	260th receive PDO Mapping	---
●	Output	261th receive PDO Mapping	---
○	Output	262th receive PDO Mapping	---
○	---	No option	---
○	Input	1st transmit PDO Mapping	Editable
○	Input	258th transmit PDO Mapping	---
●	Input	259th transmit PDO Mapping	---
○	Input	260th transmit PDO Mapping	---
○	Input	261th transmit PDO Mapping	---
○	---	No option	---
●	Input	512th transmit PDO Mapping	---

PDO entries included in 261th receive PDO Mapping

Index	Size	Data type	PDO entry name	Comment
0x6040:00	16 [bit]	WORD	Controlword	Controlword
0x607A:00	32 [bit]	DINT	Target position	Target position
0x60FF:00	32 [bit]	DINT	Target velocity	Command speed for the csv mode
0x6071:00	16 [bit]	INT	Target torque	Target torque
0x6060:00	8 [bit]	SINT	Modes of operation	Modes of operation
0x60B8:00	16 [bit]	WORD	Touch probe function	Touch probe function
0x607F:00	32 [bit]	UDINT	Max profile velocity	Max profile velocity
0x60E0:00	16 [bit]	UINT	Positive torque limit value	Positive torque limit value
0x60E1:00	16 [bit]	UINT	Negative torque limit value	Negative torque limit value

Input



PDO Map

Process Data Size : Input 208 [bit] / 240 [bit]
Output 184 [bit] / 192 [bit]

Selection	Input/Output	Name	Flag
○	---	No option	---
○	Output	1st receive PDO Mapping	Editable
○	Output	258th receive PDO Mapping	---
○	Output	259th receive PDO Mapping	---
○	Output	260th receive PDO Mapping	---
●	Output	261th receive PDO Mapping	---
○	Output	262th receive PDO Mapping	---
○	---	No option	---
○	Input	1st transmit PDO Mapping	Editable
○	Input	258th transmit PDO Mapping	---
●	Input	259th transmit PDO Mapping	---
○	Input	260th transmit PDO Mapping	---
○	Input	261th transmit PDO Mapping	---
○	---	No option	---
●	Input	512th transmit PDO Mapping	---

PDO entries included in 259th transmit PDO Mapping

Index	Size	Data type	PDO entry name	Comment
0x603F:00	16 [bit]	WORD	Error code	Error code
0x6041:00	16 [bit]	WORD	Statusword	Statusword
0x6064:00	32 [bit]	DINT	Position actual value	Position actual value
0x6077:00	16 [bit]	INT	Torque actual value	Torque actual value
0x6061:00	8 [bit]	SINT	Modes of operation display	Modes of operation display
0x60B9:00	16 [bit]	WORD	Touch probe status	Touch probe status
0x60BA:00	32 [bit]	DINT	Touch probe pos1 pos value	The latch position for Latch 1
0x60BC:00	32 [bit]	DINT	Touch probe pos2 pos value	The latch position for Latch 2
0x60FD:00	32 [bit]	DWORD	Digital inputs	Digital inputs

Usage Conditions

Conditions for Execution

This section describes the execution conditions of this function block.

● Execution Conditions of the function block

Following condition must be met:

- While *Enable* is TRUE, the G5 servo drive must be turned ON.
- There must be no error in the G5 servo drive.
- When *Enable* changes from FALSE to TRUE, the Winder/Unwinder must be stopped and line speed must be 0.

These conditions are not detected as errors and must be considered as safety measures in external control circuits.

● Condition of Task Settings

It is recommended that you execute this function block in the primary periodic task.

Precautions for Correct Use

If this FB is not executed in the primary periodic task, specify *task time* in *TaskTime* setting for the PID.

Other Settings

● Setting Rotation Direction of the Motor

When using this function block, confirm the rotation direction setting of the servo drive according to the rotation direction of the Winder/Unwinder.
Verify that motor rotation direction with FW signal is clockwise (CW) in winder mode.
Configure the following servo parameter object.

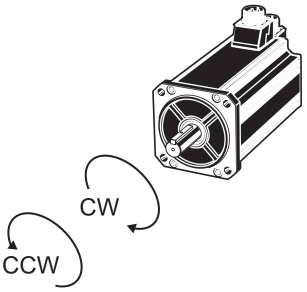
3000hex	Rotation direction Switching						All
Setting range	0 to 1	Unit	---	Default setting	1	Data attribute	C
Size	2 bytes (INT16)		Access	RW	PDO map	Not possible	

- Motor rotation direction to the position command, speed command, and torque command can be switched.

Explanation of Set Values

Set value	Description
0	A forward direction command sets the motor rotation direction to clockwise.
1	A forward direction command sets the motor rotation direction to counterclockwise.

- The motor rotation direction when viewing the shaft from the load side is called clockwise (CW) or counterclockwise (CCW).



For details, refer to the *AC Servomotors/Servo Drivers G5-series with Built-in EtherCAT Communications User's Manual* (Cat.No. I576)

Variables

Input Variables

Name	Data type	Default	Valid range	Change over during execution	Description
Enable	BOOL	FALSE	TRUE/FALSE	Yes	When *Enable* changes from FALSE to TRUE, this function block is executed. When *Enable* changes from TRUE to FALSE, this function block stops.
WinderData	sWU_SERVO	Refer to *sWU_SERVO* on page 143.	---	---	Winder configuration.
LineSpeed	LREAL	0m/min	0m/min to 2,000m/min	Yes	Actual Line Speed
Tension Reference	LREAL	0%	0% to 100%	Yes	Tension Reference
Tension Feedback	LREAL	0%	0% to 100%	Yes	Tension Feedback
WinderMode	BOOL	FALSE	TRUE/FALSE	No	Winder Mode. FALSE: Winder TRUE: Unwinder
Preset Diameter	BOOL	FALSE	TRUE/FALSE	Yes	Preset Input Function
Preset DiameterValue	LREAL	40mm	Minimum diameter to product diameter.	Yes	Preset Diameter Value
Reset Diameter	BOOL	FALSE	TRUE/FALSE	Yes	Reset Input Function
Freeze Diameter	BOOL	FALSE	TRUE/FALSE	Yes	Freeze Input Function

Output Variables

Name	Data type	Description
Enabled	BOOL	TRUE when output is enable.
Busy	BOOL	TRUE during execution. When *Error* changes to TRUE, *Busy* becomes FALSE. *Busy* is FALSE during non-execution status.
WinderSpeed Reference	LREAL	Winder/Unwinder Speed Reference in user units
ActualDiameter	LREAL	Current Winder/Unwinder diameter in mm
DiameterReached	BOOL	TRUE after the current winder diameter value reaches *PercentageJobCompletion* setting value and *TimeJobCompletion* setting time passes.
WebBreakDetection	BOOL	TRUE if a web break has been detected.
Error	BOOL	TRUE if an error occurs.
ErrorID	WORD	Outputs the error code when an error occurs. #0000 is the initial value and indicated as normal end.
ErrorIDEx	DWORD	Outputs the error code when an error occurs. #00000000 is the initial value and indicated as normal end.

In-Out Variables

Name	Data type	Default	Valid range	Change over during execution	Description
Axis	_sAxis_REF	---	---	No	Winder/Unwinder Axis

Structures

● sWU_SERVO

Name	Data type	Default	Valid range	Change over during execution	Description
BasicData	sWU_BASIC_ DATA	Refer to *sWU_BASIC_DATA* on page 144.	---	---	---
Config	sWU_CONFIG	Refer to *sWU_CONFIG* on page 144.	---	---	---
Command	sWU_ COMMAND	Refer to *sWU_COMMAND* on page 145.	---	---	---
PID	sWU_PID	Refer to *sWU_PID* on page 145.	---	---	---
InertiaData	sWU_ INERTIA_DATA	Refer to *sWU_INERTIA_DATA* on page 145.	---	---	---
Taper	sWU_TAPER	Refer to *sWU_TAPER* on page 145.	---	---	---
WebBreak	sWU_ WEBBREAK	Refer to *sWU_WEBBREAK* on page 146.	---	---	---

sWU_BASIC_DATA

Name	Data type	Default	Valid range	Change over during execution	Description
ReelDiameter	LREAL	40.0mm	1.0mm to 2,000.0mm	No	Reel Diameter in mm
MidDiameter	LREAL	100mm	Greater than Reel Diameter and lower than Product Diameter	Yes	Middle diameter in mm for Taper function
MaxDiameter	LREAL	300mm	Greater or equal to Product Diameter. (Maximum value = [ReelDiameter*MaxRatio(20)])	No	Maximum Diameter in mm
ProductDiameter	LREAL	300mm	MidDiameter < *ProductDiameter* ≤ MaxDiameter	Yes	Product Diameter in mm
MaxLineSpeed	LREAL	25m/min	0 m/min < MaxLineSpeed ≤ 2,000m/min	No	Maximum Line Speed in m/min
ActualMotor Speed (Reserved)	LREAL	---	---	---	Reserved
MaxMotorSpeed	LREAL	--- (user units)	Positive Number (bigger than 0)	No	Maximum Motor speed in user units.

sWU_CONFIG

Name	Data type	Default	Valid range	Change over during execution	Description
SensorMode	BOOL	FALSE	TRUE/FALSE	No	Diameter sensor use
DiameterSensor	LREAL	0	Positive number (MinSensorValue to MaxSensorValue)	Yes	Diameter Sensor value
MinSensorValue	LREAL	0	Positive Number (Less than MaxSensorValue)	No	Minimum Diameter sensor value
MaxSensorValue	LREAL	1,000	Greater than min sensor value	No	Maximum Diameter Sensor value
VarianceSize	DINT	100	0 to 5,000	No	Variance Size for diameter calculation
FilterTime1	TIME	0.15s	(Less or equal to FilterTime2) From 0 to 600.0s	Yes	Filter Time 1 for diameter calculation
FilterTime2	TIME	0.5s	From FilterTime1 to 600.0 s	Yes	Filter Time 2 for diameter calculation
PercentageJobCompletion	LREAL	95.0%	0 to 100%	Yes	Job Completion percentage
TimeJob Completion	TIME	0.5s	0 to 600.0 s	Yes	Job Completion Time

● sWU_COMMAND

Name	Data type	Default	Valid range	Change over during execution	Description
RunForward	BOOL	FALSE	TRUE/FALSE	Yes	Run Forward command
RunReverse	BOOL	FALSE	TRUE/FALSE	Yes	Run Reverse command

● sWU_PID

Name	Data type	Default	Valid range	Change over during execution	Description
ProportionalMin Diameter	LREAL	1.0	0.01 to 1,000.0	Yes	PID Proportional Value at minimum diameter
Proportional ProductDiameter	LREAL	1.0	0.01 to 1,000.0	Yes	PID Proportional Value at Product Diameter
IntegralMin Diameter	LREAL	1.0	0.0 to 10,000	Yes	PID Integral Value at minimum Diameter
IntegralProduct Diameter	LREAL	1.0	0.0 to 10,000	Yes	PID Integral Value at Product Diameter
Derivative	LREAL	0.0	0.0 to 10,000.0	Yes	PID Derivative value
ConvergenceGain	LREAL	1.0	0.1 to 5.0	Yes	To give smooth operation near the setpoint.
TaskTime	LREAL	0.001	0.0005 to 0.1	No	Task Time in seconds.
PIDOutputScale	LREAL	1.0	0.0 to 1.0	Yes	PID Output scale factor.
PIDUpperLimit	LREAL	100	0 to 100%	Yes	PID Upper Limit in %
PIDLowerLimit	LREAL	-100	-100 to 0%	Yes	PID Lower Limit in %

● sWU_INERTIA_DATA

Name	Data type	Default	Valid range	Change over during execution	Description
Enable	BOOL	FALSE	TRUE/FALSE	Yes	Enable/DisableInertia Compensation
ReelMass	LREAL	0.0	Greater or equal to 0.0	No	Reel Mass in Kg
ProductDensity	LREAL	0.0	Greater or equal to 0.0	Yes	Product density in Kg/m^3
ProductMass	LREAL	0.0	Greater or equal to 0.0	Yes	Product Mass in Kg
ProductWidth	LREAL	0.0	Greater or equal to 0.0	Yes	Product width in mm
GearRatio	LREAL	1.0	Greater or equal to 1.0	No	From Motor to Winder/Unwinder
MotorInertia	LREAL	0.0	Greater than 0.0	No	Motor inertia in Kg/m^2
GearInertia	LREAL	0.0	Greater or equal to 0.0	No	Gear inertia in Kg/m^2

● sWU_TAPER

Name	Data type	Default	Valid range	Change over during execution	Description
TensionMin Diameter	LREAL	100.0%	0.0% to 200.0%	Yes	Tension at minimum diameter
TensionMid Diameter	LREAL	100.0%	0.0% to 200.0%	Yes	Tension at middle diameter
TensionProduct Diameter	LREAL	100.0%	0.0% to 200.0%	Yes	Tension at Product diameter

- **sWU_WEBBREAK**

Name	Data type	Default	Valid range	Change over during execution	Description
Enable	BOOL	FALSE	TRUE/FALSE	Yes	Enable web break detection
MinLine Speed	LREAL	80.0m/min	0 to MaxLineSpeed	Yes	Minimum Line Speed for web break detection
UpperLimit	LREAL	80.0%	Greater than Lower Limit (0% to 100%)	Yes	Upper Limit level for web break detection
LowerLimit	LREAL	10%	Lower than UpperLimit (0% to 100%)	Yes	Lower Limit for web break detection
WebBreak Time	TIME	1.0s	0.1 s to 10.0 s	Yes	Web break detection time

Function

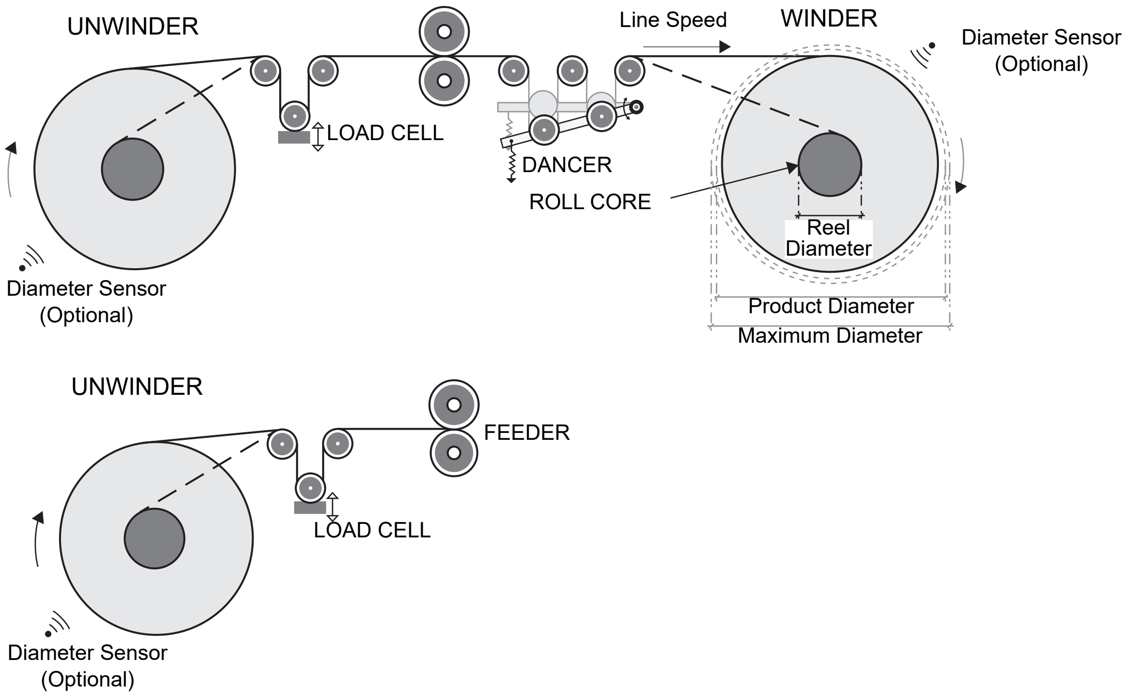
Application Configuration

This section describes an application example of this function block.

The figure below shows an example of a process with winders.

To control the web tension, a load cell or dancer is needed.

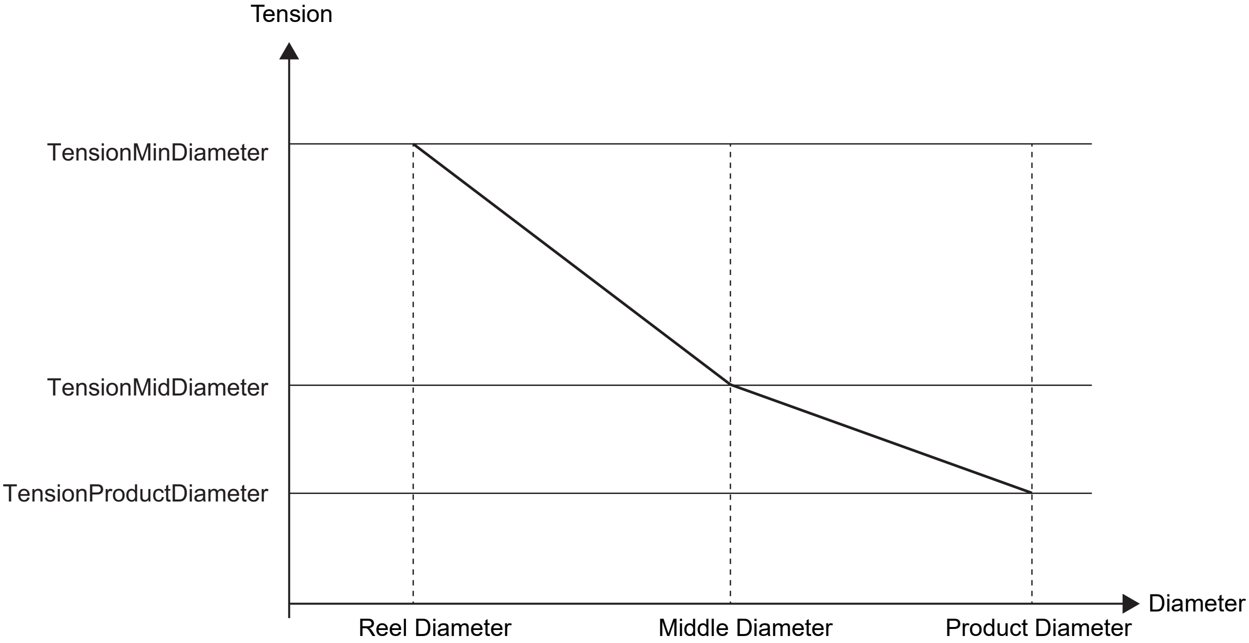
The diameter sensor is optional in order to enable the function block to estimate the roll diameter.



Functions and features of WinderVelCtrl_Servo

● Taper

This section describes the Taper function.
It provides for compensation of the tension reference relating to diameter and helps to avoid defects like telescoping and crushed rolls.
Set the desired tension reference for the three defined diameters, where *sWU_TAPER.TensionMinDiameter* is the Taper tension reference for reel diameter (*sWU_BASIC_DATA.ReelDiameter*), *sWU_TAPER.TensionMidDiameter* is the Taper tension for middle diameter (*sWU_BASIC_DATA.MidDiameter*) and *sWU_TAPER.TensionProductDiameter* is the Taper tension for Product diameter (*sWU_BASIC_DATA.ProductDiameter*)



Refer to the related parameters:

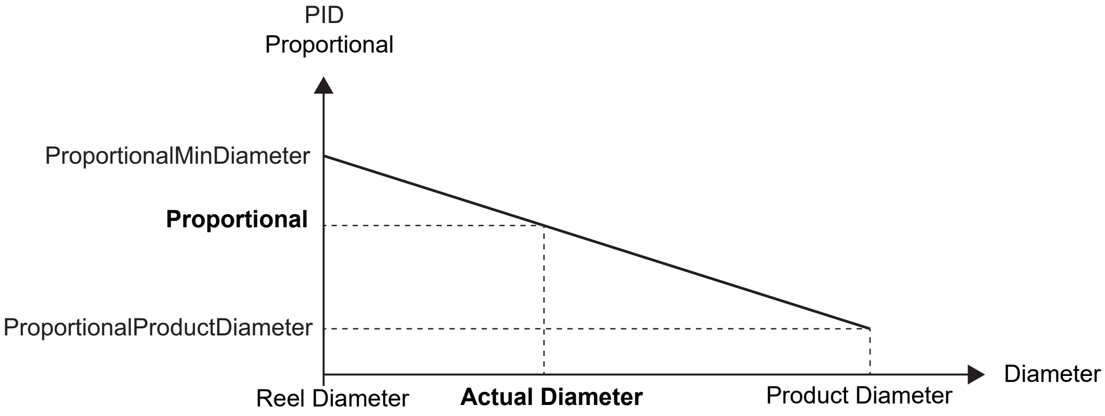
Variable Name	Data Type	Initial Value	Valid Range	Change over during execution (*Enable*=TRUE)	Comment
sWU_BASIC_DATA .ReelDiameter	LREAL	40.0mm	1.0 to 2,000.0mm	No	Reel Diameter in mm
sWU_BASIC_DATA .MidDiameter	LREAL	100mm	Greater than Reel Diameter and lower than Product Diameter	Yes	Middle diameter in mm for Taper function
sWU_BASIC_DATA .ProductDiameter	LREAL	300mm	MidDiameter < *ProductDiameter* ≤ MaxDiameter	Yes	Product Diameter in mm
sWU_TAPER.Tension MinDiameter	LREAL	100.0%	0.0% to 200.0%	Yes	Tension at minimum diameter
sWU_TAPER .Tension MidDiameter	LREAL	100.0%	0.0% to 200.0%	Yes	Tension at middle diameter
sWU_TAPER .Tension ProductDiameter	LREAL	100.0%	0.0% to 200.0%	Yes	Tension at Product diameter

● Variable PID

This section describes the Variable PID function.

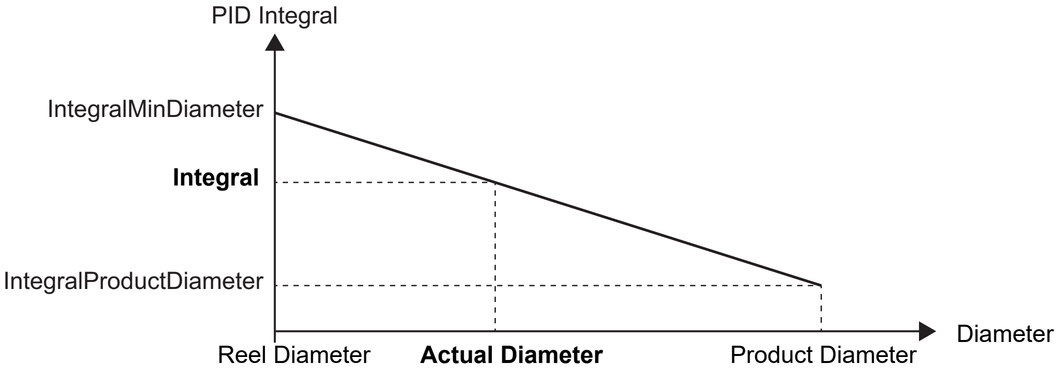
This function modifies Proportional and Integral PID terms relating to the diameter. For PID proportional term, set the proportional value for minimum diameter (reel diameter) in *sWU_PID.ProportinalMinDiameter* and the proportional value for product diameter in *sWU_PID.ProportionalProductDiameter*.

Variable PID - Proportional



For PID Integral term, set the integral value for minimum diameter (reel diameter) in *sWU_PID.IntegralMinDiameter* and the integral value for product diameter in *sWU_PID.IntegralProductDiameter*.

Variable PID – Integral



Derivative term is not modified.

Refer to the related parameters:

Variable Name	Data Type	Initial Value	Valid Range	Change over during execution (*Enable*=TRUE)	Comment
sWU_BASIC_DATA .ReelDiameter	LREAL	40.0mm	1.0mm to 2,000.0mm	No	Reel Diameter in mm
sWU_ASIC_DATA .ProductDiameter	LREAL	300mm	MidDiameter < *ProductDiameter* ≤ MaxDiameter	Yes	Product Diameter in mm
sWU_PID .ProportionalMin Diameter	LREAL	1.0	0.01 to 1,000.0	Yes	PID Proportional Value at minimum diameter
sWU_PID .ProportionalProduct Diameter	LREAL	1.0	0.01 to 1,000.0	Yes	PID Proportional Value at Product Diameter
sWU_PID.Integral MinDiameter	LREAL	1.0	0.0 to 10,000	Yes	PID Integral Value at minimum Diameter
sWU_PID.Integral ProductDiameter	LREAL	1.0	0.0 to 10,000	Yes	PID Integral Value at Product Diameter

● Convergence Error

This section describes Convergence Error function.

It modifies the PID error. It stabilizes the PID error near the setpoint and provides a more aggressive response when the feedback is far from the setpoint as the *sWU_PID.ConvergenceGain* value increases.

When this gain increases, the PID gain can be higher.

When *sWU_PID.ConvergenceGain* is equal to 1.0, the PID Error is not modified.

Refer to the related parameter:

Variable Name	Data Type	Initial Value	Valid Range	Change over during execution (*Enable*=TRUE)	Comment
sWU_PID .ConvergenceGain	LREAL	1.0	0.1 to 5.0	Yes	To provide smooth operation near the setpoint.

● Diameter Completion

This section describes the Diameter Completion function.

DiameterReached output becomes TRUE when the desired diameter is reached. The diameter completion value is set in *sWU_CONFIG.PercentageJobCompletion* in percentage, and diameter completion time in *sWU_CONFIG.TimeJobCompletion*.
This function can be used on winder and unwinder modes.

Refer to the related parameters:

Variable Name	Data Type	Initial Value	Valid Range	Change over during execution (*Enable*=TRUE)	Comment
sWU_CONFIG .PercentageJob Completion	LREAL	95.0%	0 to 100%	Yes	Job Completion percentage
sWU_CONFIG.Time JobCompletion	TIME	0.5s	0 to 600.0 s	Yes	Job Completion Time

- ## Web Break Detection

This section describes the Web Break Detection function.

It detects a web break (material loss) setting *WebBreakDetection* output to TRUE when the function is enabled, line speed is over a specified line speed level and tension feedback from load cell/dancer is out of the specified tension limits during a specific time. Detection result is output in *WebBreakDetection*. Enable the web break function setting *sWU_WEBBREAK.Enable* to TRUE, *set in sWU_WEBBREAK.MinLineSpeed* the minimum line speed and specify the upper web tension limit in *sWU_WEBBREAK.UpperLimit*, lower web tension limit in *sWU_WEBBREAK.LowerLimit* and web break detection time in *sWU_WEBBREAK.WebBreakTime* for web break detection.

Refer to the related parameters:

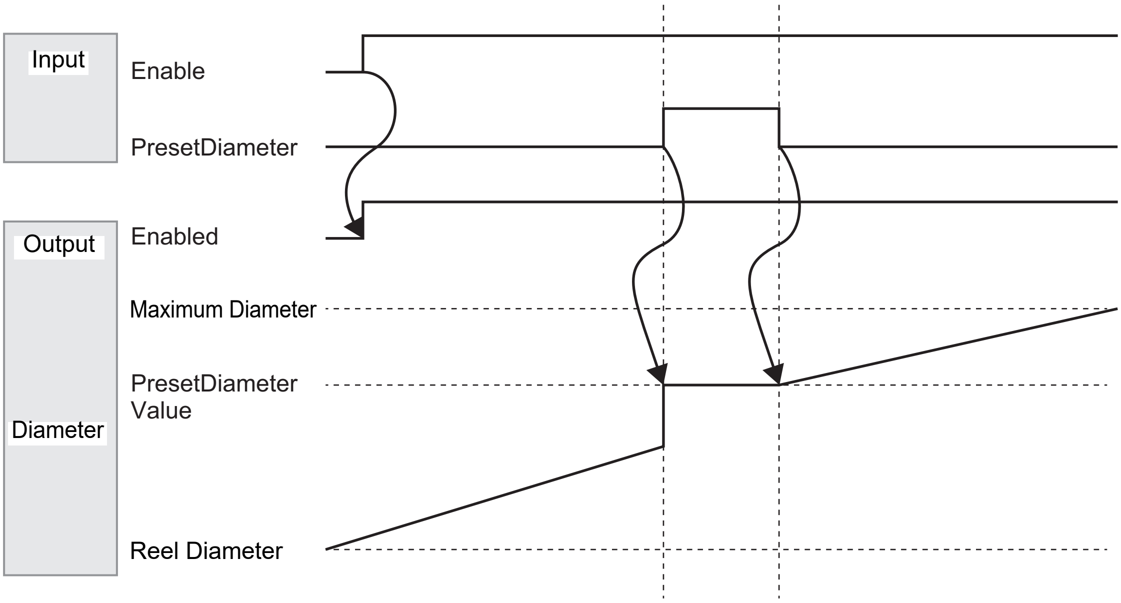
Variable Name	Data Type	Initial Value	Valid Range	Change over during execution (*Enable*=TRUE)	Comment
sWU_WEB BREAK.Enable	BOOL	FALSE	TRUE/FALSE	Yes	Enable/disable web break detection
sWU_WEBBREAK MinimumLineSpeed	LREAL	10.0m/min	0 to MaxLineSpeed	Yes	Minimum Line Speed for web break detection
sWU_WEBBREAK UpperLimit	LREAL	80.0%	Greater than Lower Limit (0% to 100%)	Yes	Upper Limit level for web break detection
sWU_WEBBREAK LowerLimit	LREAL	10%	Lower than UpperLimit (0% to 100%)	Yes	Lower Limit level for web break detection
sWU_WEBBREAK WebBreakTime	TIME	1.0s	0.100 to 10.0 s	Yes	Web Break Detection time

● Preset Diameter

This section describes the Preset Diameter function.

The diameter value is immediately updated to *the PresetValue* when *PresetInput* is set to TRUE. The preset diameter function works only when the Preset Diameter Input (*PresetDiameter*) is set to TRUE. Refer to the following timing diagram:

Timing Diagram (Winder Mode)



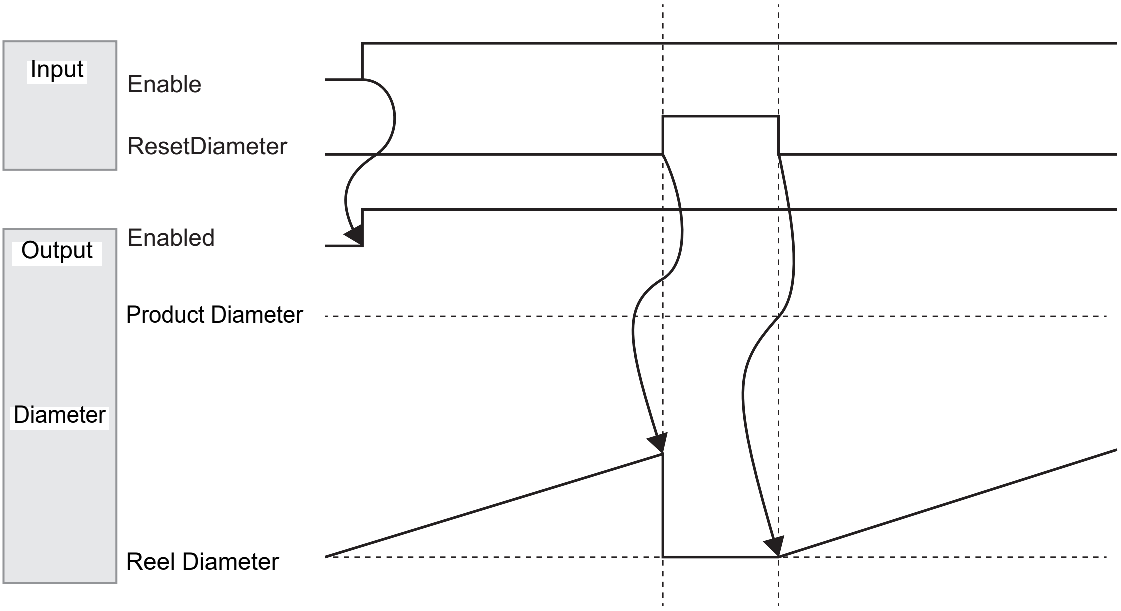
● Reset Diameter

This section describes the Reset Diameter function.

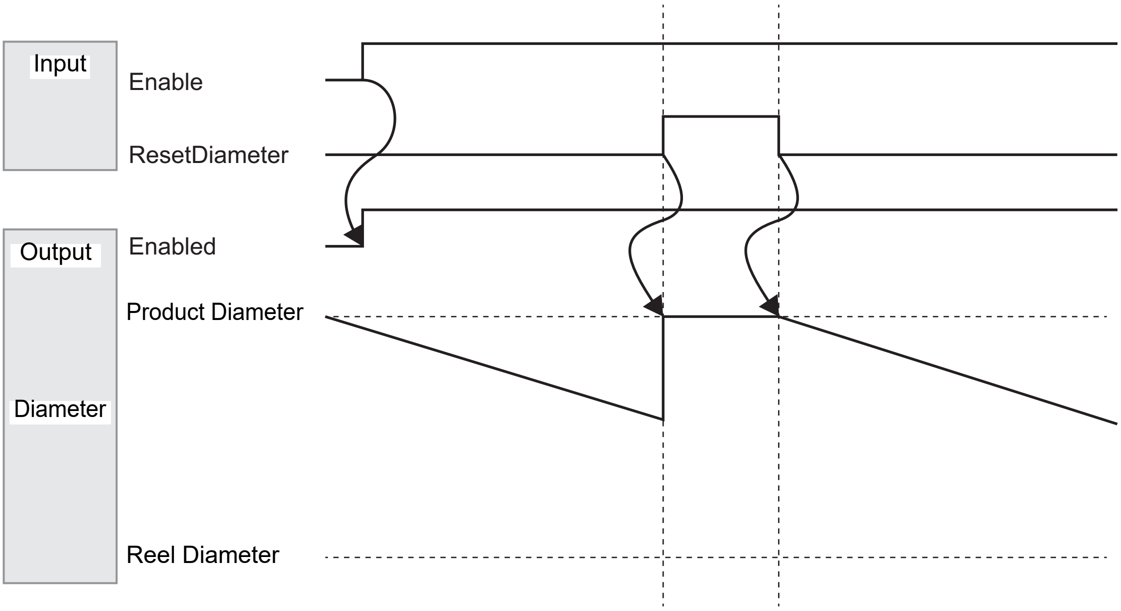
This function forces the diameter value to the initial value of the process in accordance with the winder mode criteria.

When *WinderMode* is set as Winder and the reset diameter input is TRUE, the diameter output will be set to the reel diameter. Also if *WinderMode* is set to Unwinder and the reset diameter input is TRUE, the diameter output will be set to the product diameter (*sWU_BASIC_DATA.ProductDiameter*). The Reset Diameter function works only when the *ResetDiameter* Input is set to TRUE. Refer to the timings diagrams:

Timing Diagram – Winder Mode



Timing Diagram – Unwinder Mode

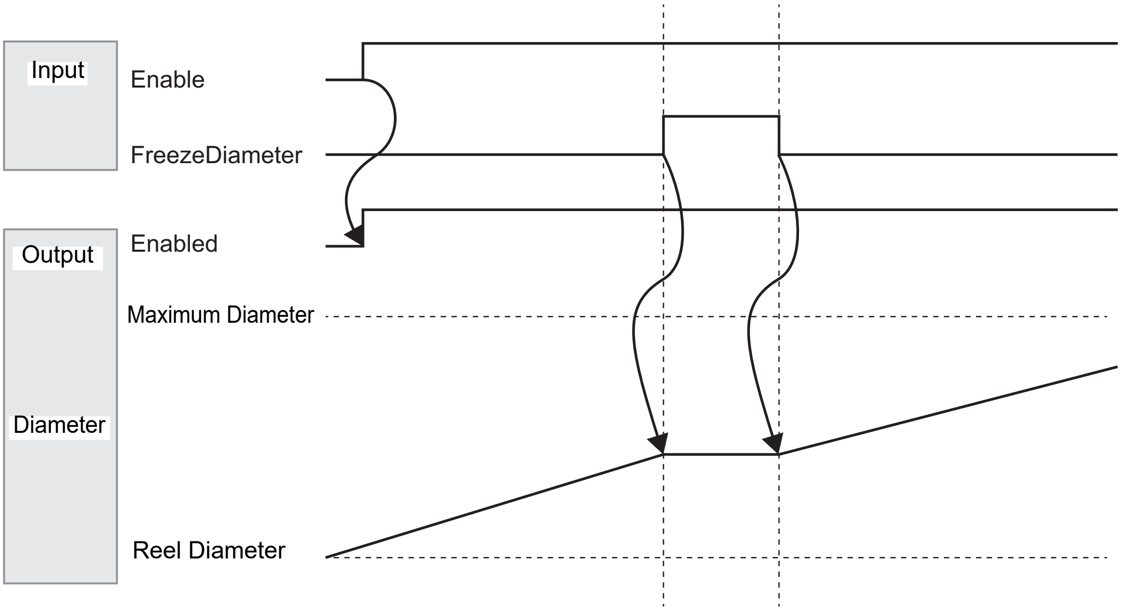


Freeze Diameter

This section describes the Freeze Diameter function.

It freezes the diameter output value when the *FreezeDiameter* input is set to TRUE. *FreezeDiameter* input works only when the *FreezeDiameter* Input is set to TRUE. Refer to the following timing diagram:

Timing Diagram (Winder Mode)



Inertia Compensation (When G5-series Servo Drive is used)

The Inertia Compensation Function calculates an inertia ratio between load inertia and motor inertia, and updates an inertia ratio parameter for G5-series Servo Drive.

Set *sWU_INERTIA_DATA.Enable* to TRUE to enable Inertia compensation. The mass moment of coil inertia can be calculated with product mass (*sWU_INERTIA_DATA.ProductMass*) or with product density (*sWU_INERTIA_DATA.ProductDensity*) and product width (*sWU_INERTIA_DATA.ProductWidth*). Reel mass (*sWU_INERTIA_DATA.ReelMass*), gear ratio (*sWU_INERTIA_DATA.GearRatio*), motor inertia (*sWU_INERTIA_DATA.MotorInertia*) and gear inertia (*sWU_INERTIA_DATA.GearInertia*) must be introduced in both cases.

For detailes, refer to the related parameters:

sWU_INERTIA_DATA

Variable Name	Data Type	Initial Value	Valid Range	Change over during execution (*Enable*=TRUE)	Comment
Enable	BOOL	FALSE	TRUE/FALSE	Yes	Enable/Disable Inertia Compensation Function
ReelMass	LREAL	0.0	Greater or equal to 0.0	No	Reel Mass in Kg
Product Density	LREAL	0.0	Greater or equal to 0.0	Yes	Product density in Kg/m^3
ProductMass	LREAL	0.0	Greater or equal to 0.0	Yes	Product Mass in Kg
ProductWidth	LREAL	0.0	Greater or equal to 0.0	Yes	Product width in mm
GearRatio	LREAL	1.0	Greater or equal to 1.0	No	From Motor to Winder/unwinder
MotorInertia	LREAL	0.0	Greater than 0.0	No	Motor inertia in Kg/m^2
GearInertia	LREAL	0.0	Greater or equal to 0.0	No	Gear inertia in Kg/m^2

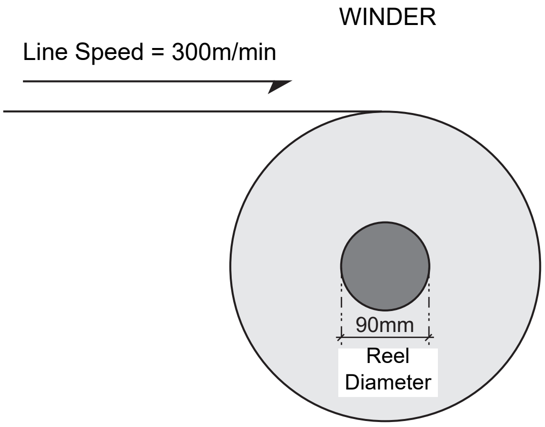
Setting Maximum Winder/Unwinder Speed

Follow next steps before setting the Maximum Speed setting:

1. Calculate Maximum Motor Speed from Reel Diameter and Maximum Line Speed.
2. Set the Maximum Motor Speed which was calculated in procedure 1 in the function block.

Example of a winder:

- Servo Motor = 2,000 rpm
- Reel Diameter = 90mm
- Gear = 2



1 Calculate Maximum Motor Speed with maximum Line Speed and Minimum diameter:

$L = 2\pi r = \pi D = \pi \cdot 90mm = 282.74mm$

$MaxLineSpeed = 300m/min = 300{,}000mm/min$

$$Winder\_Unwinder_{Speed} = \frac{300,000mm/min}{282.74mm} = 1{,}061.04\ rpm$$

$Winder\_Unwinder_{Shaft\_Speed} = Winder\_Unwinder_{Speed} \cdot GearRatio$

$Motor_{Shatf\_Speed} = 1{,}061.04\ rpm \cdot 2 = 2{,}122.09\ rpm$

Convert to *rev*/*sec*:

$$Motor_{Shaft\_Speed} = 2{,}122.09rpm \cdot \frac{1min}{60sec} = 35.36\ rev/sec$$

2 Set Stwinder.BasicData.MaxMotorSpeed = 35.36 *rev*/*sec*

3 Calculate the work travel distance per motor rotation.
WinderSpeedReference is a speed command for the winder expressed in user units.
NJ/NX/NY Axis speed is expressed in seconds.

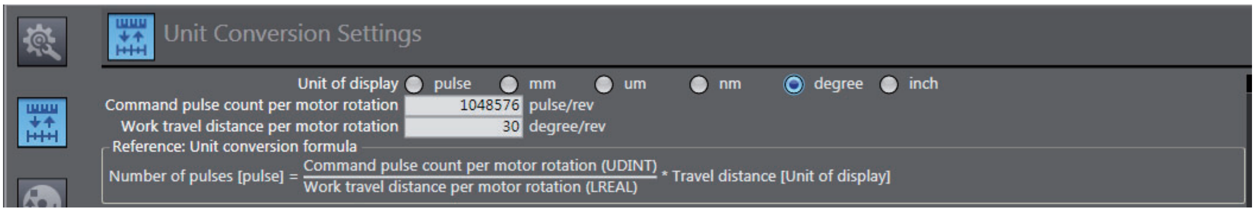
● Convert Speed User Time Units to Seconds.

In the following example, WinderSpeedReference in [rpm] is converted to rotations per second (1 rpm corresponds to 1/60 revolutions per second).

● Set the encoder pulse count per rotation.

G5 Incremental encoder resolution is 20 Bit (2^{20} =1,048,576) pulses) per rotation.

$$\frac{Command\ Pulse\ count\ per\ motor\ rotation}{Work\ travel\ distance\ per\ motor\ rotation} = \frac{2^{20}}{1}$$



Unit Conversion Settings

Unit of display ○ pulse ○ mm ○ um ○ nm ● degree ○ inch

Command pulse count per motor rotation 1048576 pulse/rev

Work travel distance per motor rotation 30 degree/rev

Reference: Unit conversion formula

Number of pulses [pulse] = Command pulse count per motor rotation (UDINT) / Work travel distance per motor rotation (LREAL) * Travel distance [Unit of display]

Unit of display can be set as desired.

Setting Diameter Sensor Configuration

This section describes how to configure the use of a diameter sensor in the winder/unwinder function block.

In this function block, a diameter sensor can be used. If using a diameter sensor in the winder/unwinder application, set *sWU_CONFIG.SensorMode* to TRUE.

It is necessary to specify the minimum sensor value (*sWU_CONFIG.MinSensorValue*) and maximum sensor value (*sWU_CONFIG.MaxSensorValue*) in the function block according to the minimum and maximum winder/unwinder diameter. Set the relevant reel diameter in mm (*sWU_BASIC_DATA.ReelDiameter*). Refer to the related input variables:

- **sWU_BASIC_DATA**

Variable Name	Data Type	Initial Value	Valid Range	Change over during execution (*Enable*=TRUE)	Comment
ReelDiameter	LREAL	40.0mm	1.0mm to 2,000.0mm	No	Reel Diameter in mm

- **sWU_CONFIG**

Variable Name	Data Type	Initial Value	Valid Range	Change over during execution (*Enable*=TRUE)	Comment
SensorMode	BOOL	FALSE	TRUE/FALSE	No	Diameter sensor use
DiameterSensor	LREAL	0	Positive number (MinSensorValue to MaxSensorValue)	Yes	Diameter Sensor value
MinSensorValue	LREAL	0	Positive Number (Less than MaxSensorValue)	No	Minimum Diameter sensor value
MaxSensorValue	LREAL	1,000	Greater than min sensor value	No	Maximum Diameter Sensor value

Setting Diameter Filter

This section describes how to set diameter filter.

In real winder/unwinder application, diameter sensor value or internal estimated diameter value may fluctuate. For filtering the instantaneous diameter, define the number of samples in *sWU_CONFIG.VarianceSize* and set *sWU_CONFIG.FilterTime1* and *sWU_CONFIG.FilterTime2* time values for the diameter filter. Refer to the related input variables:

- **sWU_CONFIG**

Variable Name	Data Type	Initial Value	Valid Range	Change over during execution (*Enable*=TRUE)	Comment
VarianceSize	DINT	100	0 to 5,000	No	Variance Size for diameter calculation
FilterTime1	TIME	0.15s	(Less or equal to FilterTime2) From 0 to 600.0s	Yes	Filter Time 1 for diameter calculation
FilterTime2	TIME	0.5s	From FilterTime1 to 600.0 s	Yes	Filter Time 2 for diameter calculation

Setting Rotation Direction

This section describes how to set rotation direction.

Set *sWU_COMMAND*.*RunForward* to TRUE to give run forward command.

Set *sWU_COMMAND*.*RunReverse* to TRUE to give run reverse command.

With both input variables (*RunForward*/*RunReverse*) set to TRUE, the function block gives an error(Error code: 16#8201 000A).

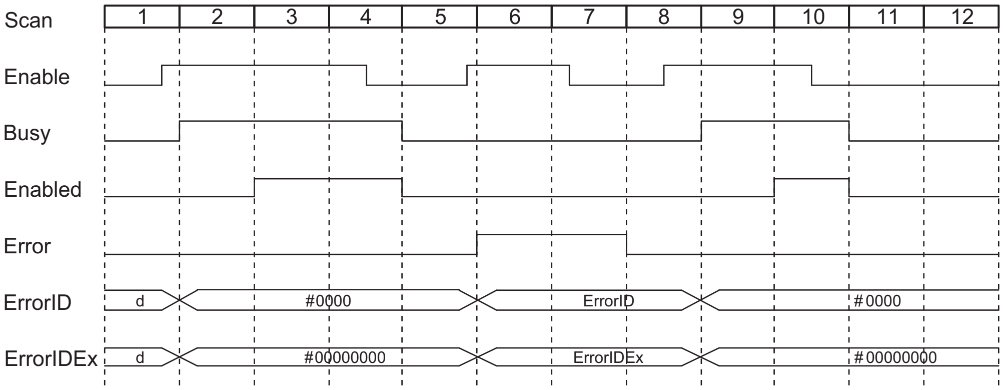
When *sWU_COMMAND.RunForward* and *sWU_RUN_COMMAND.RunReverse* are set to FALSE *WinderSpeedReference* output is set to 0. Set one of the variables to TRUE. Refer to the related input variables:

● sWU_COMMAND

Variable Name	Data Type	Initial Value	Valid Range	Change over during execution (*Enable*=TRUE)	Comment
RunForward	BOOL	FALSE	TRUE/FALSE	Yes	Forward command
RunReverse	BOOL	FALSE	TRUE/FALSE	Yes	Reverse command

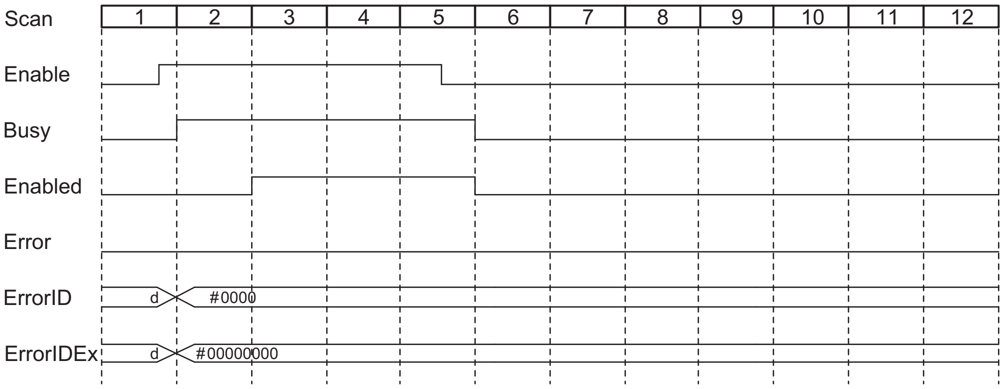
Timing Charts

Normal end, or Error end to Normal end

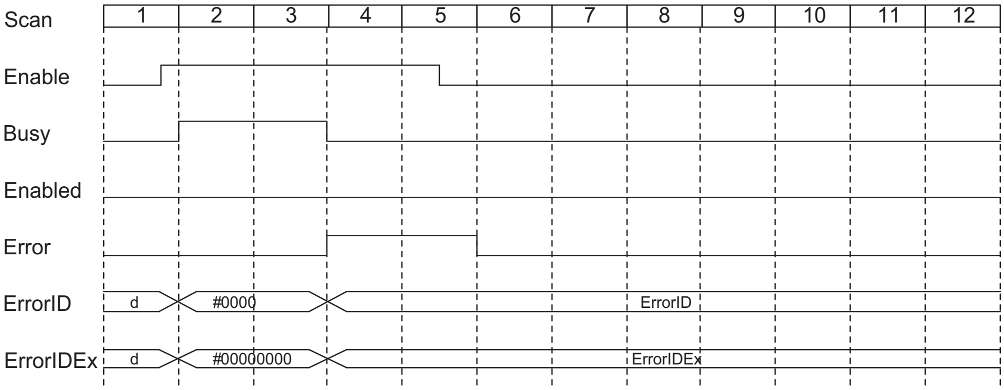


- The processing is executed when the *Enable* input variable is TRUE.
- After completion of the processing series, the *Enabled* output variable changes to TRUE.
- If an error occurs and the *Error* output variable changes to TRUE, the next processing does not continue even if the *Enable* input variable remains TRUE.
- The *Error* output variable retains its value while the *Enable* input variable remains TRUE
- The *ErrorID* and *ErrorIDEx* output variables retain their values even when the processing is executed when the *Enable* input variable is FALSE.
- When *Enable* changes TRUE->FALSE->TRUE after resetting an error, the error code is cleared to zero.

Normal end: The *Enabled* output variable is TRUE for multiple periods



Error completion: The *Error* output variable is TRUE for multiple periods



- At error, *Error* keeps TRUE while *Enable* is TRUE.
- The dotted lines of the Scan shown in the figure indicate multiple scans/task periods.

For information on the error code and cause of error, refer to *Troubleshooting* on page 161.

Troubleshooting

ErrorID

The following table lists the error codes that are stored in *ErrorID*.

Error code	Status	Description	Correction
16#0000	Normal end (Normal)	---	---
16#3C44	Error end (Error)	An error occurred in a standard instruction that is used in the function block.	Check the last four digits of the error code which were sotred in *ErrorIDEx*. Take countermeasures according to instructions in instructions reference manual.
16#3C45	Error end (Error)	An error occurred in the function block.	Check the error code stored in *ErrorIDEx*.

ErrorIDEx

The following table lists the error codes that are stored in *ErrorIDEx* when #3C45 is stored in the *ErrorID* output variable.

Expansion error code	Status	Description	Correction
16#0001 0001	Invalid Input Parameter	Reel diameter value is out of range.	Check Reel Diameter value.
16#0001 0002	Invalid Input Parameter	Middle diameter is lower or equal to reel diameter or middle diameter is bigger or equal to product diameter.	Check middle diameter value.
16#0001 0003	Invalid Input Parameter	Maximum diameter is lower or equal to reel diameter, or maximum diameter is bigger than maximum diameter ratio.	Check Maximum diameter value
16#0001 0004	Invalid Input Parameter	Product diameter is bigger than maximum diameter	Check Product Diameter value.
16#0001 0005	Invalid Input Parameter	Maximum Line Speed is out of range.	Check Maximum Line Speed setting.
16#0001 0006	Invalid Input Parameter	Maximum Motor Speed is out of range.	Check Maximum Motor Speed setting.
16#0001 0007	Invalid Input Parameter	Minimum Diameter Sensor Value is bigger or equal to maximum diameter sensor value when a diameter sensor is used.	Check Minimum Diameter sensor value.
16#0001 0008	Invalid Input Parameter	Variance Size value is out of range.	Check Variance Size.
16#0001 0009	Invalid Input Parameter	Preset Value is bigger than Product Diameter or Maximum Diameter.	Check Diameter Preset Value.
16#0001 000A	Invalid Input Parameter	Both signals running Forward and running Reverse are activated simultaneously.	Check running forward and running reverse signals.
16#0001 000B	Invalid Input Parameter	Minimum diameter sensor value is lower than 0.	Check minimum diameter sensor value.
16#0001 000C	Invalid Input Parameter	Maximum diameter sensor value is lower than 0.	Check maximum diameter sensor value.
16#0001 000D	Invalid Input Parameter	With Diameter Sensor, the actual diameter sensor value is bigger than the maximum diameter sensor value.	Check Diameter sensor value settings.
16#0001 000E	Invalid Input Parameter	PID Task time is out of range	Check PID task time setting.

Expansion error code	Status	Description	Correction
16#0001 0010	Invalid Input Parameter	With Diameter Sensor, the actual diameter sensor value is lower than the minimum diameter sensor value.	Check Diameter sensor value settings.
16#0001 0011	Invalid Input Parameter	Job Completion Time is out of range	Check Job Completion Time value.
16#0001 0013	Invalid Input Parameter	Preset Diameter input function and Reset Diameter input function are activated simultaneously.	Check Preset and Reset Diameter input signals.
16#0001 0014	Invalid Input Parameter	Preset Diameter input function and Reset Diameter input function and Freeze Diameter input function are activated simultaneously.	Check Preset, Reset and Freeze Diameter Input Signals.
16#0001 0016	Invalid Input Parameter	Line Speed value is out of range.	Check Line Speed value.
16#0001 001B	Invalid Input Parameter	Taper Tension at Minimum Diameter is out of range.	Check Taper tension at minimum diameter.
16#0001 001C	Invalid Input Parameter	Taper Tension at Middle Diameter is out of range.	Check Taper tension at middle diameter.
16#0001 001D	Invalid Input Parameter	Taper Tension at Product Diameter is out of range.	Check Taper tension at Product diameter
16#0001 001F	Invalid Input Parameter	Tension Reference is lower than 0% or bigger than 100%	Check Tension Reference setting.
16#0001 0020	Invalid Input Parameter	Tension Feedback is lower than 0% or bigger than 100%	Check Tension Feedback.
16#0001 0021	Invalid Input Parameter	Diameter Filter Time 1 is out of range.	Check Diameter Filter Timer 1.
16#0001 0022	Invalid Input Parameter	Diameter Filter Time 2 is out of range.	Check Diameter Filter Time 2.
16#0001 0023	Invalid Input Parameter	Percentage Diameter completion is out of range.	Check Percentage Diameter Completion setting.
16#0001 0024	Invalid Input Parameter	PID Proportional at Minimum Diameter value is out of range	Check PID Proportional at Minimum Diameter setting.
16#0001 0025	Invalid Input Parameter	PID Proportional at Product Diameter value is out of range.	Check PID Proportional at Product Diameter setting.
16#0001 0026	Invalid Input Parameter	PID Integral at Minimum Diameter value is out of range.	Check PID Integral at Minimum Diameter setting.
16#0001 0027	Invalid Input Parameter	PID Integral at Product Diameter value is out of range.	Check PID Integral at Product Diameter setting.
16#0001 0028	Invalid Input Parameter	PID Derivative value is out of range.	Check PID Derivative setting.
16#0001 0029	Invalid Input Parameter	PID Convergence Gain value is out of range.	Check PID Convergence Gain setting.
16#0001 0030	Invalid Input Parameter	Calculated Inertia Ratio is out of range.	Check Calculated Inertia Ratio.
16#0001 0031	Invalid Input Parameter	Diameter Filter Time 2 is lower than filter time 1 value.	Check Diameter Filter Time settings.
16#0001 0032	Invalid Input Parameter	PID Lower Limit is out of range.	Check PID Lower Limit setting.
16#0001 0033	Invalid Input Parameter	PID Upper Limit is out of range.	Check PID Upper Limit setting.
16#0001 0034	Invalid Input Parameter	PID Output Scale is out of range.	Check PID Output Scale value.
16#0001 0035	Invalid Input Parameter	Minimum Line Speed for web break function is out of range	Check Minimum Line Speed value.
16#0006 0001	Invalid Input Parameter	Web Break Lower Limit value is lower than zero, or Web Break Upper Limit value is lower than zero.	Check Web break Lower Limit or Web break Upper Limit setting.
16#0006 0002	Invalid Input Parameter	Web Break Upper Limit value is bigger than 100.0%, or Web Break Lower Limit value is bigger than 100.0.	Check Web Break Upper Limit or Web Break Lower Limit setting.
16#0006 0003	Invalid Input Parameter	Web Break Upper Limit value is lower or equal to Web Break Lower Limit value	Check Web Break Upper Limit setting.

Expansion error code	Status	Description	Correction
16#0006 0007	Invalid Input Parameter	Web Break Time is out of range.	Check Web Break Detection Time setting.
16#0010 0001	Invalid Parameter	Gear Ratio is out of range.	Check Gear Ratio setting.
16#0010 0002	Invalid Parameter	Motor Inertia is lower or equal to 0.	Check Motor Inertia setting.
16#0010 0003	Invalid Parameter	Product Density and Product Mass are bigger than 0.	Check product density and Product mass settings.
16#0010 0004	Invalid Parameter	Reel Mass value, product density value, product Mass value, product width value or gear inertia value is lower than 0.	Check reel mass, product density, product mass, product width and gear inertia settings for inertia.

Precautions for Correct Use

Multiple error codes listed above may occur at the same time. In this case, the latest error code will be output.

Error Recovery Procedure

● Error Ends When this function block is Started or Being Executed

1. Refer to *Troubleshooting* on page 161 to find and check the cause of the error.
2. Remove the cause of the error.
3. Refer to *Conditions for Execution* on page 141 and check the execution conditions of the function block.
4. Make sure that the execution conditions are met. By changing *Enable* of input variable to TRUE, this function block can be executed.

● When an Error Occurred in the NJ/NX/NY

1. Find and check the cause of the error.
2. Remove the cause of the error.
3. Reset/clear the error occurred in the NJ/NX/NY.
4. After resetting the error, refer to *Conditions for Execution* on page 141 and check the execution conditions of the function block.
5. Make sure that the execution conditions are met. By changing *Enable* of input variable to TRUE, this function block can be executed.

For information on how to reset errors in NJ/NX/NY, refer to the troubleshooting manual.

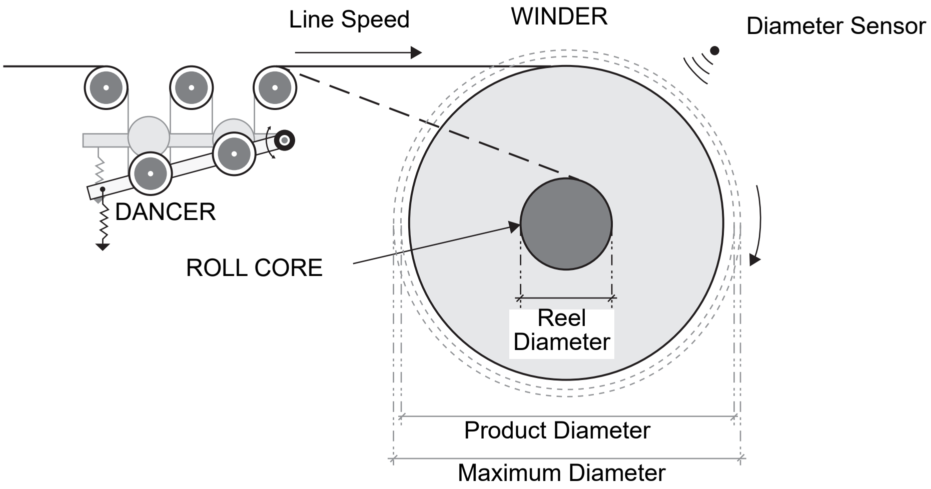
Sample Programming

This section describes WinderVelCtrl_Servo function block in a sample program.

Machine Sample

● Overview

This sample system winds the material with G5 servo for the winder using a diameter sensor and a dancer.



Each dimension is shown in the table below:

Name	Value
Reel Diameter	40mm
Product Diameter	300mm
Maximum Diameter	300mm
Line Speed	75m/min (max. line speed)
Gear Box	1:5
G5 Motor	R88M-K40020F-S2

● Hardware configuration

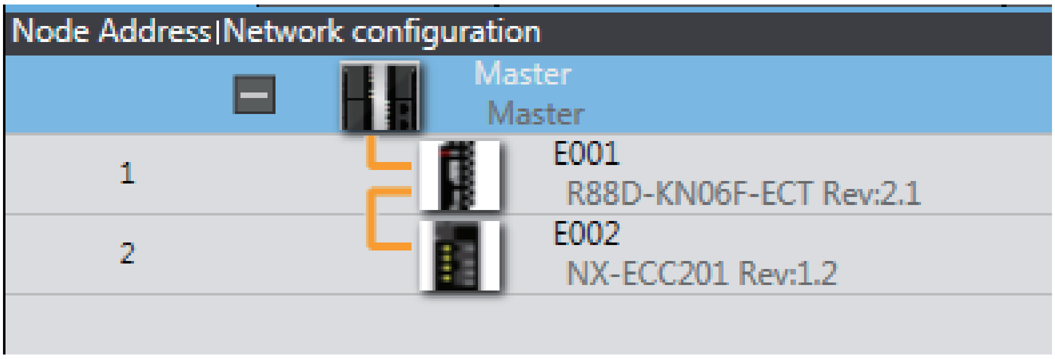
The hardware configuration of the above sample system is as follows.

Name	Type	Version	Number
Machine Automation Controller	NJ501-1300	Ver.1.10	---
G5 Servo Drive	R88D-KN06F-ECT	Ver. 2.1	---
Encoder Line Speed	---	---	---
Diameter sensor	---	---	---

Project Sample

EtherCAT Configuration

The EtherCAT network configuration is as follows.

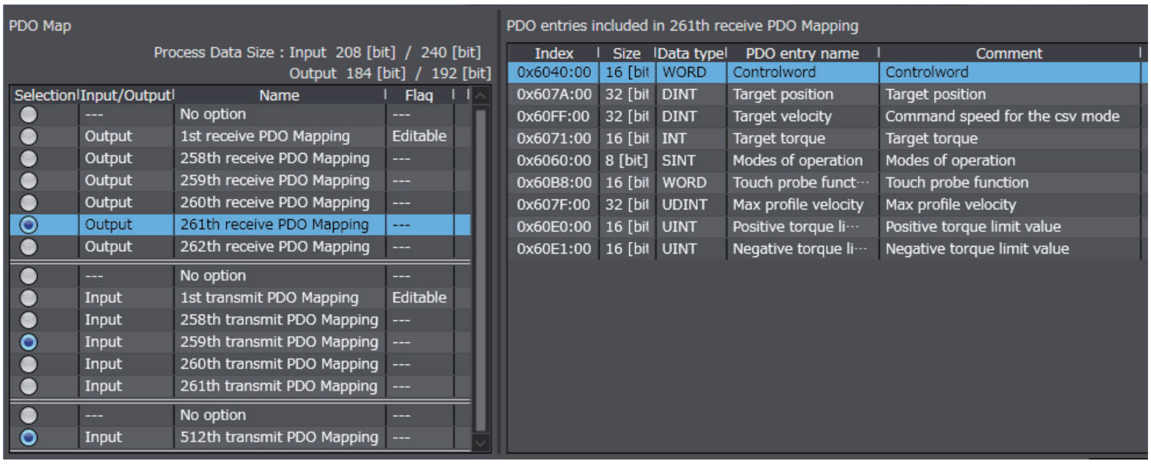


Node Address	Network configuration
	Master Master
1	E001 R88D-KN06F-ECT Rev:2.1
2	E002 NX-ECC201 Rev:1.2

PDO Map

Uses default PDO map setting:

Output



PDO Map

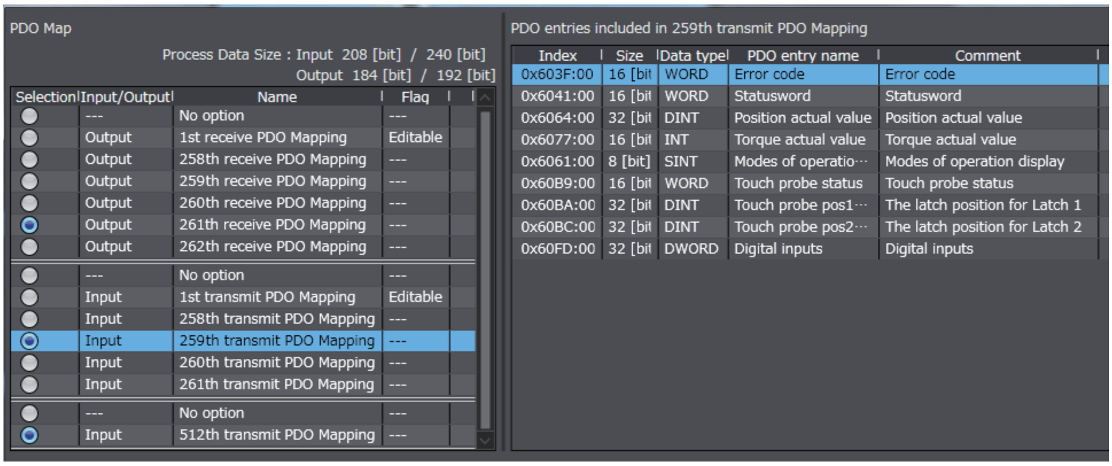
Process Data Size : Input 208 [bit] / 240 [bit]
Output 184 [bit] / 192 [bit]

Selection	Input/Output	Name	Flag
○	---	No option	---
○	Output	1st receive PDO Mapping	Editable
○	Output	258th receive PDO Mapping	---
○	Output	259th receive PDO Mapping	---
○	Output	260th receive PDO Mapping	---
●	Output	261th receive PDO Mapping	---
○	Output	262th receive PDO Mapping	---
○	---	No option	---
○	Input	1st transmit PDO Mapping	Editable
○	Input	258th transmit PDO Mapping	---
●	Input	259th transmit PDO Mapping	---
○	Input	260th transmit PDO Mapping	---
○	Input	261th transmit PDO Mapping	---
○	---	No option	---
●	Input	512th transmit PDO Mapping	---

PDO entries included in 261th receive PDO Mapping

Index	Size	Data type	PDO entry name	Comment
0x6040:00	16 [bit	WORD	Controlword	Controlword
0x607A:00	32 [bit	DINT	Target position	Target position
0x60FF:00	32 [bit	DINT	Target velocity	Command speed for the csv mode
0x6071:00	16 [bit	INT	Target torque	Target torque
0x6060:00	8 [bit]	SINT	Modes of operation	Modes of operation
0x60B8:00	16 [bit	WORD	Touch probe funct···	Touch probe function
0x607F:00	32 [bit	UDINT	Max profile velocity	Max profile velocity
0x60E0:00	16 [bit	UINT	Positive torque li···	Positive torque limit value
0x60E1:00	16 [bit	UINT	Negative torque li···	Negative torque limit value

Input



PDO Map

Process Data Size : Input 208 [bit] / 240 [bit]
Output 184 [bit] / 192 [bit]

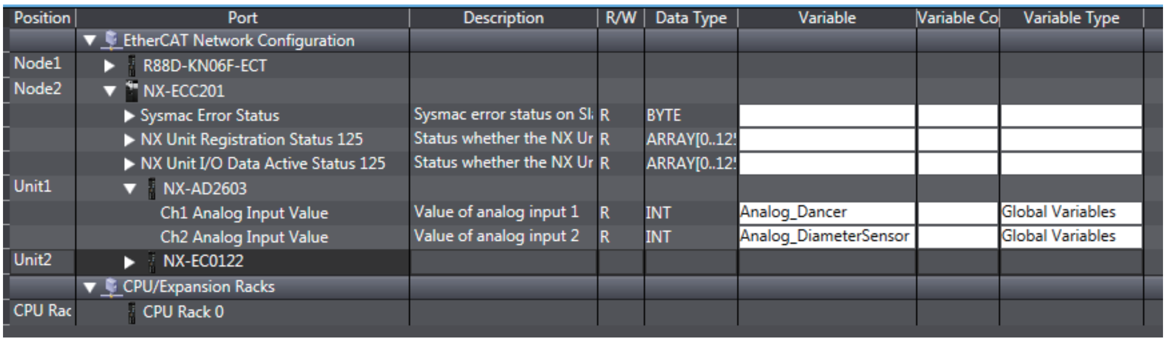
Selection	Input/Output	Name	Flag
○	---	No option	---
○	Output	1st receive PDO Mapping	Editable
○	Output	258th receive PDO Mapping	---
○	Output	259th receive PDO Mapping	---
○	Output	260th receive PDO Mapping	---
●	Output	261th receive PDO Mapping	---
○	Output	262th receive PDO Mapping	---
○	---	No option	---
○	Input	1st transmit PDO Mapping	Editable
○	Input	258th transmit PDO Mapping	---
●	Input	259th transmit PDO Mapping	---
○	Input	260th transmit PDO Mapping	---
○	Input	261th transmit PDO Mapping	---
○	---	No option	---
●	Input	512th transmit PDO Mapping	---

PDO entries included in 259th transmit PDO Mapping

Index	Size	Data type	PDO entry name	Comment
0x603F:00	16 [bit	WORD	Error code	Error code
0x6041:00	16 [bit	WORD	Statusword	Statusword
0x6064:00	32 [bit	DINT	Position actual value	Position actual value
0x6077:00	16 [bit	INT	Torque actual value	Torque actual value
0x6061:00	8 [bit]	SINT	Modes of operatio···	Modes of operation display
0x60B9:00	16 [bit	WORD	Touch probe status	Touch probe status
0x60BA:00	32 [bit	DINT	Touch probe pos1···	The latch position for Latch 1
0x60BC:00	32 [bit	DINT	Touch probe pos2···	The latch position for Latch 2
0x60FD:00	32 [bit	DWORD	Digital inputs	Digital inputs

I/O Map

Set I/O mapping as shown below.



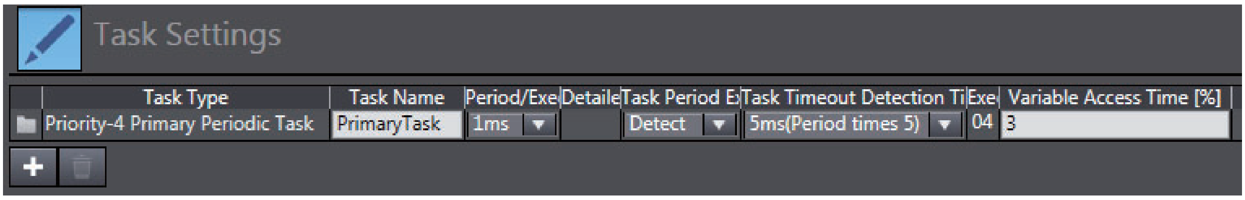
Position	Port	Description	R/W	Data Type	Variable	Variable Co	Variable Type
	▼ EtherCAT Network Configuration						
Node1	▶ R88D-KN06F-ECT						
Node2	▼ NX-ECC201						
	▶ Sysmac Error Status	Sysmac error status on Sl	R	BYTE			
	▶ NX Unit Registration Status 125	Status whether the NX Ur	R	ARRAY[0..12!			
	▶ NX Unit I/O Data Active Status 125	Status whether the NX Ur	R	ARRAY[0..12!			
Unit1	▼ NX-AD2603						
	Ch1 Analog Input Value	Value of analog input 1	R	INT	Analog_Dancer		Global Variables
	Ch2 Analog Input Value	Value of analog input 2	R	INT	Analog_DiameterSensor		Global Variables
Unit2	▶ NX-EC0122						
	▼ CPU/Expansion Racks						
CPU Rac	CPU Rack 0						

Task Settings

The task settings are as follows.

Item	Settings
Task Type	Primary Periodic Task
Task Period	1ms
Assigned Program to the Task	Program0

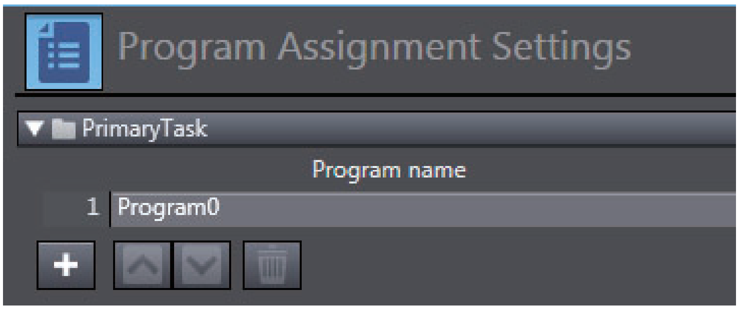
Task Settings



Task Settings

	Task Type	Task Name	Period/Exe	Detaile	Task Period E	Task Timeout Detection Ti	Exe	Variable Access Time [%]
	Priority-4 Primary Periodic Task	PrimaryTask	1ms ▼		Detect ▼	5ms(Period times 5) ▼	04	3

Program assignment Settings



Program Assignment Settings

▼ PrimaryTask

	Program name
1	Program0

Variable List

Internals

Name	Data Type	Initial Value	Comment
Winder	OmronLib\PKG_WU\WinderVelCtrl_Servo		
Error	BOOL		
ErrorID	WORD		
ErrorIDEx	DWORD		
EncoderLineSpeed_Fil	LREAL		
WinderSrv	OmronLib\PKG_WU\sWU_SERVO	Refer to the following description.	
Ready	BOOL		
ServoWU_PWR	MC_Power		
Powered	BOOL		
ServoWU_CSV	MC_SyncMoveVelocity		
ServoWU_STP	MC_Stop		

Externals

Name	Data Type	Constant	Comment
EncoderLine	_sAXIS_REF	TRUE	
ServoWU	_sAXIS_REF	TRUE	
Analog_Dancer	INT	FALSE	
Analog_DiameterSensor	INT	FALSE	
HMI_EnableFB	BOOL	FALSE	
HMI_WinderMode	BOOL	FALSE	
HMI_StartWinder	BOOL	FALSE	
HMI_StopWinder	BOOL	FALSE	
HMI_DecRate	LREAL	FALSE	
FinalWinderSpeedReference	LREAL	FALSE	
EncoderLineSpeed	LREAL	FALSE	
DancerFeedback_Value	LREAL	FALSE	
MinDancer_Value	INT	FALSE	
MaxDancer_Value	INT	FALSE	
ActualDiameterSensorValue	LREAL	FALSE	
PresetInput	BOOL	FALSE	
DiameterPresetValue	LREAL	FALSE	
ResetInput	BOOL	FALSE	
FreezeInput	BOOL	FALSE	
ActualDiameterFil	LREAL	FALSE	
DiameterReached	BOOL	FALSE	
WebBreakDetection	BOOL	FALSE	
TensionSetPoint	LREAL	FALSE	

Global Variables

Name	Data Type	Initial Value	AT	Constant	Comment
Analog_Dancer	INT		“ECAT://node#[2,1]/Ch1 Analog Input Value”	FALSE	
Analog_DiameterSensor	INT		“ECAT://node#[2,1]/Ch2 Analog Input Value”	FALSE	
HMI_EnableFB	BOOL			FALSE	
HMI_WinderMode	BOOL			FALSE	
FinalWinderSpeedReference	LREAL			FALSE	
EncoderLineSpeed	LREAL			FALSE	
DancerFeedback_Value	LREAL			FALSE	
MinDancer_Value	INT			FALSE	
MaxDancer_Value	INT			FALSE	
ActualDiameterSensorValue	LREAL			FALSE	
PresetInput	BOOL			FALSE	
DiameterPresetValue	LREAL			FALSE	
ResetInput	BOOL			FALSE	
FreezeInput	BOOL			FALSE	
ActualDiameterFil	LREAL			FALSE	
DiameterReached	BOOL			FALSE	
WebBreakDetection	BOOL			FALSE	
TensionSetPoint	LREAL			FALSE	
EncoderLine	_sAXIS_REF		MC://_MC_AX[0]	TRUE	
ServoWU	_sAXIS_REF		MC://_MC_AX[1]	TRUE	
HMI_StartWinder	BOOL			FALSE	
HMI_StopWinder	BOOL			FALSE	
HMI_DecRate	LREAL			FALSE	

Setting *WinderSrv.BasicData* (Data Type: sWU_BASIC_DATA)

Name	Data Type	Initial Value
ReelDiameter	LREAL	42
MidDiameter	LREAL	150
MaxDiameter	LREAL	290
ProductDiameter	LREAL	285
MaxLineSpeed	LREAL	50
ActualMotorSpeed	LREAL	0.0
MaxMotorSpeed	LREAL	63.15

Setting *WinderSrv.Config* (Data Type: sWU_CONFIG)

Name	Data Type	Initial Value
SensorMode	BOOL	TRUE
DiameterSensor	LREAL	0.0
MinSensorValue	LREAL	0.0
MaxSensorValue	LREAL	4350
VarianceSize	DINT	500
FilterTime1	TIME	T#5S
FilterTime2	TIME	T#10S
PercentageJobCompletion	LREAL	98
TimeJobCompletion	TIME	T#0.5S

Setting *WinderSrv.Command* (Data Type: sWU_COMMAND)

Name	Data Type	Initial Value
RunForward	BOOL	FALSE
RunReverse	BOOL	FALSE

Setting *WinderSrv.PID* (Data Type: sWU_PID)

Name	Data Type	Initial Value
ProportionalMinDiameter	LREAL	0.02
ProportionalProductDiameter	LREAL	0.02
IntegralMinDiameter	LREAL	0.01
IntegralProductDiameter	LREAL	0.01
Derivative	LREAL	0
ConvergenceGain	LREAL	3
TaskTime	LREAL	0.001
PIDOutputScale	LREAL	1
PIDUpperLimit	LREAL	100
PIDLowerLimit	LREAL	-100

Setting *WinderSrv.InertiaData* (Data Type: sWU_INERTIA_DATA)

Name	Data Type	Initial Value
Enable	LREAL	0.0
ReelMass	LREAL	0.0
ProductDensity	LREAL	0.0
ProductMass	LREAL	0.0
ProductWidth	LREAL	0.0
GearRatio	LREAL	0.0
MotorInertia	BOOL	FALSE
GearInertia	LREAL	0.0

Setting *WinderSrv.Taper* (Data Type: sWU_TAPER)

Name	Data Type	Initial Value
TensionMinDiameter	LREAL	100
TensionMidDiameter	LREAL	100
TensionProductDiameter	LREAL	100

Setting *WinderSrv.WebBreak* (Data Type: sWU_WEBBREAK)

Name	Data Type	Initial Value
Enable	BOOL	FALSE
MinLineSpeed	LREAL	10
UpperLimit	LREAL	80
LowerLimit	LREAL	10
WebBreakTime	TIME	T#2S

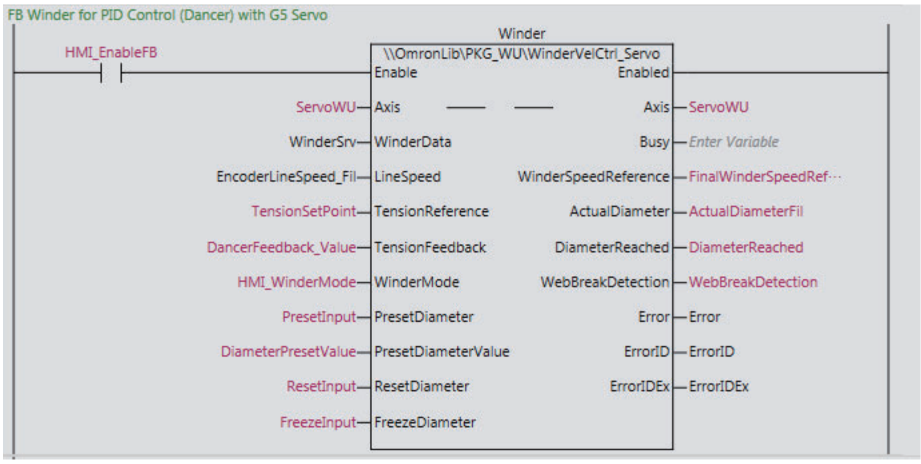
Ladder Diagram

A sample program with WinderVelCtrl_Servo.

1

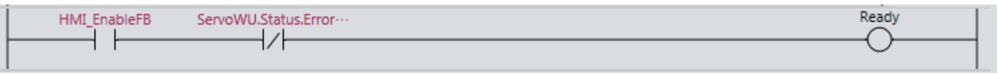
```
// WinderMode Setting
HMI_WinderMode := FALSE;

// Encoder Line Speed conversion from mm/sec to meters/minute
EncoderLineSpeed := (ABS(EncoderLine.Act.Vel) * 60.0 ) / 1000.0;

//Filter for Encoder Line Speed.
EncoderLineSpeed_Fil := EncoderLineSpeed_Fil + 0.001 * ( EncoderLine-
Speed - EncoderLineSpeed_Fil);

//Dancer Feedback value. Scaled from 0 to 100%.
DancerFeedback_Value := ((INT_TO_LREAL ( Analog_Dancer - Min-
Dancer_Value)*100.0) / INT_TO_LREAL (MaxDancer_Value - Min-
Dancer_Value));

// Diameter Sensor Value
ActualDiameterSensorValue := INT_TO_LREAL (Analog_DiameterSensor);
```

2 Execution of Winder FB with G5 servo drive

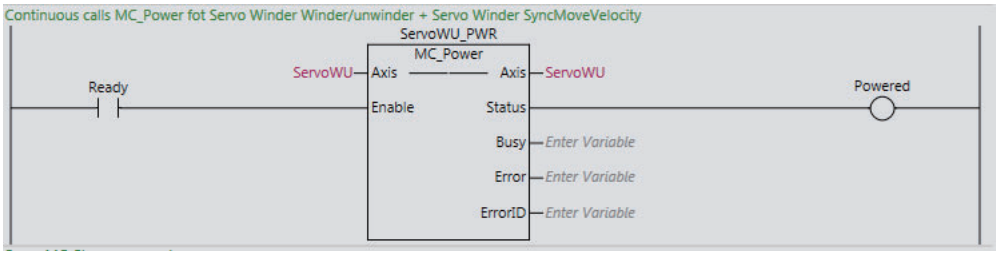


WinderVelCtrl_Servo execution

3 Supervision status

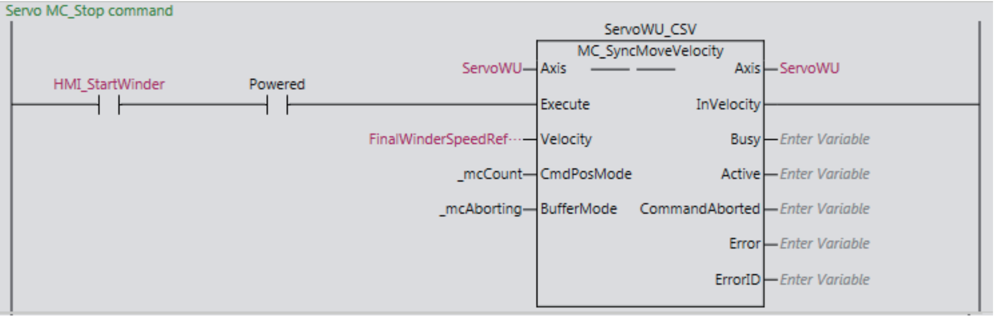


Continuous calls MC_Power fot Servo Winder



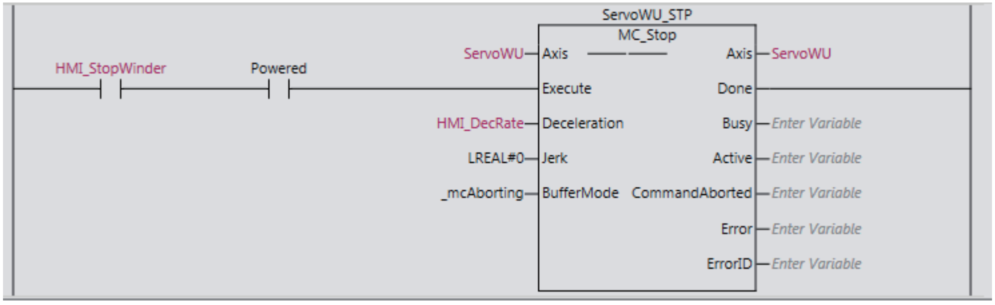
MC_Power execution

4 MC_SyncMoveVelocity fot Servo Winder



MC_SyncMoveVelocity execution

5 Servo MC_Stop command



MC_Stop execution

Operation procedure

This section describes operation procedure to execute WinderVelCtrl_Servo.

● From Power ON to Running

1.	Power on the system	Turn the system main power on
	▼	
2.	ResetECError	In case of E-CAT error, reset the error.
	▼	
3.	ResetMCError	In case of Motion error, reset the error.
	▼	
4.	G5 Servo Drive Reset	In case of an error in G5 Servo Drive, reset the error.
	▼	
5.	Configure winder application	Configure winder application. Set setting items.
	▼	
6.	Servo MC_Power	Servo MC_Power.
	▼	
7.	Stop Line Speed	Confirm the line speed is stopped.
	▼	
8.	Set *Enable* to TRUE	Set *Enable* to TRUE.
	▼	
9.	Servo SyncMoveVelocity	SyncMoveVelocity with calculated speed by WinderVelCtrl_Servo.
	▼	
10.	Start Line Speed	Start Line Speed

● From Running (Winding/Unwinding) to Error stop

1.	An Error occurs	When error(s) appear(s) in WinderVelCtrl_Servo, the winding/unwinding operation must be stopped (Error Stop).
	▼	
2.	Reset the error	Check the cause of error(s). After checking and confirming safety, reset the error. For troubleshooting, refer to the user's manuals.

● From Running (Winding/Unwinding) to Emergency stop

1.	Emergency stop occurs	When emergency(s) stop or fatal error(s) appears in the machine, the winding operation stops immediately (Emergency Stop).
	▼	
2.	Reset the error.	Check cause of the emergency. After checking and confirming safety, reset the error. For troubleshooting, refer to the user's manuals.

● From Emergency/Error stop to Restart

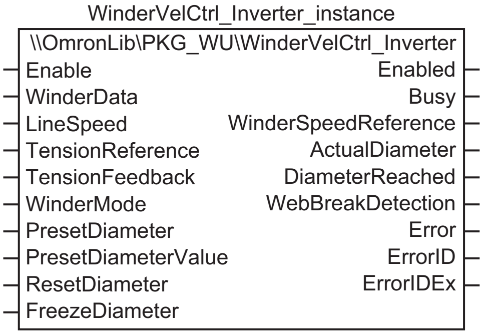
1.	Resetting Error(s)	Reset the error(s).
	▼	
2.	Line Speed stopped	Be sure that line speed is stopped before enabling the function block.
	▼	
3.	Set *Enable* TRUE	Set *Enable* to TRUE.

WinderVelCtrl_Inverter

WinderVelCtrl_Inverter is a function block that controls the tension of a continuous material being wound or unwound with inverter.

This function block can be used for the following applications.

- Wound or unwound material from a drum or roll.
- Feeding wrapping material (unwinder) in packaging application.
- Other sheet transfer applications.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
WinderVel Ctrl_Inverter	Velocity Control Winder (for Inverter)	FB	 WinderVelCtrl_Inverter_instance \\OmronLib\PKG_WU\WinderVelCtrl_Inverter Enable WinderData LineSpeed TensionReference TensionFeedback WinderMode PresetDiameter PresetDiameterValue ResetDiameter FreezeDiameter Enabled Busy WinderSpeedReference ActualDiameter DiameterReached WebBreakDetection Error ErrorID ErrorIDEx	WinderVelCtrl_Inverter_instance (Enable, WinderData, LineSpeed, TensionReference, TensionFeedback, WinderMode, PresetDiameter, PresetDiameterValue, ResetDiameter, FreezeDiameter, Enabled, Busy, WinderSpeedReference, ActualDiameter, DiameterReached, WebBreakDetection, Error, ErrorID, ErrorIDEx);

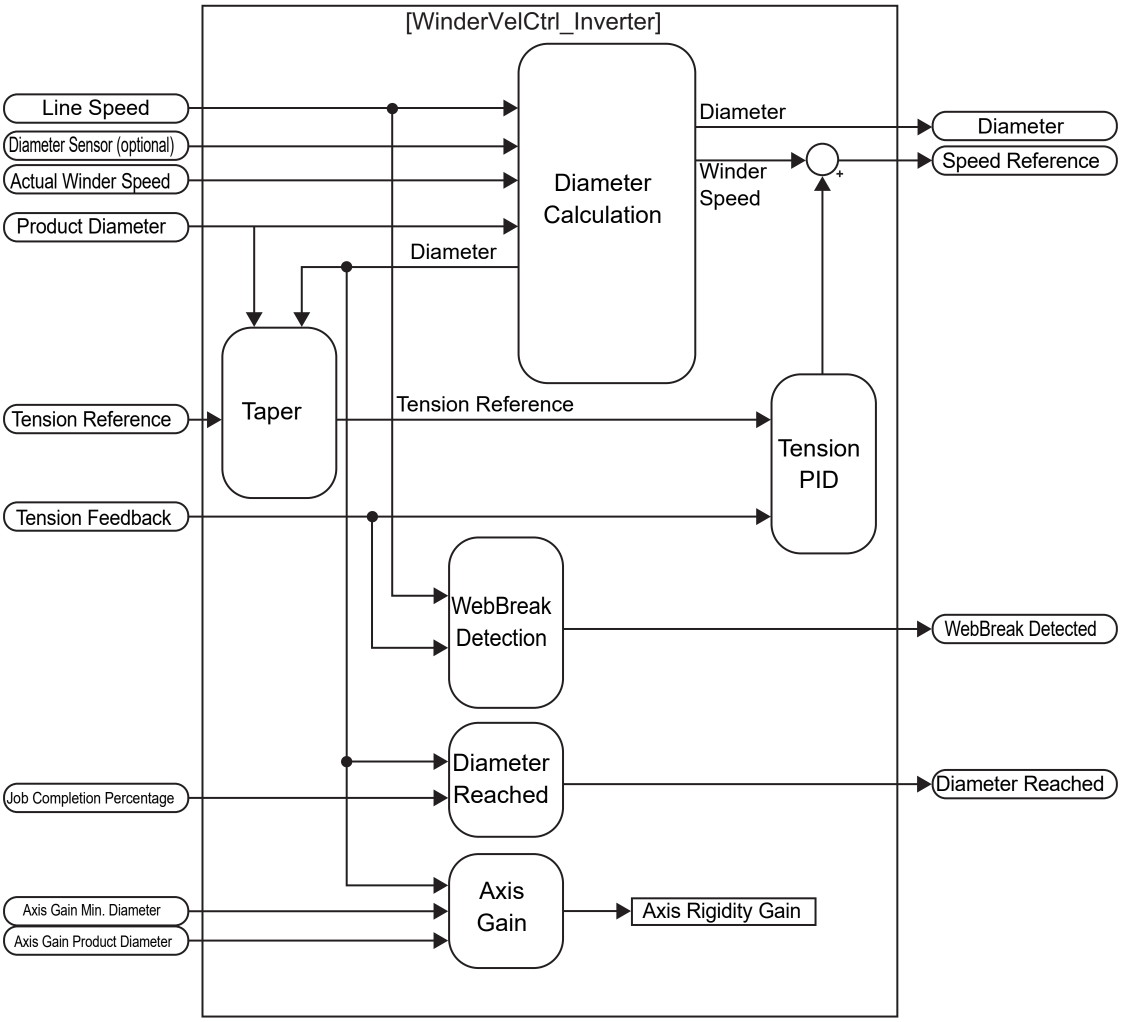
Function Block and Function Information

Item	Description
Library file name	OmronLib_PKG_WU_Vx_x.sir*1
Namespace	OmronLib\PKG_WU
Function block and function number	00066
Publish/Do not publish source code	Not Published.
Function block and function version	1.00

*1. Vx_x indicates version.

Overview of Processing

This section describes the internal processing of this function block.



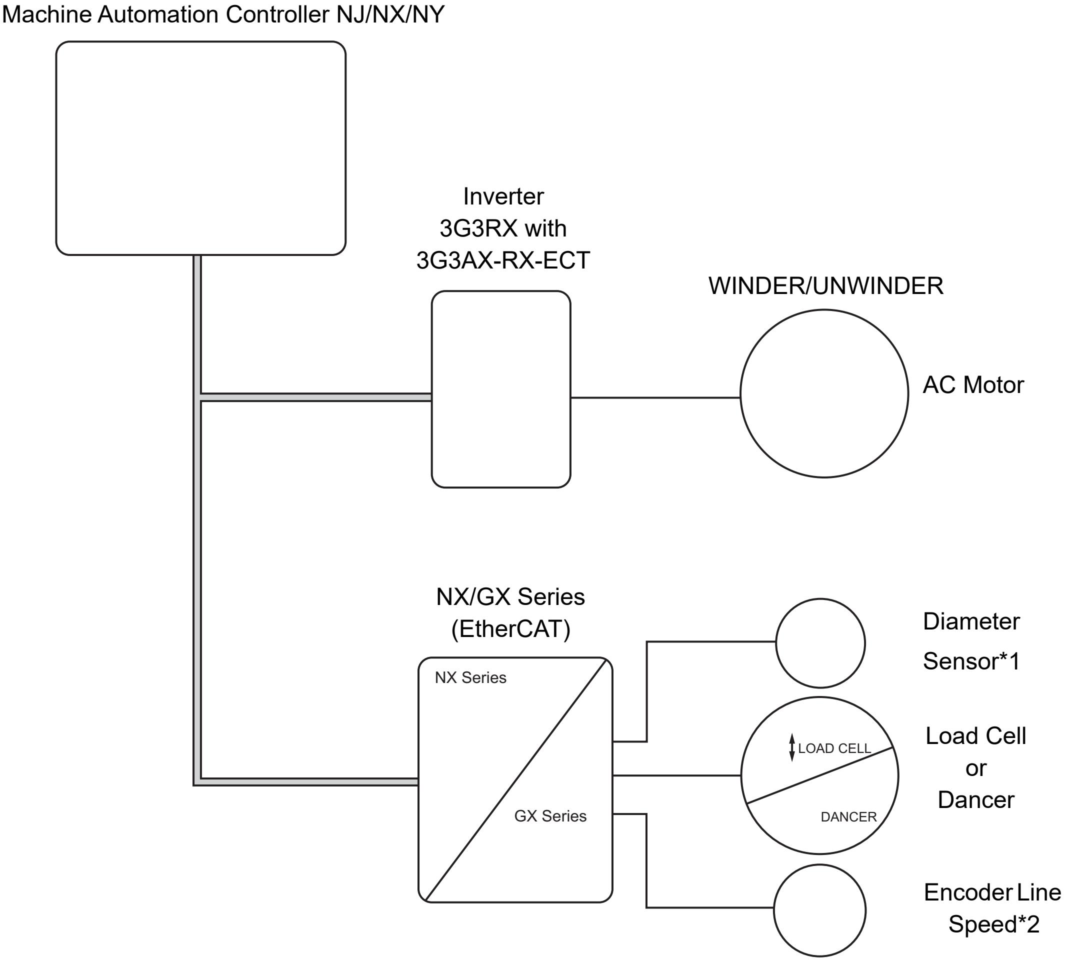
Hardware and Software Configuration

This function block covers the following system configurations.

By 3G3RX Frequency Inverter

● Hardware Configuration

Winder/Unwinder is driven by 3G3RX frequency inverter using NX-Series/GX Series I/Os for the diameter sensor, line speed (encoder) and the load cell/dancer.



- The use of a load cell or dancer is necessary for winder speed tension control.

Note 1. Diameter sensor is optional in order to estimate the diameter with the function block.

2. Line Speed could be provided by analog input or other device trough EtherCAT communications.

● Drive and Software Configuration

Configure the axis as shown below.

Line Speed Axis Configuration

Classification	Parameter name	Set value
Axis Basic Settings	Axis Type[*1]	1: Encoder axis
Position Count Settings	Count Mode	1: Rotary Mode (Infinite length)
	Modulo Maximum Position Setting Value	Negative value, positive value or 0 for LREAL data type[*2]
	Modulo Minimum Position Setting Value	Negative value, positive value or 0 for LREAL data type[*2]

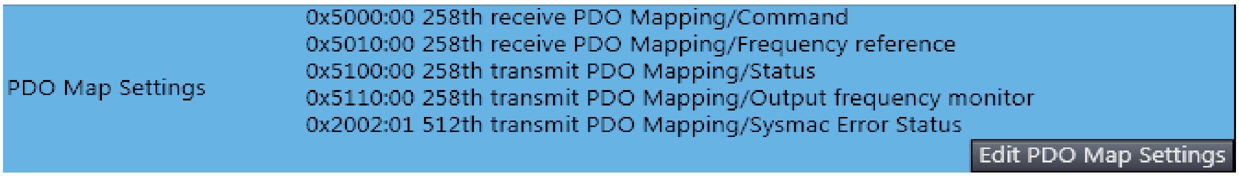
[*1] When using an encoder for line speed.

[*2] Set the value according to the device where this function block is used.

Configure the inverter as shown below.

Sysmac Studio Configuration

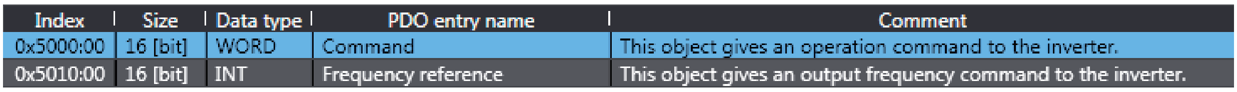
PDO Map Settings example



PDO Map Settings

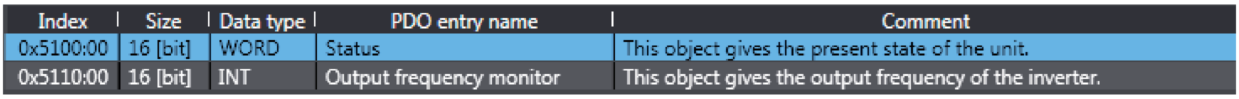
0x5000:00 258th receive PDO Mapping/Command
0x5010:00 258th receive PDO Mapping/Frequency reference
0x5100:00 258th transmit PDO Mapping/Status
0x5110:00 258th transmit PDO Mapping/Output frequency monitor
0x2002:01 512th transmit PDO Mapping/Sysmac Error Status

Output

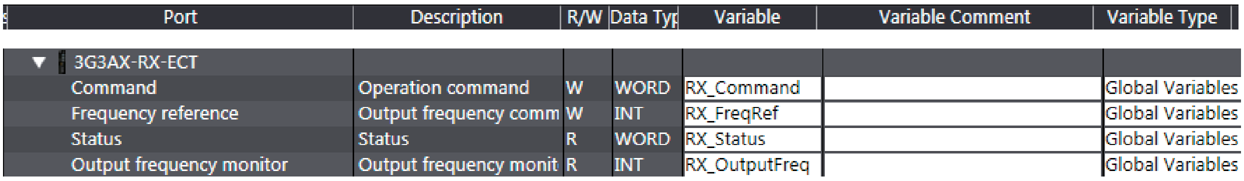


Index	Size	Data type	PDO entry name	Comment
0x5000:00	16 [bit]	WORD	Command	This object gives an operation command to the inverter.
0x5010:00	16 [bit]	INT	Frequency reference	This object gives an output frequency command to the inverter.

Input



Index	Size	Data type	PDO entry name	Comment
0x5100:00	16 [bit]	WORD	Status	This object gives the present state of the unit.
0x5110:00	16 [bit]	INT	Output frequency monitor	This object gives the output frequency of the inverter.



Port	Description	R/W	Data Typ	Variable	Variable Comment	Variable Type
▼ 3G3AX-RX-ECT						
Command	Operation command	W	WORD	RX_Command		Global Variables
Frequency reference	Output frequency comm	W	INT	RX_FreqRef		Global Variables
Status	Status	R	WORD	RX_Status		Global Variables
Output frequency monitor	Output frequency monit	R	INT	RX_OutputFreq		Global Variables

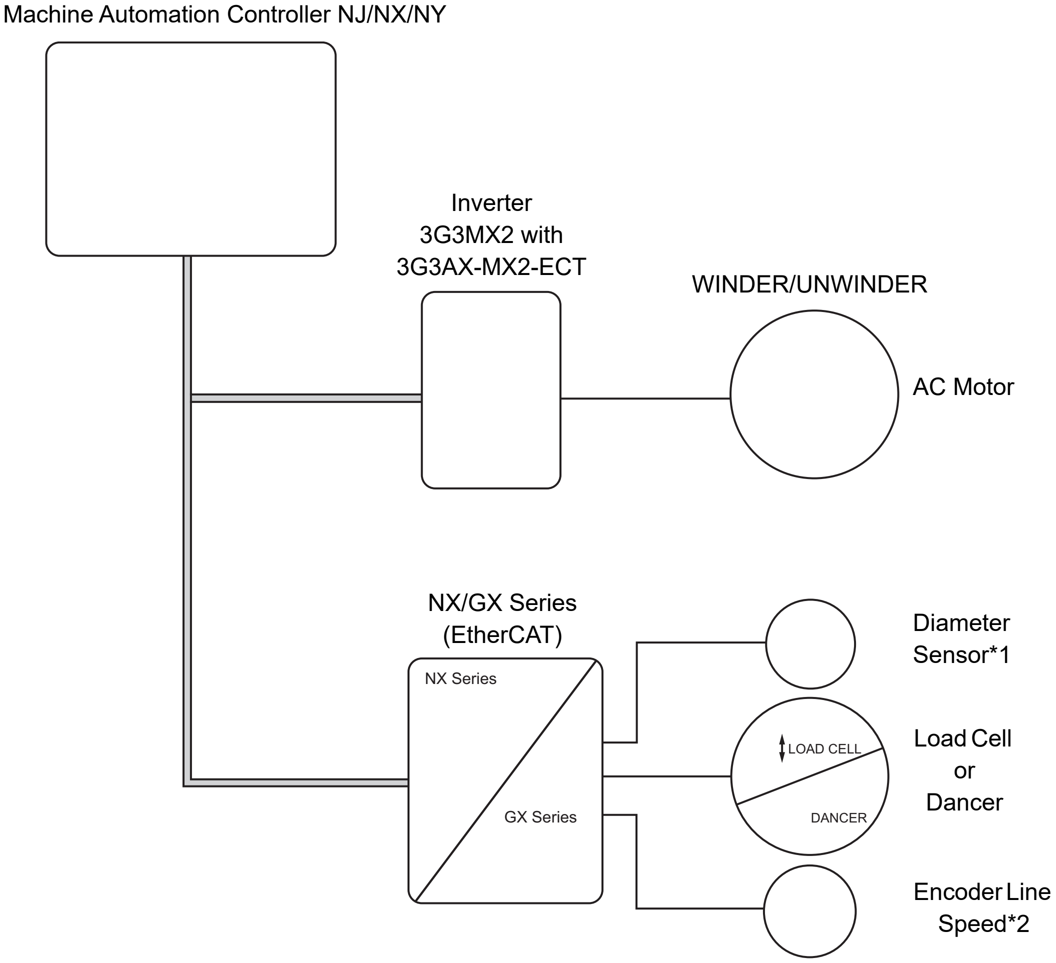
Installation and wiring power supply

This manual does not cover how to install the inverters in cabinets, how to wire power supply, and how to satisfy other application-specific requirements. For these, refer to *Related Manuals* on page 16.

By 3G3MX2 Frequency Inverter

● Hardware Configuration

Winder/Unwinder controls 3G3MX2 series inverter by using NX/GX series I/O unit, diameter sensor, line speed (encoder), and Load Cell/Dancer.



- Load cell or dancer is necessary for winder tension control.

*1. Diameter sensor is optional to estimate the diameter with the function block.

*2. Line Speed can be provided by analog input or other device trough EtherCAT communications.

● Drive Configuration

Configure the axis as shown below.

Line Speed Axis Configuration

Classification	Parameter name	Set value
Axis Basic Settings	Axis Type[*1]	Encoder axis
Position Count Settings	Count Mode	1: Rotary Mode
	Modulo Maximum Position Setting Value	Positive value [*2]
	Modulo Minimum Position Setting Value	Negative value [*2]

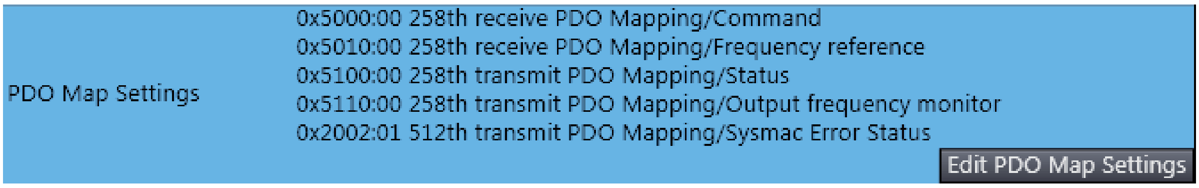
[*1] When using an encoder for line speed.

[*2] Set the value according to the device where this function block is used.

Configure the inverter as shown below

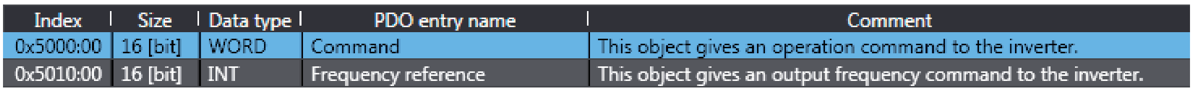
Sysmac Studio Configuration

- PDO Map Settings example



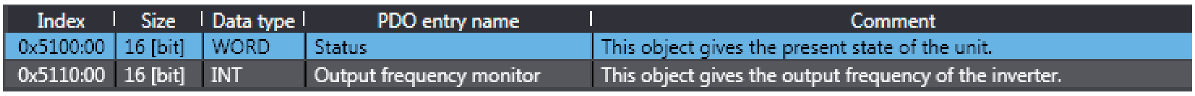
PDO Map Settings	0x5000:00 258th receive PDO Mapping/Command 0x5010:00 258th receive PDO Mapping/Frequency reference 0x5100:00 258th transmit PDO Mapping/Status 0x5110:00 258th transmit PDO Mapping/Output frequency monitor 0x2002:01 512th transmit PDO Mapping/Sysmac Error Status

Output

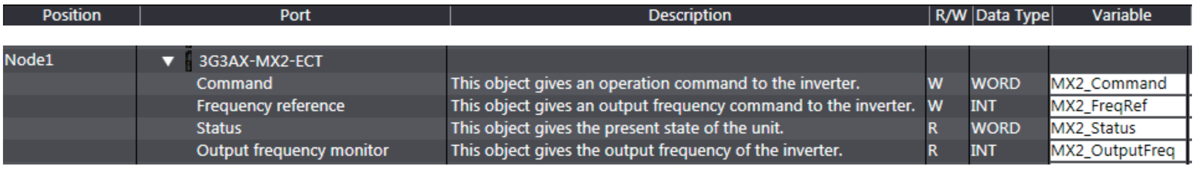


Index	Size	Data type	PDO entry name	Comment
0x5000:00	16 [bit]	WORD	Command	This object gives an operation command to the inverter.
0x5010:00	16 [bit]	INT	Frequency reference	This object gives an output frequency command to the inverter.

Input



Index	Size	Data type	PDO entry name	Comment
0x5100:00	16 [bit]	WORD	Status	This object gives the present state of the unit.
0x5110:00	16 [bit]	INT	Output frequency monitor	This object gives the output frequency of the inverter.



Position	Port	Description	R/W	Data Type	Variable
Node1	▼ 3G3AX-MX2-ECT				
	Command	This object gives an operation command to the inverter.	W	WORD	MX2_Command
	Frequency reference	This object gives an output frequency command to the inverter.	W	INT	MX2_FreqRef
	Status	This object gives the present state of the unit.	R	WORD	MX2_Status
	Output frequency monitor	This object gives the output frequency of the inverter.	R	INT	MX2_OutputFreq

Installation and Power Circuits

This manual does not cover how to install the inverters in cabinets, how to wire power supply, and how to satisfy specific requirements for other application. For these, refer to *Related Manuals* on page 16.

Usage Conditions

Conditions for Execution

This section describes the execution conditions of this function block.

Execution Conditions of the function block

Following condition must be met:

- While *Enable* is TRUE, the Inverter must be turned ON.
- There must be no error in the Inverter.
- When *Enable* changes from FALSE to TRUE, the winder/unwinder must be stopped and line speed must be 0.

These conditions are not detected as errors and must be considered as safety measures in external control circuits.

Selecting Inverter Model and Node Number

Select the inverter model and set the node number according to EtherCAT configuration.

Condition of Task Settings

This function block should be executed in the primary periodic task.

Precautions for Correct Use

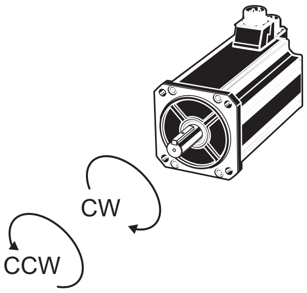
If this function block is not executed in the primary periodic task, specify the task time in *Task-Time* setting for the PID.

Other Configurations

Setting Rotation Direction of the Motor

When using this function block, confirm the rotation direction setting of the frequency inverter drive according to the rotation direction of the winder/unwinder.

Verify that motor rotation direction with Fw signal is clockwise (CW) in winder mode. The motor rotation direction when viewing the shaft from the load side is called clockwise (CW) or counterclockwise (CCW).



Variables

Input Variables

Name	Data type	Default	Valid range	Change over during execution	Description
Enable	BOOL	FALSE	TRUE, FALSE	Yes	When *Enable* changes from FALSE to TRUE, this function block is executed. When *Enable* changes from TRUE to FALSE, this function block stops.
WinderData	sWU_IN-VERTER	Refer to *sWU_SERVO* on page 143.	---	---	Winder configuration.
LineSpeed	LREAL	0m/min	0~2,000m/min	Yes	Actual Line Speed
TensionReference	LREAL	0%	0~100%	Yes	Tension Reference
TensionFeedback	LREAL	0%	0~100%	Yes	Tension Feedback
WinderMode	BOOL	FALSE	TRUE, FALSE	No	Winder Mode FALSE: Winder, TRUE: Unwinder
PresetDiameter	BOOL	FALSE	TRUE, FALSE	Yes	Preset Input Function
PresetDiameter-Value	LREAL	40mm	MinDiameter~ ProductDiameter	Yes	Preset Diameter Value
ResetDiameter	BOOL	FALSE	TRUE, FALSE	Yes	Reset Input Function
FreezeDiameter	BOOL	FALSE	TRUE, FALSE	Yes	Freeze Input Function

Output Variables

Name	Data type	Description
Enabled	BOOL	TRUE when output is enable.
Busy	BOOL	TRUE during execution. When *Error* changes to TRUE, *Busy* becomes FALSE. *Busy* is FALSE during non-execution status.
WinderSpeedReference	LREAL	Winder/Unwinder Speed Reference in Hz
ActualDiameter	LREAL	Current Winder/Unwinder diameter in mm
DiameterReached	BOOL	TRUE after the current winder diameter value reaches *PercentageJobCompletion* setting value and *TimeJobCompletion* setting time passes.
WebBreakDetection	BOOL	TRUE if a web break has been detected.
Error	BOOL	TRUE if an error occurs.
ErrorID	WORD	Outputs the error code when an error occurs. #0000 is the initial value and indicated as normal end.
ErrorIDEx	DWORD	Outputs the error code when an error occurs. #00000000 is the initial value and indicates a normal end.

Structures

● sWU_INVERTER

Name	Data type	Default	Valid range	Change over during execution	Description
BasicData	sWU_BASIC_DATA	Refer to *sWU_BASIC_DATA* on page 144.	---	---	---
Config	sWU_CONFIG	Refer to *sWU_CONFIG* on page 144.	---	---	---
Status	sWU_RUN_STATUS	Refer to *sWU_RUN_STATUS* on page 182.	---	---	---
PID	sWU_PID	Refer to *sWU_PID* on page 145.	---	---	---
AxisGain	sWU_RIGIDITY_GAIN	Refer to *sWU_RIGIDITY_GAIN* on page 183.	---	---	---
Taper	sWU_TAPER	Refer to *sWU_TAPER* on page 145.	---	---	---
WebBreak	sWU_WEBBREAK	Refer to *sWU_WEBBREAK* on page 146.	---	---	---

● sWU_BASIC_DATA

Name	Data type	Default	Valid range	Change over during execution	Description
ReelDiameter	LREAL	40.0mm	1.0 to 2,000.0mm	No	Reel Diameter in mm
MidDiameter	LREAL	100mm	Greater than Reel Diameter and lower than Product Diameter	Yes	Middle diameter in mm for Taper function
MaxDiameter	LREAL	300mm	Greater or equal to Product diameter. (Maximum value = *ReelDiameter*MaxRatio(20)*)	No	Maximum Diameter in mm
ProductDiameter	LREAL	300mm	MidDiameter < *ProductDiameter* ≤ MaxDiameter	Yes	Product Diameter in mm
MaxLineSpeed	LREAL	25m/min	0 m/min < MaxLineSpeed ≤ 2,000m/min	No	Maximum Line Speed in m/min
ActualMotorSpeed	LREAL	0.0 Hz	Positive Number	Yes	Actual Motor speed in Hz
MaxMotorSpeed	LREAL	50 Hz	Positive Number (bigger than 0)	No	Maximum Motor speed in Hz.

● sWU_CONFIG

Name	Data type	Default	Valid range	Change over during execution	Description
SensorMode	BOOL	FALSE	TRUE/FALSE	No	Diameter sensor use
DiameterSensor	LREAL	0	Positive number (MinSensorValue to MaxSensorValue)	Yes	Diameter Sensor value
MinSensorValue	LREAL	0	Positive Number (Less than MaxSensorValue)	No	Minimum Diameter sensor value
MaxSensorValue	LREAL	1,000	Greater than MinSensorValue	No	Maximum Diameter Sensor value
VarianceSize	DINT	100	0 to 5,000	No	Variance Size for diameter calculation
FilterTime1	TIME	0.150	(Less or equal to FilterTime2) From 0 to 600.0s	Yes	Filter Time 1 for diameter calculation
FilterTime2	TIME	0.5	From FilterTime1 to 600.0 s	Yes	Filter Time 2 for diameter calculation
PercentageJob Completion	LREAL	95.0%	0 to 100%	Yes	Job Completion percentage
TimeJob Completion	TIME	0.5s	0 to 600.0s	Yes	Job Completion Time

● sWU_RUN_STATUS

Name	Data type	Default	Valid range	Change over during execution	Description
RunningForward	BOOL	FALSE	TRUE/FALSE	Yes	Run Forward status
RunningReverse	BOOL	FALSE	TRUE/FALSE	Yes	Run Reverse status

sWU_PID

Name	Data type	Default	Valid range	Change over during execution	Description
ProportionalMin Diameter	LREAL	1.0	0.01 to 1,000.0	Yes	PID Proportional Value at minimum diameter
Proportional ProductDiameter	LREAL	1.0	0.01 to 1,000.0	Yes	PID Proportional Value at Product Diameter
IntegralMin Diameter	LREAL	1.0	0.0 to 10,000	Yes	PID Integral Value at minimum Diameter
IntegralProduct Diameter	LREAL	1.0	0.0 to 10,000	Yes	PID Integral Value at Product Diameter
Derivative	LREAL	0.0	0.0 to 10,000.0	Yes	PID Derivative value
ConvergenceGain	LREAL	1.0	0.1 to 5.0	Yes	To give smooth operation near the setpoint.
TaskTime	LREAL	0.001	0.0005 to 0.1	No	Task Time in seconds.
PIDOutputScale	LREAL	1.0	0.0 to 1.0	Yes	PID Output scale factor
PIDUpperLimit	LREAL	100.0	0 to 100.0%	Yes	PID upper limit in %
PIDLowerLimit	LREAL	-100.0	-100.0 to 0%	Yes	PID lower limit in %

sWU_RIGIDITY_GAIN

Name	Data type	Default	Valid range	Change over during execution	Description
Enable	BOOL	FALSE	TRUE/FALSE	Yes	Enable/Disable inverter Rigidity Gain Function
InverterModel	BOOL	FALSE	TRUE/FALSE (TRUE = RX, FALSE = MX2)	No	It indicates the inverter that is being used in the function block.
GainMin Diameter	LREAL	2	RX: 0.001 to 80.000 MX2: 1 to 1,000	Yes	Inverter Rigidity Gain at minimum diameter.[*1]
GainProduct Diameter	LREAL	2	RX: 0.001 to 80.000 MX2: 1 to 1,000	Yes	Inverter Rigidity Gain at product diameter.[*1]
Node	UINT	1	1 to 192	No	Node to write Rigidity Gain .

[*1] The H005 parameter range is different depending on inverter model selection.

sWU_TAPER

Name	Data type	Default	Valid range	Change over during execution	Description
TensionMin Diameter	LREAL	100.0%	0.0% to 200.0%	Yes	Tension at minimum diameter
TensionMid Diameter	LREAL	100.0%	0.0% to 200.0%	Yes	Tension at middle diameter
TensionProduct Diameter	LREAL	100.0%	0.0% to 200.0%	Yes	Tension at Product diameter

● sWU_WEBBREAK

Name	Data type	Default	Valid range	Change over during execution	Description
Enable	BOOL	FALSE	TRUE/FALSE	Yes	Enable web break detection
MinLineSpeed	LREAL	10.0m/min	0 to MaxLineSpeed	Yes	Minimum Line Speed for web break detection
UpperLimit	LREAL	80.0%	Greater than Lower Limit (0% to 100%)	Yes	Upper Limit level for web break detection
LowerLimit	LREAL	10%	Lower than Upper-Limit (0% to 100%)	Yes	Lower Limit for web break detection
WebBreakTime	TIME	1.0s	0.1 s to 10.0 s	Yes	Web break detection time

Function

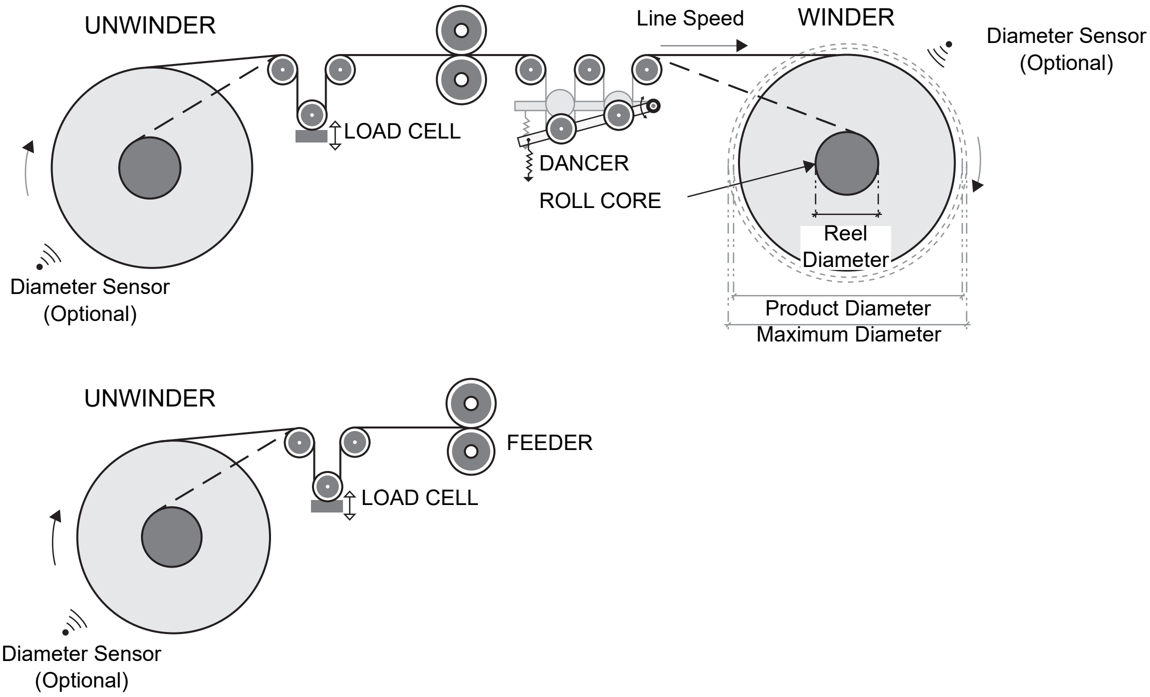
Application Configuration

This section describes an application example of this function block.

The figure below shows an example of a process with winders.

To control the web tension, a load cell or dancer is needed.

The diameter sensor is optional in order to estimate the roll diameter with the function block.



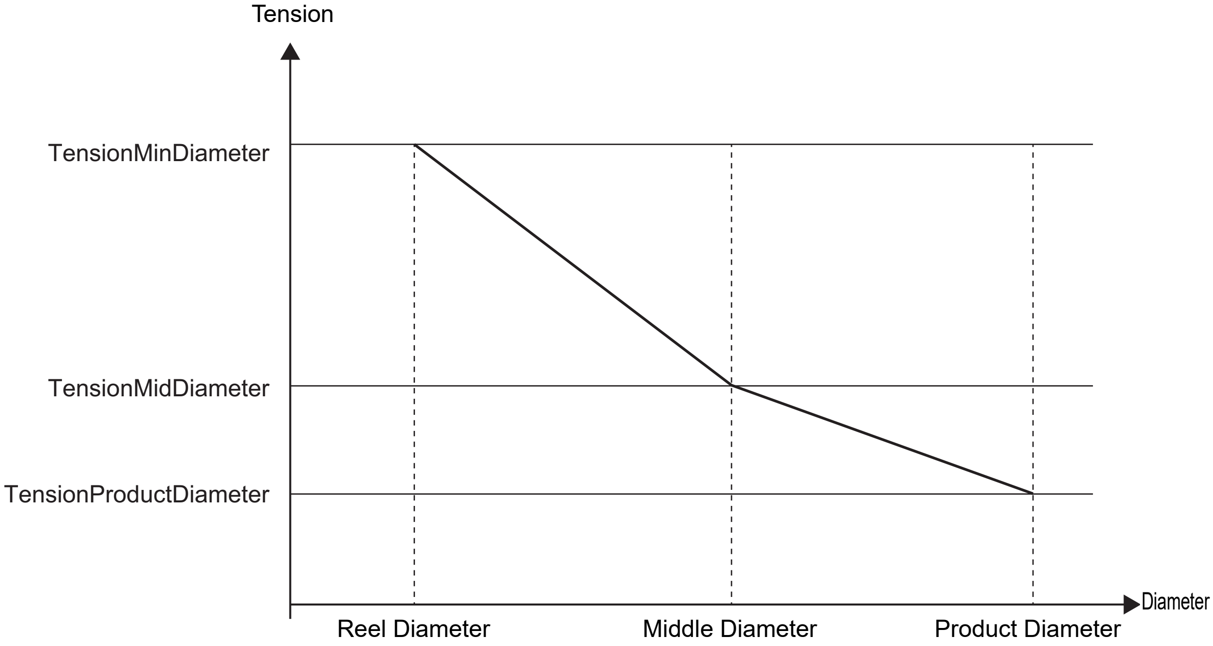
Functions and features of WinderVelCtrl_Inverter

● Taper

This section describes the Taper function.

It provides for compensation of the tension reference relating to diameter and helps to avoid defects like telescoping and crushed rolls.

Set the desired tension reference for the three defined diameters, where *sWU_TAPER.TensionMinDiameter* is the taper tension reference for reel diameter (*sWU_BASIC_DATA.ReelDiameter*), *sWU_TAPER.TensionMidDiameter* is the taper tension for middle diameter (*sWU_BASIC_DATA.MidDiameter*) and *sWU_TAPER.TensionProductDiameter* is the taper tension for product diameter (*sWU_BASIC_DATA.ProductDiameter*).



For datails, refer to the following related parameters:

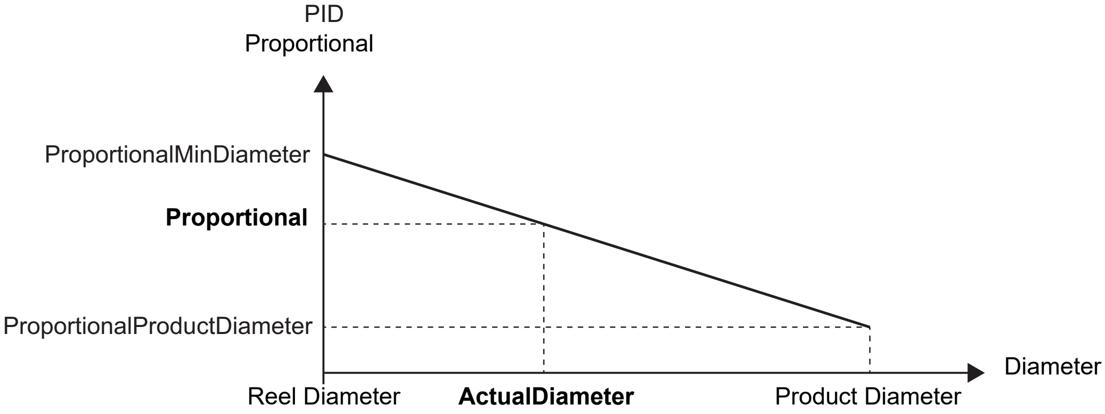
Name	Data type	Default	Valid range	Change over during execution	Description
sWU_BASIC _DATA.Reel Diameter	LREAL	40.0mm	1.0mm to 2,000.0mm	No	Reel Diameter in mm
sWU_BASIC _DATA.Mid Diameter	LREAL	100mm	Greater than Reel Diameter and lower than Product Diameter	Yes	Middle diameter in mm for Taper function
sWU_BASIC _DATA.Product Diameter	LREAL	300mm	MidDiameter < *ProductDiameter* ≤ MaxDiameter	Yes	Product Diameter in mm
sWU_TAPER .TensionMin Diameter	LREAL	100.0%	0.0% to 200.0%	Yes	Tension at minimum diameter
sWU_TAPER .TensionMid Diameter	LREAL	100.0%	0.0% to 200.0%	Yes	Tension at middle diameter
sWU_TAPER .Tension ProductDiameter	LREAL	100.0%	0.0% to 200.0%	Yes	Tension at Product diameter

- **Variable PID**

This section describes the Variable PID function.

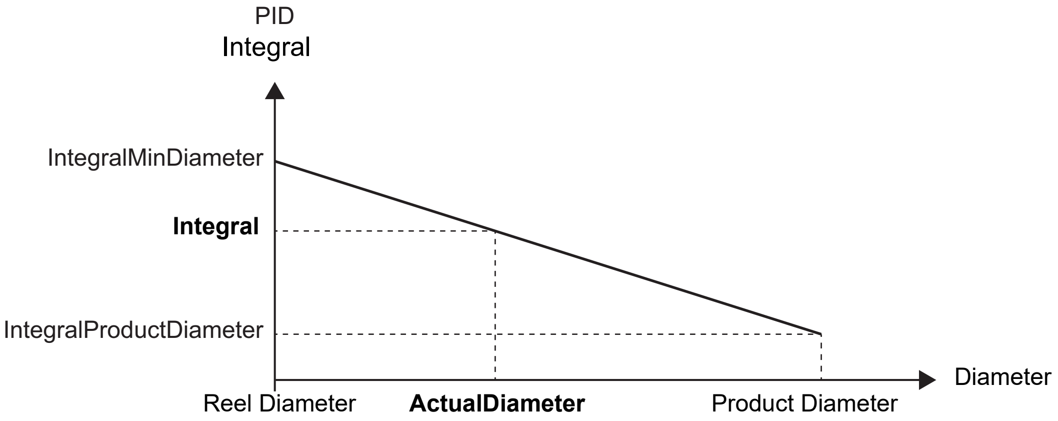
This function modifies proportional and integral PID terms relating to the diameter. For PID proportional term, set the proportional value for minimum diameter (reel diameter) in *sWU_PID.ProportinalMinDiameter* and the proportional value for product diameter in *sWU_PID.ProportionalProductDiameter*.

Variable PID - Proportional



For PID integral term, set the integral value for minimum diameter (reel diameter) in *sWU_PID.IntegralMinDiameter* and the integral value for product diameter in *sWU_PID.IntegralProductDiameter*.

Variable PID – Integral



Derivative term is not modified.

Refer to the related parameters:

Name	Data type	Default	Valid range	Change over during execution	Description
sWU_BASIC _DATA.Reel Diameter	LREAL	40.0mm	1.0mm to 2,000.0mm	No	Reel Diameter in mm
sWU_ASIC _DATA.Product Diameter	LREAL	300mm	MidDiameter < *ProductDiameter* ≤ MaxDiameter	Yes	Product Diameter in mm
sWU_PID .ProportionalMin Diameter	LREAL	1.0	0.01 to 1,000.0	Yes	PID Proportional Value at minimum diameter
sWU_PID .Proportional ProductDiameter	LREAL	1.0	0.01 to 1,000.0	Yes	PID Proportional Value at Product Diameter
sWU_PID .IntegralMin Diameter	LREAL	1.0	0.0 to 10,000	Yes	PID Integral Value at minimum Diameter
sWU_PID .IntegralProduct Diameter	LREAL	1.0	0.0 to 10,000	Yes	PID Integral Value at Product Diameter

● Convergence Error

This section describes the Convergence Error function.

It modifies the PID error. It stabilizes the PID error near the setpoint and provides a more aggressive response when the feedback is far from the setpoint as the *sWU_PID.ConvergenceGain* value increases.

When this gain increases, the PID gain can be higher.

When *sWU_PID.ConvergenceGain* is equal to 1.0, the PID error is not modified.

For details, refer to the related parameters:

Name	Data type	Default	Valid range	Change over during execution	Description
sWU_PID .ConvergenceGain	LREAL	1.0	0.1 to 5.0	Yes	To provide smooth operation near the set point.

Diameter Completion

This section describes the Diameter Completion function.

DiameterReached output becomes TRUE when the desired diameter is reached. The diameter completion value is set in *sWU_CONFIG.PercentageJobCompletion* in percentage, and diameter completion time in *sWU_CONFIG.TimeJobCompletion*.

This function can be used on winder and unwinder modes.

For details, refer to the related parameters:

Name	Data type	Default	Valid range	Change over during execution	Description
sWU_CONFIG .PercentageJob Completion	LREAL	95.0%	0 to 100%	Yes	Job Completion percentage
sWU_CONFIG.Time JobCompletion	TIME	0.5s	0 to 600.0s	Yes	Job Completion Time

Web Break Detection

This section describes Web Break Detection function.

It detects a web break (material loss) setting *WebBreakDetection* output to TRUE when the function is enabled, line speed is over a specified line speed level and tension feedback from load cell/dancer is out of the specified tension limits during a specific time. Detection result is output to *WebBreakDetection*.

Enable the web break function setting *sWU_WEBBREAK.Enable* to TRUE, set in *sWU_WEBBREAK.MinLineSpeed* the minimum line speed and specify the upper web tension limit in *sWU_WEBBREAK.UpperLimit*, lower web tension limit in *sWU_WEBBREAK.LowerLimit* and web break detection time in *sWU_WEBBREAK.WebBreakTime* for web break detection.

For datails, refer to the related parameters:

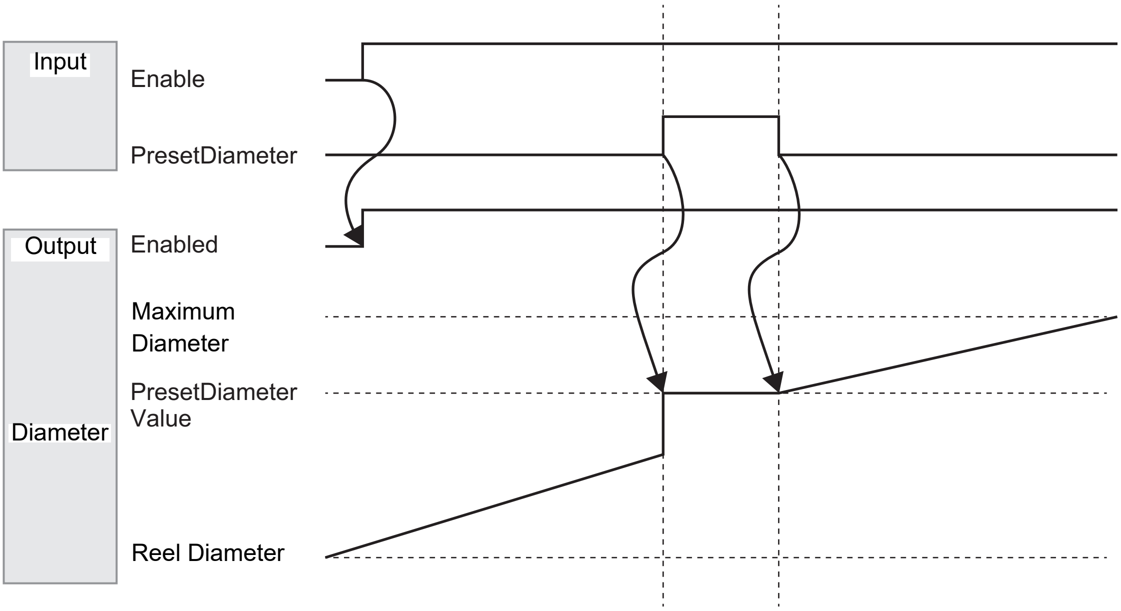
Name	Data type	Default	Valid range	Change over during execution	Description
sWU _WEBBREAK .Enable	BOOL	FALSE	TRUE/FALSE	Yes	Enable web break detection
sWU _WEBBREAK.Min LineSpeed	LREAL	10.0m/min	0 to MaxLineSpeed	Yes	Minimum Line Speed for web break detection
sWU _WEBBREAK .UpperLimit	LREAL	80.0%	Greater than Lower Limit (0% to 100%)	Yes	Upper Limit level for web break detection
sWU _WEB-BREAK.LowerLimit	LREAL	10%	Lower than Upper-Limit (0% to 100%)	Yes	Lower Limit for web break detection
sWU _WEBBREAK.Web BreakTime	TIME	1.0s	0.100 to 10.0s	Yes	Web Break Detection time

● Preset Diameter

This section describes the Preset Diameter function.

The diameter value is immediately updated to the *PresetValue when PresetInput* is set to TRUE. The preset diameter function works only when the Preset Diameter Input (*PresetDiameter*) is set to TRUE. Refer to the timing diagram:

Timing Diagram (Winder Mode)



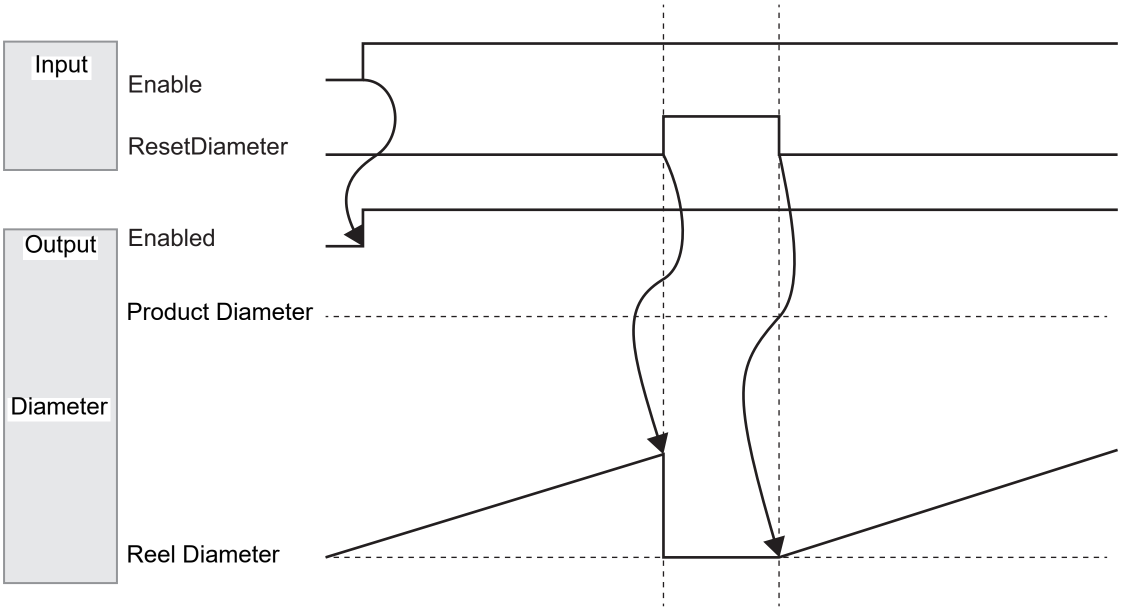
● Reset Diameter

This section describes the Reset Diameter function.

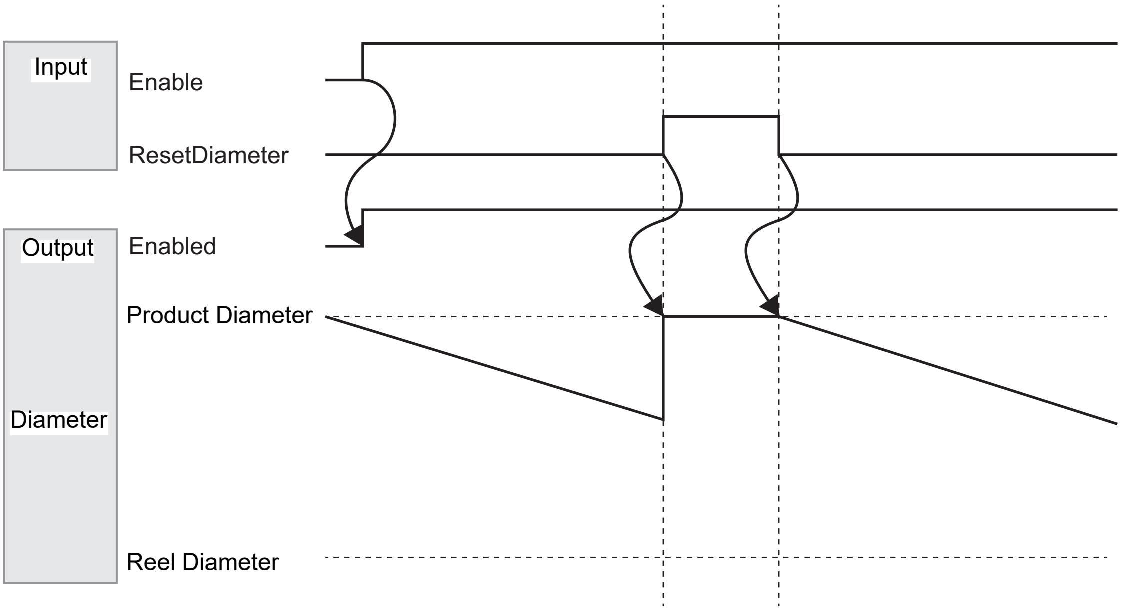
This function forces the diameter value to the initial value of the process in accordance with the winder mode criteria.

When *WinderMode* is set to Winder and the reset diameter input is TRUE, the diameter output will be set to the reel diameter. Also, if *WinderMode* is set to Unwinder and the reset diameter input is TRUE, the diameter output will be set to the product diameter (*sWU_BASIC_DATA.ProductDiameter*). The Reset Diameter function works only when the *ResetDiameter* Input is set to TRUE. Refer to the timing diagrams:

Timing Diagram – Winder Mode



Timing Diagram – Unwinder Mode

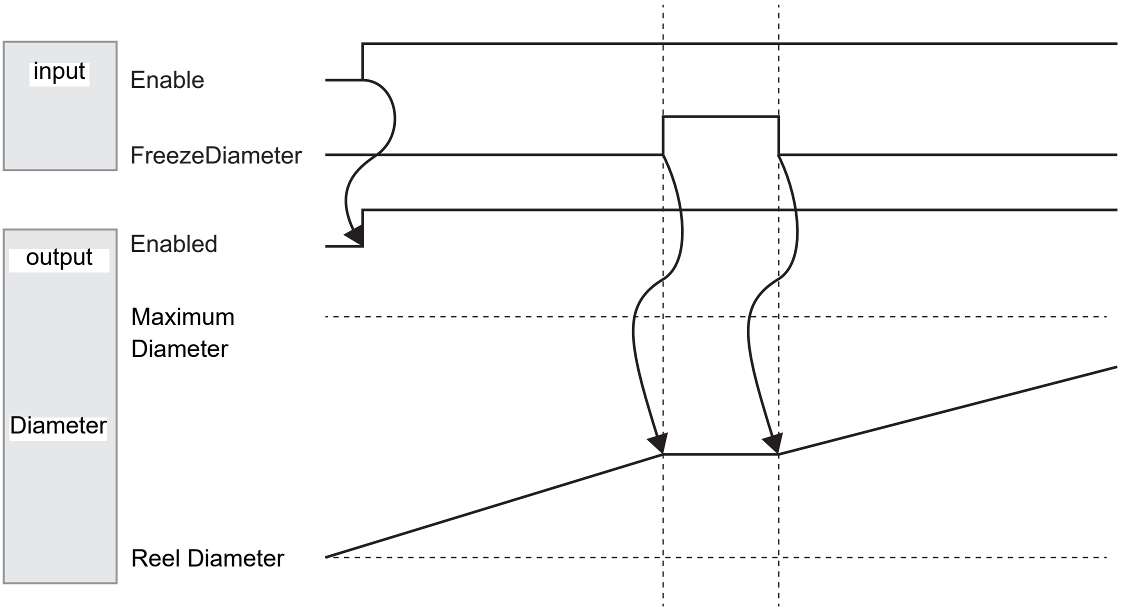


Freeze Diameter

This section describes the Freeze Diameter function.

It freezes the diameter output value when the *FreezeDiameter* input is set to TRUE. The Freeze Diameter function works only when the *FreezeDiameter* Input (*FreezeDiameter*) is set to TRUE. For details, refer to the timing diagram:

Timing Diagram (Winder Mode)



Rigidity Gain (When RX-series or MX2-series Inverter is used)

This section describes the Rigidity Gain function.

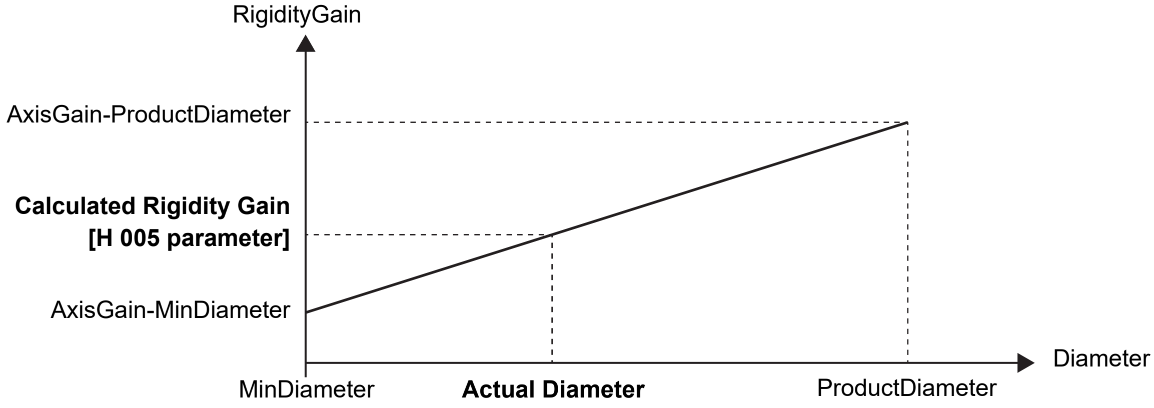
This function modifies the inverter speed response (H005 parameter) with the calculated/estimated diameter from reel diameter to product diameter.

Set *sWU_RIGIDITY_GAIN.Enable* to true and set the InverterModel in *sWU_RIGIDITY_GAIN.InverterModel* and node number in *sWU_RIGIDITY_GAIN.Node.*

Set the desired gains at minimum diameter in sWU_RIGIDITY_GAIN.GainMinDiameter and at product diameter in sWU_RIGIDITY_GAIN.GainProductDiameter. Refer to the related parameters:

sWU_RIGIDITY_GAIN

Name	Data type	Default	Valid range	Change over during execution	Description
Enable	BOOL	FALSE	TRUE/FALSE	Yes	Enable/Disable Rigidity Gain Function
InverterModel	BOOL	FALSE	TRUE/FALSE (TRUE = RX, FALSE = MX2)	No	Indicates the inverter used for the function block
GainMinDiameter	LREAL	2	RX: 0.001 to 80.000 MX2: 1 to 1,000	Yes	Rigidity Gain at minimum diameter
GainProduct Diameter	LREAL	2	RX: 0.001 to 80.000 MX2: 1 to 1,000	Yes	Rigidity Gain at product diameter
Node	UINT	1	1 to 192	No	Node to write Rigidity Gain trough SDO.



Note when rigidity gain function is enabled, do not use a command to save parameter change to EEPROM.

Setting Maximum Winder/Unwinder Speed

Follow next steps before setting the Maximum Speed setting:

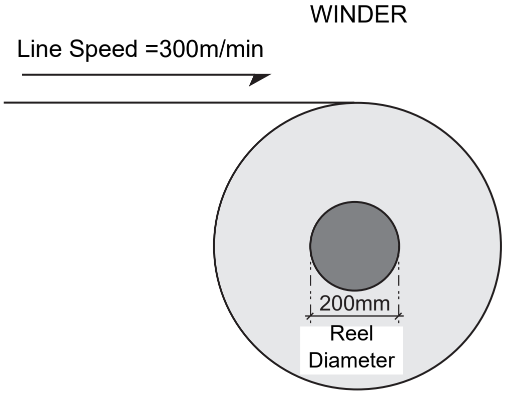
1. Calculate Maximum Motor Speed from reel diameter and maximum line speed.
2. Set Maximum Motor Speed which was calculated in Procedure 1 in the function block.
3. Set a and speed offset to the calculated speed in point 1.

   When a speed offset is not set, the PID cannot catch-up with the line speed in many cases.
4. Set inverter parameter A004 (maximum frequency) with the result obtained in the Procedure 3 trough parameter editor or digital operator.

   Maximum Frequency cannot be changed during the inverter is running.

Example:

- Motor = 50Hz/4 poles
- Gear = 3



1 Calculate Maximum Motor Speed:

$L = 2\pi r = \pi D = \pi \cdot 200mm = 628.31mm$

$MaxLineSpeed = 300m/\text{min} = 300{,}000mm/min$

$$Winder\_Unwinder_{Speed} = \frac{300,000mm/min}{628.31mm} = 477.47\ rpm$$

$Winder\_Unwinder_{Shaft\_Speed} = Winder\_Unwinder_{Speed} \cdot GearRatio$

$Motor_{Shatf\_Speed} = 477.47\ rpm \cdot 3 = 1432.41\ rpm$

$$Motor_{Shaft\_Speed} = \frac{1,432.41rpm}{1,500rpm} \cdot 50Hz = 47.74\ Hz$$

2 Set Stwinder.BasicData.MaxMotorSpeed = 47.74Hz

3 Assign a Speed Offset. Set 10% in this example.

4 Set Inverter Maximum Frequency:

A004 = 47.74Hz · 1.10 = 52.52 Hz

A004 = 52.52Hz

Setting Diameter Sensor Configuration

This section describes how to configure the use of a diameter sensor in the winder/unwinder function block.

In this function block, the diameter sensor can be used. If using a diameter sensor in the winder/unwinder application, set *sWU_CONFIG.SensorMode* to TRUE.

Specify the minimum sensor value (*sWU_CONFIG.MinSensorValue*) and maximum sensor value (*sWU_CONFIG.MaxSensorValue*) in the function block according to the minimum and maximum diameter of the winder/unwinder. Set the relevant reel diameter in mm (*sWU_BASIC_DATA.ReelDiameter*). For details, refer to the related input variables:

● sWU_BASIC_DATA

Name	Data type	Default	Valid range	Change over during execution	Description
ReelDiameter	LREAL	40.0mm	1.0mm to 2,000.0mm	No	Reel Diameter in mm

- **sWU_CONFIG**

Name	Data type	Default	Valid range	Change over during execution	Description
SensorMode	BOOL	FALSE	TRUE/FALSE	No	Diameter sensor use
DiameterSensor	LREAL	0	Positive number (MinSensorValue to MaxSensorValue)	Yes	Diameter Sensor value
MinSensorValue	LREAL	0	Positive Number (Less than MaxSensorValue)	No	Minimum Diameter sensor value
MaxSensorValue	LREAL	1,000	Greater than min sensor value	No	Maximum Diameter Sensor value

Setting Diameter Filter

This section describes how to set diameter filter.

In real winder/unwinder application, diameter sensor value or internal estimated diameter value may fluctuate. For filtering the instantaneous diameter, define the number of samples in *sWU_CONFIG.VarianceSize* and set *sWU_CONFIG.FilterTime1* and *sWU_CONFIG.FilterTime2* time values for the diameter filter. Refer to the related input variables:

- **sWU_CONFIG**

Name	Data type	Default	Valid range	Change over during execution	Description
VarianceSize	DINT	100	0 to 5,000	No	Variance Size for diameter calculation
FilterTime1	TIME	0.15s	(Less or equal to FilterTime2) From 0 to 600.0s	Yes	Filter Time 1 for diameter calculation
FilterTime2	TIME	0.5s	From FilterTime1 to 600.0 s	Yes	Filter Time 2 for diameter calculation

Setting Inverter Model

This function block can work with the next two OMRON inverter models:

- 3G3RX-series
- 3G3MX2-series

Set *sWU_RIGIDITY_GAIN.InverterModel* to FALSE if 3G3MX2 inverter is used, or set to TRUE if 3G3RX inverter is used.

- **sWU_RIGIDITY_GAIN**

Name	Data type	Default	Valid range	Change over during execution	Description
InverterModel	BOOL	FALSE	TRUE/FALSE (TRUE = RX, FALSE = MX2)	No	Indicates the inverter that is used in the function block.

Setting Rotation Direction

This section describes how to set rotation direction.

When *sWU_RUN_STATUS.RunningForward* (Input variable) is set to TRUE, it changes to running forward status. When *sWU_RUN_STATUS.RunningReverse* (Input variable) is set to TRUE, it changes to running reverse status.

When the both input variables are set to TRUE, ERROR (Error code: 16#0002 000A) changes to TRUE. When the both variables are set to FALSE, *WinderSpeedReference* (Output variable) changes to 0. Therefore, set one of the input variables to TRUE.

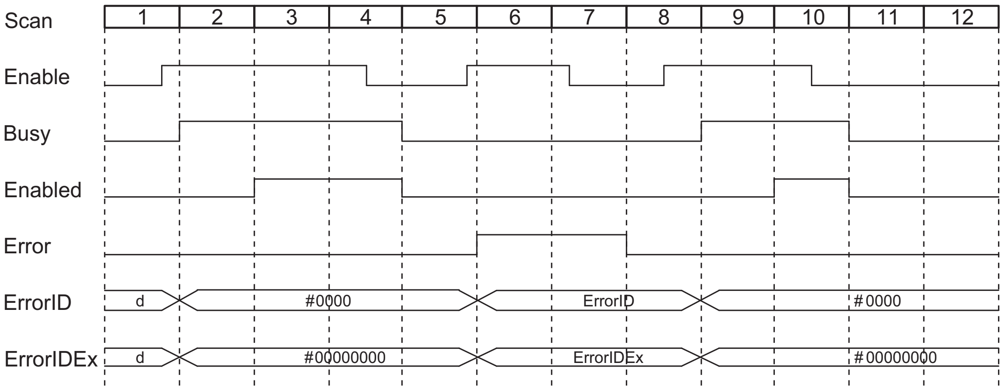
Refer to the related input variables:

sWU_RUN_STATUS

Name	Data type	Default	Valid range	Change over during execution	Description
RunningForward	BOOL	FALSE	TRUE/FALSE	Yes	Run Forward status
RunningReverse	BOOL	FALSE	TRUE/FALSE	Yes	Run Reverse status

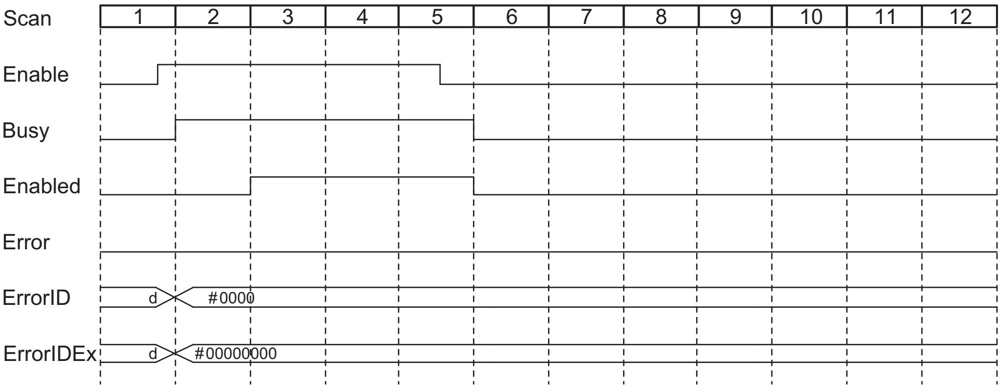
Timing Charts

Normal end, or Error end to Normal completion

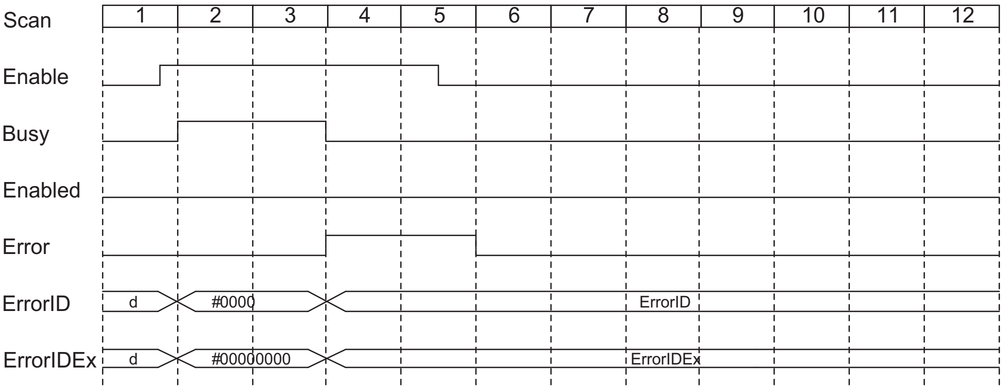


- The processing is executed when the *Enable* input variable is TRUE.
- After the completion of the processing series, the *Enabled* output variable changes to TRUE.
- If an error occurs and the *Error* output variable changes to TRUE, the next processing does not continue even if the *Enable* input variable remains TRUE.
- The *Error* output variable retains its value while the *Enable* input variable remains TRUE
- The *ErrorID* and *ErrorIDEx* output variables retain their values even when the processing is executed when the *Enable* input variable is FALSE.
- When *Enable* changes TRUE->FALSE->TRUE after resetting an error, the error code is cleared to zero.

Normal end: The *Enable* output variable is TRUE for multiple periods



Error completion: The *Error* output variable is TRUE for multiple periods.



- At error, *Error* keeps TRUE while *Enable* is TRUE.
- The dotted lines of the Scan shown in the figure indicate multiple task periods.
  For information on the error code and cause of error, refer to *Troubleshooting* on page 197.

Troubleshooting

ErrorID

The following table lists the error codes that are stored in *ErrorID*.

Error code	Status	Description	Correction
16#0000	Normal end (Normal)	---	---
16#3C46	Error end (Error)	An error occurred in a standard instruction that is used in the function block.	Check the last four digits of the error code which were sotred in *ErrorIDEx*. Take countermeasures according to instructions in instructions reference manual.
16#3C47	Error end (Error)	An error occurred in the function block.	Check the error code stored in *ErrorIDEx*.

ErrorIDEx

The following table lists the error codes that are stored in *ErrorIDEx* when #2801 is stored in the *ErrorID* output variable.

Expansion error code	Status	Description	Correction
16#0002 0001	Invalid Input Parameter	Reel Diameter value is out of range.	Check Reel Diameter value.
16#0002 0002	Invalid Input Parameter	Middle diameter is lower or equal to reel diameter or middle diameter is bigger or equal to product diameter	Check middle diameter value.
16#0002 0003	Invalid Input Parameter	Maximum diameter is lower or equal to reel diameter or maximum diameter is bigger than maximum diameter ratio.	Check Maximum diameter value
16#0002 0004	Invalid Input Parameter	Product diameter is bigger than maximum diameter	Check Product Diameter value.
16#0002 0005	Invalid Input Parameter	Maximum Line Speed is out of range.	Check Maximum Line Speed setting.
16#0002 0006	Invalid Input Parameter	Maximum Motor Speed is out of range.	Check Maximum Motor Speed setting.
16#0002 0007	Invalid Input Parameter	Minimum Diameter Sensor Value is bigger or equal to maximum diameter sensor value when a diameter sensor is used	Check Minimum Diameter sensor value.
16#0002 0008	Invalid Input Parameter	Variance Size is out of range...	Check Variance Size.
16#0002 0009	Invalid Input Parameter	Preset Value is bigger than Product Diameter or Maximum Diameter.	Check Diameter Preset Value.
16#0002 000A	Invalid Input Parameter	Both signals running Forward and running Reverse are activated simultaneously.	Check running forward and running reverse signals.
16#0002 000B	Invalid Input Parameter	Minimum diameter sensor value is lower than 0.	Check minimum diameter sensor value.
16#0002 000C	Invalid Input Parameter	Maximum diameter sensor value is lower than 0.	Check maximum diameter sensor value.
16#0002 000D	Invalid Input Parameter	With Diameter Sensor, the actual diameter sensor value is bigger than the maximum diameter sensor value.	Check Diameter sensor value settings.
16#0002 000E	Invalid Input Parameter	PID Task time out of range	Check PID task time setting.

Expansion error code	Status	Description	Correction
16#0002 0010	Invalid Input Parameter	With Diameter Sensor, the actual diameter sensor value is lower than the minimum diameter sensor value.	Check Diameter sensor value settings.
16#0002 0011	Invalid Input Parameter	Job Completion Time is out of range	Check Job Completion Time value.
16#0002 0013	Invalid Input Parameter	Preset Diameter input function and Reset Diameter input function are activated simultaneously.	Check Preset and Reset Diameter input signals.
16#0002 0014	Invalid Input Parameter	Preset Diameter input function and Reset Diameter input function and Freeze Diameter input function are activated simultaneously.	Check Preset, Reset and Freeze Diameter Input Signals.
16#0002 0015	Invalid Input Parameter	Node number out of range 1 to 192	Check node number.
16#0002 0016	Invalid Input Parameter	Line Speed value out of range	Check Line Speed Value.
16#0002 0017	Invalid Input Parameter	With MX2 inverter, Rigidity Gain function is enabled and Gain at Minimum Diameter is out of range MX2 H005 parameter 1 to 1,000.	Check Gain at Minimum Diameter for Rigidity function.
16#0002 0018	Invalid Input Parameter	With MX2 inverter, Rigidity Gain function is enabled and Gain at Product Diameter is out of range MX2 H005 parameter 1 to 1,000	Check Gain at Product Diameter for Rigidity function.
16#0002 0019	Invalid Input Parameter	With RX inverter, Rigidity Gain function is enabled and Gain at Minimum Diameter is out of range RX H005 parameter 0.001 to 80.000	Check Gain at Minimum Diameter for Rigidity function.
16#0002001A	Invalid Input Parameter	With RX inverter, Rigidity Gain function is enabled and Gain at Product Diameter is out of range RX H005 parameter 0.001 to 80.000	Check gain at Product Diameter for Rigidity function.
16#0002 001B	Invalid Input Parameter	Taper Tension at Minimum Diameter is bigger than 200.0%.	Check Taper Tension at minimum diameter.
16#0002 001C	Invalid Input Parameter	Taper Tension at Middle Diameter is bigger than 200.0%.	Check Taper Tension at middle diameter.
16#0002 001D	Invalid Input Parameter	Taper Tension at Product Diameter is bigger than 200.0%.	Check Taper Tension at Product diameter
16#0002 001F	Invalid Input Parameter	Tension Reference is lower than 0% or bigger than 100%	Check Tension Reference setting.
16#0002 0020	Invalid Input Parameter	Tension Feedback is lower than 0% or bigger than 100%	Check Tension Feedback.
16#0002 0021	Invalid Input Parameter	Diameter Filter Time 1 is out of range.	Check Diameter Filter Timer 1.
16#0002 0022	Invalid Input Parameter	Diameter Filter Time 2 is out of range.	Check Diameter Filter Time 2.
16#0002 0023	Invalid Input Parameter	Percentage Diameter completion is out of range.	Check Percentage Diameter completion setting.
16#0002 0024	Invalid Input Parameter	PID Proportional at Minimum Diameter value is out of range	Check PID Proportional at Minimum Diameter setting.
16#0002 0025	Invalid Input Parameter	PID Proportional at Product Diameter value is out of range.	Check PID Proportional at Product Diameter setting.
16#0002 0026	Invalid Input Parameter	PID Integral at Minimum Diameter value is out of range.	Check PID Integral at Minimum Diameter setting.
16#0002 0027	Invalid Input Parameter	PID Integral at Product Diameter value is out of range.	Check PID Integral at Product Diameter setting.
16#0002 0028	Invalid Input Parameter	PID Derivative value is out of range.	Check PID Derivative setting.

Expansion error code	Status	Description	Correction
16#0002 0029	Invalid Input Parameter	PID Convergence Gain value is out of range.	Check PID Convergence Gain setting.
16#0002 0030	Invalid Input Parameter	Inverter Model does not match with EtherCAT node.	Check inverter model selection or node number.
16#0002 0031	Invalid Input Parameter	Diameter Filter Time 2 is lower than filter time 1 value	Check Diameter Filter Time settings.
16#0002 0032	Invalid Input Parameter	PID Lower Limit is out of range.	Check PID Lower Limit setting.
16#0002 0033	Invalid Input Parameter	PID Upper Limit is out of range.	Check PID Upper Limit.
16#0002 0034	Invalid Input Parameter	PID Output Scale is out of range.	Check PID Output Scale value.
16#0002 0035	Invalid Input Parameter	Minimum Line Speed for web break function is out of range	Check Minimum Line Speed value.
16#0006 0001	Invalid Input Parameter	Web Break Lower Limit value is lower than zero, or Web Break Upper Limit value is lower than zero.	Check Web Break Lower Limit or Web Break Upper Limit setting.
16#0006 0002	Invalid Input Parameter	Web Break Upper Limit value is bigger than 100.0%, or Web Break Lower Limit value is bigger than 100.0.	Check Web Break Upper Limit or Web Break Lower Limit setting.
16#0006 0003	Invalid Input Parameter	Web break Upper Limit value is lower or equal to web break Lower Limit value	Check Web Break Upper Limit setting.
16#0006 0007	Invalid Input Parameter	Web Break Time is out of range.	Check Web Break Detection time setting.

Precautions for Correct Use

Multiple error codes listed above may occur at the same time. In this case, the most recent exit code will be output.

Error Reset Procedure

● Error Ends When this function block is Started or Being Executed

1. Refer to *Troubleshooting* on page 197 to find and check the cause of the error.
2. Remove the cause of the error.
3. Refer to *Conditions for Execution* on page 179 and check the execution conditions of the function block.
4. Make sure that the execution conditions are met. By changing *Enable* of input variable to TRUE, this function block can be executed.

● When an Error Occurred in the NJ/NX/NY

1. Find and check the cause of the error.
2. Remove the cause of the error.
3. Reset/clear the error occurred in the NJ/NX/NY.
4. After resetting the error, refer to *Conditions for Execution* on page 179 and check the execution conditions of the function block.
5. Make sure that the execution conditions are met. By changing *Enable* of input variable to TRUE, this function block can be executed.

For information on how to reset errors in NJ/NX/NY, refer to the troubleshooting manual.

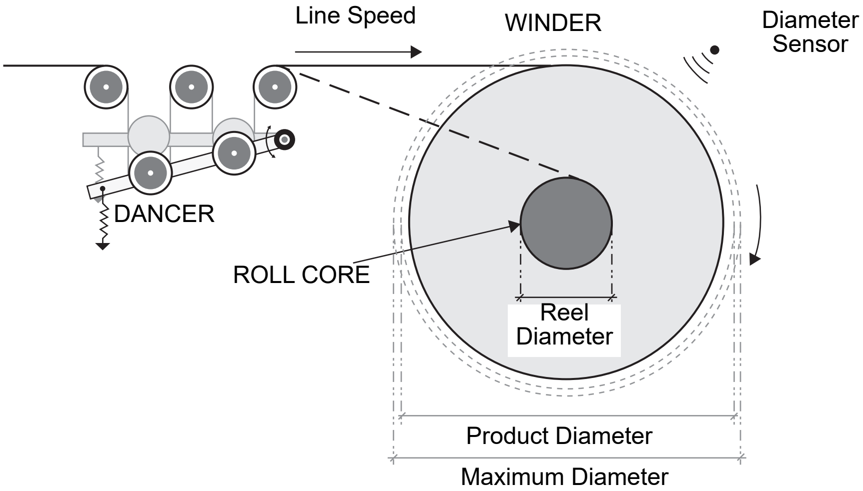
Sample Programming

This section describes WinderVelCtrl_Inverter function block in a sample program.

Machine Sample

● Overview and Dimensions

This sample system winds the material with 3G3RX inverter for the winder using a diameter sensor and a dancer.



Each dimension is shown in the table below:

Name	Value
Reel Diameter	40mm
Product Diameter	300mm
Maximum Diameter	300mm
Line Speed	75m/min (max. line speed)
Gear Box	1:5
Three-phase inductive motor	400VAC/400W/50Hz/4 poles

● Hardware configuration

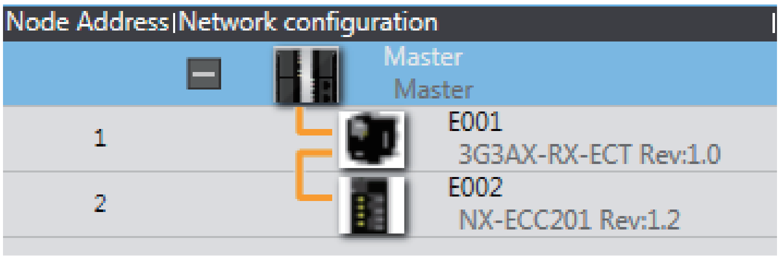
The hardware configuration of the above sample system is as follows.

Name	Type	Version	Number
Machine Automation Controller	NJ501-1300	Ver.1.10	---
3G3RX-series inverter	3G3RX-□□□-□□□	Ver.1.0	---
Encoder Line Speed	---	---	---
Diameter sensor	---	---	---

Project Sample

● EtherCAT Network Configuration

The EtherCAT network configuration is as follows.

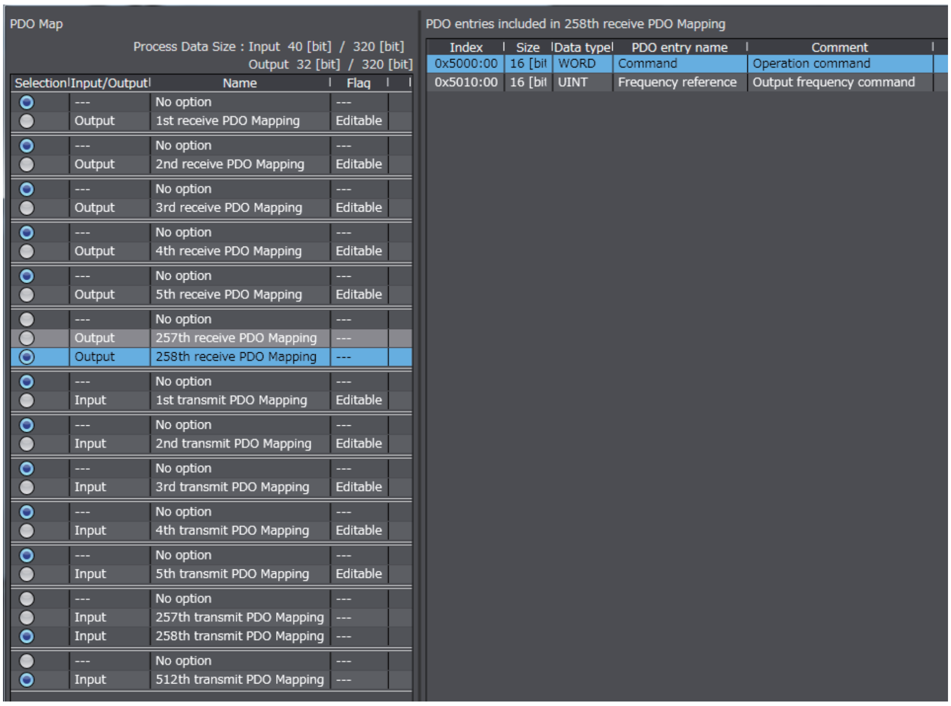


Node Address	Network configuration
	Master Master
1	E001 3G3AX-RX-ECT Rev:1.0
2	E002 NX-ECC201 Rev:1.2

● PDO Map

Set PDO Mapping as shown below.

Output



PDO Map

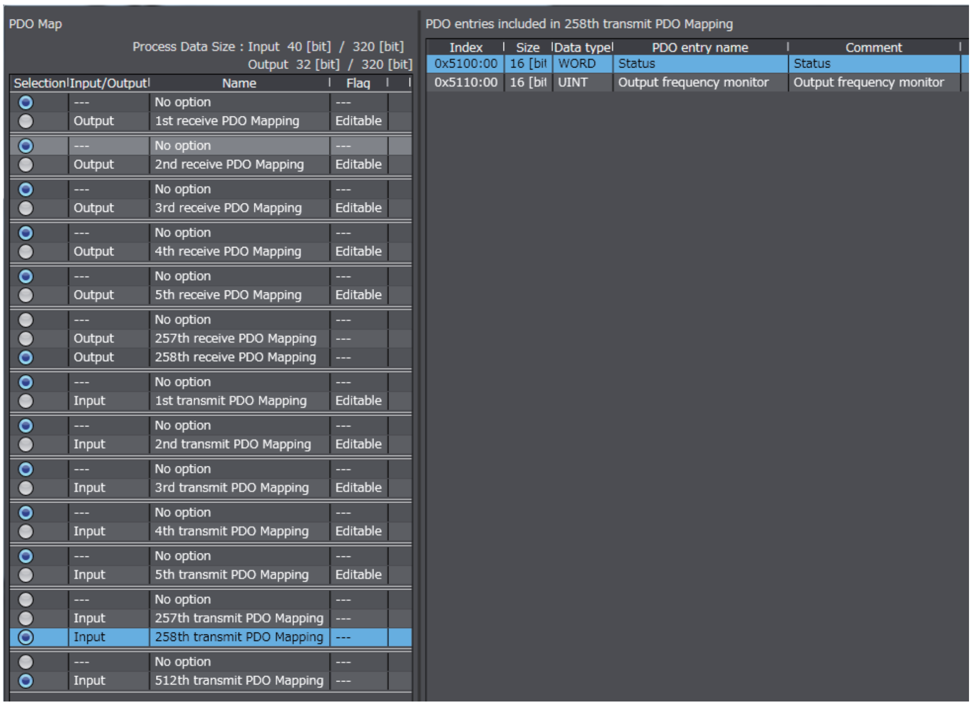
Process Data Size : Input 40 [bit] / 320 [bit]
Output 32 [bit] / 320 [bit]

Selection	Input/Output	Name	Flag
●	---	No option	---
○	Output	1st receive PDO Mapping	Editable
●	---	No option	---
○	Output	2nd receive PDO Mapping	Editable
●	---	No option	---
○	Output	3rd receive PDO Mapping	Editable
●	---	No option	---
○	Output	4th receive PDO Mapping	Editable
●	---	No option	---
○	Output	5th receive PDO Mapping	Editable
○	---	No option	---
○	Output	257th receive PDO Mapping	---
●	Output	258th receive PDO Mapping	---
●	---	No option	---
○	Input	1st transmit PDO Mapping	Editable
●	---	No option	---
○	Input	2nd transmit PDO Mapping	Editable
●	---	No option	---
○	Input	3rd transmit PDO Mapping	Editable
●	---	No option	---
○	Input	4th transmit PDO Mapping	Editable
●	---	No option	---
○	Input	5th transmit PDO Mapping	Editable
○	---	No option	---
○	Input	257th transmit PDO Mapping	---
●	Input	258th transmit PDO Mapping	---
○	---	No option	---
●	Input	512th transmit PDO Mapping	---

PDO entries included in 258th receive PDO Mapping

Index	Size	Data type	PDO entry name	Comment
0x5000:00	16 [bit	WORD	Command	Operation command
0x5010:00	16 [bit	UINT	Frequency reference	Output frequency command

Input



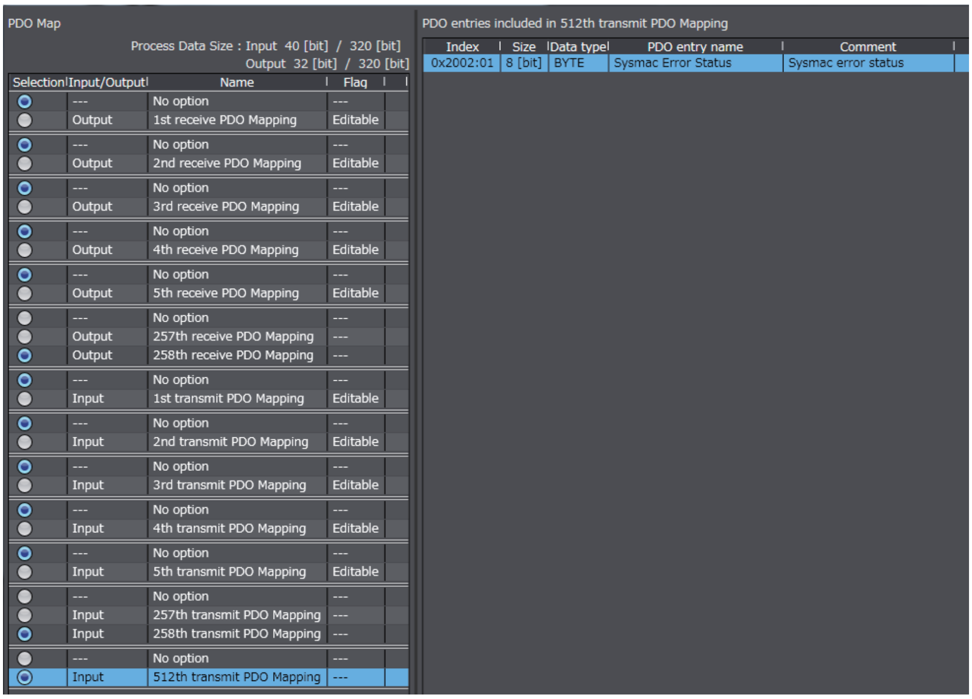
PDO Map

Process Data Size : Input 40 [bit] / 320 [bit]
Output 32 [bit] / 320 [bit]

Selection	Input/Output	Name	Flag
●	---	No option	---
○	Output	1st receive PDO Mapping	Editable
●	---	No option	---
○	Output	2nd receive PDO Mapping	Editable
●	---	No option	---
○	Output	3rd receive PDO Mapping	Editable
●	---	No option	---
○	Output	4th receive PDO Mapping	Editable
●	---	No option	---
○	Output	5th receive PDO Mapping	Editable
○	---	No option	---
○	Output	257th receive PDO Mapping	---
●	Output	258th receive PDO Mapping	---
●	---	No option	---
○	Input	1st transmit PDO Mapping	Editable
●	---	No option	---
○	Input	2nd transmit PDO Mapping	Editable
●	---	No option	---
○	Input	3rd transmit PDO Mapping	Editable
●	---	No option	---
○	Input	4th transmit PDO Mapping	Editable
●	---	No option	---
○	Input	5th transmit PDO Mapping	Editable
○	---	No option	---
○	Input	257th transmit PDO Mapping	---
●	Input	258th transmit PDO Mapping	---
○	---	No option	---
●	Input	512th transmit PDO Mapping	---

PDO entries included in 258th transmit PDO Mapping

Index	Size	Data type	PDO entry name	Comment
0x5100:00	16 [bit	WORD	Status	Status
0x5110:00	16 [bit	UINT	Output frequency monitor	Output frequency monitor



PDO Map

Process Data Size : Input 40 [bit] / 320 [bit]
Output 32 [bit] / 320 [bit]

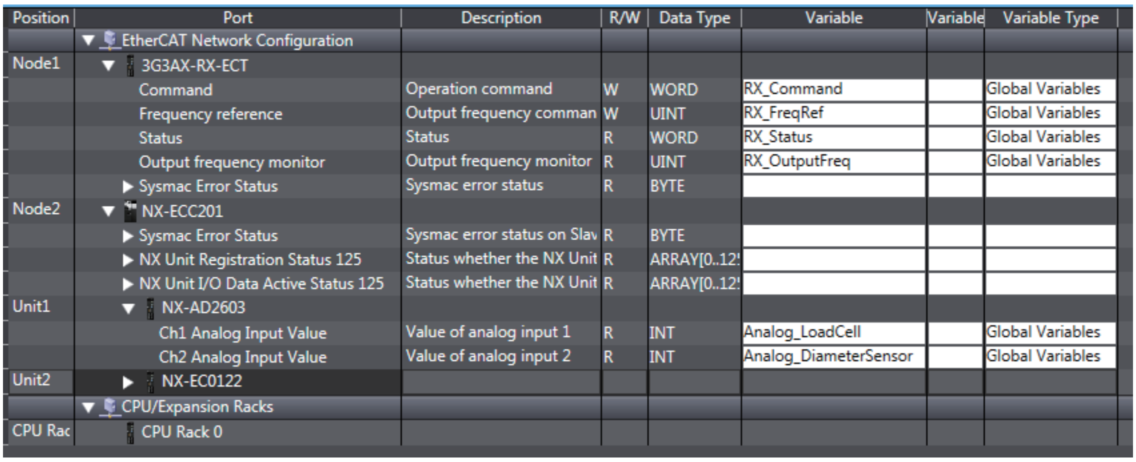
Selection	Input/Output	Name	Flag
●	---	No option	---
○	Output	1st receive PDO Mapping	Editable
●	---	No option	---
○	Output	2nd receive PDO Mapping	Editable
●	---	No option	---
○	Output	3rd receive PDO Mapping	Editable
●	---	No option	---
○	Output	4th receive PDO Mapping	Editable
●	---	No option	---
○	Output	5th receive PDO Mapping	Editable
○	---	No option	---
○	Output	257th receive PDO Mapping	---
●	Output	258th receive PDO Mapping	---
●	---	No option	---
○	Input	1st transmit PDO Mapping	Editable
●	---	No option	---
○	Input	2nd transmit PDO Mapping	Editable
●	---	No option	---
○	Input	3rd transmit PDO Mapping	Editable
●	---	No option	---
○	Input	4th transmit PDO Mapping	Editable
●	---	No option	---
○	Input	5th transmit PDO Mapping	Editable
○	---	No option	---
○	Input	257th transmit PDO Mapping	---
●	Input	258th transmit PDO Mapping	---
○	---	No option	---
●	Input	512th transmit PDO Mapping	---

PDO entries included in 512th transmit PDO Mapping

Index	Size	Data type	PDO entry name	Comment
0x2002:01	8 [bit]	BYTE	Sysmac Error Status	Sysmac error status

I/O Map

Set I/O mapping as shown below.



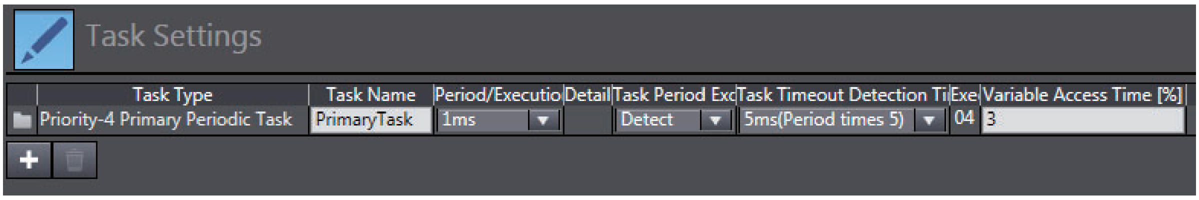
Position	Port	Description	R/W	Data Type	Variable	Variable	Variable Type
	▼ EtherCAT Network Configuration						
Node1	▼ 3G3AX-RX-ECT						
	Command	Operation command	W	WORD	RX_Command		Global Variables
	Frequency reference	Output frequency comman	W	UINT	RX_FreqRef		Global Variables
	Status	Status	R	WORD	RX_Status		Global Variables
	Output frequency monitor	Output frequency monitor	R	UINT	RX_OutputFreq		Global Variables
	▶ Sysmac Error Status	Sysmac error status	R	BYTE			
Node2	▼ NX-ECC201						
	▶ Sysmac Error Status	Sysmac error status on Slav	R	BYTE			
	▶ NX Unit Registration Status 125	Status whether the NX Unit	R	ARRAY[0..12!			
	▶ NX Unit I/O Data Active Status 125	Status whether the NX Unit	R	ARRAY[0..12!			
Unit1	▼ NX-AD2603						
	Ch1 Analog Input Value	Value of analog input 1	R	INT	Analog_LoadCell		Global Variables
	Ch2 Analog Input Value	Value of analog input 2	R	INT	Analog_DiameterSensor		Global Variables
Unit2	▶ NX-EC0122						
	▼ CPU/Expansion Racks						
CPU Rac	CPU Rack 0						

Task Settings

The task settings are as follows.

Item	Settings
Task Type	Primary Periodic Task
Task Period	1ms
Assigned Program to the Task	Program0

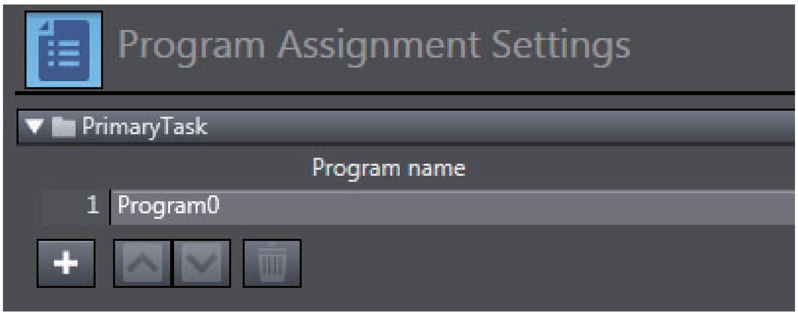
Task Settings



Task Settings

Task Type	Task Name	Period/Executio	Detail	Task Period Exc	Task Timeout Detection Ti	Exe	Variable Access Time [%]
Priority-4 Primary Periodic Task	PrimaryTask	1ms		Detect	5ms(Period times 5)	04	3

Program assignment Settings



Program Assignment Settings

▼ PrimaryTask

	Program name
1	Program0

● Variable List

Internals

Name	Data Type	Initial Value	Comment
INV_RXCommand	Data_Bit16		
INV_RXStatus	Data_Bit16		
Winder	OmronLib\PKG_WU\WinderVelCtrl_Inverter		
WinderInv	OmronLib\PKG_WU\sWU_INVERTER	Refer to the following description.	
Error	BOOL		
ErrorID	WORD		
ErrorIDEx	DWORD		

Externals

Name	Data Type	Constant	Comment
EncoderLine	_sAXIS_REF	TRUE	
RX_Command	WORD	FALSE	
RX_Status	WORD	FALSE	
RX_FreqRef	UINT	FALSE	
RX_OutputFreq	UINT	FALSE	
Analog_LoadCell	INT	FALSE	
Analog_DiameterSensor	INT	FALSE	
HMI_EnableFB	BOOL	FALSE	
HMI_WinderMode	BOOL	FALSE	
InverterRunFw	BOOL	FALSE	
InverterRunRev	BOOL	FALSE	
InverterReset	BOOL	FALSE	
FinalWinderSpeedReference	LREAL	FALSE	
InverterDuringFw	BOOL	FALSE	
InverterDuringRv	BOOL	FALSE	
InverterFault	BOOL	FALSE	
RXFrequencyMonitor	LREAL	FALSE	
ActualMotorSpeed	LREAL	FALSE	
ActualMotorSpeedPer	LREAL	FALSE	
FinalWinderSpeedRefPer	LREAL	FALSE	
ActualLineSpeedPer	LREAL	FALSE	
ActualLineSpeed	LREAL	FALSE	
EncoderLineSpeed	LREAL	FALSE	
LoadCellFeedback_Value	LREAL	FALSE	
MinLoadCellSensor_Value	INT	FALSE	
MaxLoadCellSensor_Value	INT	FALSE	
ActualDiameterSensorValue	LREAL	FALSE	
PresetInput	BOOL	FALSE	
DiameterPresetValue	LREAL	FALSE	
ResetInput	BOOL	FALSE	
FreezeInput	BOOL	FALSE	
ActualDiameterFil	LREAL	FALSE	
DiameterReached	BOOL	FALSE	
WebBreakDetection	BOOL	FALSE	
TensionSetPoint	LREAL	FALSE	

Global Variables

Name	Data Type	Initial Value	AT	Constant	Comment
RX_Command	WORD		ECAT://node#1/Command	FALSE	
RX_Status	WORD		ECAT://node#1/Status	FALSE	
RX_FreqRef	UINT		ECAT://node#1/Frequency reference	FALSE	
RX_OutputFreq	UINT		ECAT://node#1/Output frequency monitor	FALSE	
Analog_LoadCell	INT		“ECAT://node#[2,1]/Ch1 Analog Input Value”	FALSE	
Analog_DiameterSensor	INT		“ECAT://node#[2,1]/Ch2 Analog Input Value”	FALSE	
HMI_EnableFB	BOOL			FALSE	
HMI_WinderMode	BOOL			FALSE	
InverterRunFw	BOOL			FALSE	
InverterRunRev	BOOL			FALSE	
InverterReset	BOOL			FALSE	
FinalWinderSpeedReference	LREAL			FALSE	
InverterDuringFw	BOOL			FALSE	
InverterDuringRv	BOOL			FALSE	
InverterFault	BOOL			FALSE	
RXFrequencyMonitor	LREAL			FALSE	
ActualMotorSpeed	LREAL			FALSE	
ActualMotorSpeedPer	LREAL			FALSE	
FinalWinderSpeedRefPer	LREAL			FALSE	
ActualLineSpeedPer	LREAL			FALSE	
ActualLineSpeed	LREAL			FALSE	
EncoderLineSpeed	LREAL			FALSE	
LoadCellFeedback_Value	LREAL			FALSE	
MinLoadCellSensor_Value	INT			FALSE	
MaxLoadCellSensor_Value	INT			FALSE	
ActualDiameterSensorValue	LREAL			FALSE	
PresetInput	BOOL			FALSE	
DiameterPresetValue	LREAL			FALSE	
ResetInput	BOOL			FALSE	
FreezeInput	BOOL			FALSE	
ActualDiameterFil	LREAL			FALSE	
DiameterReached	BOOL			FALSE	
WebBreakDetection	BOOL			FALSE	
TensionSetPoint	LREAL			FALSE	
EncoderLine	_sAXIS_REF		MC://_MC_AX[0]	TRUE	

Union

Name	Data Type	Initial Value
Data_Bit16	UNION	
bit	ARRAY[0..15] OF BOOL	
asword	WORD	

Setting *WinderInv.BasicData* (Data Type: sWU_BASIC_DATA)

Name	Data Type	Initial Value
ReelDiameter	LREAL	42
MidDiameter	LREAL	150
MaxDiameter	LREAL	290
ProductDiameter	LREAL	285
MaxLineSpeed	LREAL	50
ActualMotorSpeed	LREAL	0.0
MaxMotorSpeed	LREAL	63.15

Setting *WinderInv.Config* (Data Type: sWU_CONFIG)

Name	Data Type	Initial Value
SensorMode	BOOL	TRUE
DiameterSensor	LREAL	0.0
MinSensorValue	LREAL	0.0
MaxSensorValue	LREAL	4350
VarianceSize	DINT	500
FilterTime1	TIME	T#5S
FilterTime2	TIME	T#10S
PercentageJobCompletion	LREAL	98
TimeJobCompletion	TIME	T#0.5S

Setting *stWinderInv.Status* (Data Type:_sWU_RUN_STATUS)

Name	Data Type	Initial Value
RunningForward	BOOL	FALSE
RunningReverse	BOOL	FALSE

Setting *stWinderInv.PID* (Data Type: sWU_PID)

Name	Data Type	Initial Value
ProportionalMinDiameter	LREAL	0.02
ProportionalProductDiameter	LREAL	0.02
IntegralMinDiameter	LREAL	0.01
IntegralProductDiameter	LREAL	0.01
Derivative	LREAL	0.0
ConvergenceGain	LREAL	3
TaskTime	LREAL	0.001
PIDOutputScale	LREAL	1
PIDUpperLimit	LREAL	100
PIDLowerLimit	LREAL	-100

Setting *stWinderInv.AxisGain* (Data Type: sWU_RIGIDITY_GAIN)

Name	Data Type	Initial Value
Enable	BOOL	FALSE
InverterModel	BOOL	TRUE
GainMinDiameter	LREAL	100
GainProductDiameter	LREAL	300
Node	UINT	1

Setting *stWinderInv.Taper* (Data Type: sWU_TAPER)

Name	Data Type	Initial Value
TensionMinDiameter	LREAL	100
TensionMidDiameter	LREAL	100
TensionProductDiameter	LREAL	100

Setting *stWinderInv.WebBreak* (Data Type: sWU_WEBBREAK)

Name	Data Type	Initial Value
Enable	BOOL	FALSE
MinLineSpeed	LREAL	10
UpperLimit	LREAL	80
LowerLimit	LREAL	10
WebBreakTime	TIME	T#2S

Ladder Diagram

A sample program with WinderVelCtrl_Inverter

1

```
(*
----------------------------------------------------------------------
RX - PDO DATA
----------------------------------------------------------------------
(*
----------------------------------------------------------------------
RX - Write To Command RX PDO
----------------------------------------------------------------------
*)
INV_RXCommand.bit[0] := InverterRunFw;
INV_RXCommand.bit[1] := InverterRunRev;
INV_RXCommand.bit[7] := InverterReset;
RX_Command := INV_RXCommand.asword;
(*
----------------------------------------------------------------------
RX - Write To Frequency Reference PDO
----------------------------------------------------------------------
*)
RX_FreqRef:=LREAL_TO_UINT(FinalWinderSpeedReference * 100);
(*
----------------------------------------------------------------------
RX - Read the RX PDO Status
----------------------------------------------------------------------
*)
INV_RXStatus.asword := RX_Status;
InverterDuringFw := INV_RXStatus.bit[0];
InverterDuringRv := INV_RXStatus.bit[1];
InverterFault := INV_RXStatus.bit[3];

//Read PDO -inverter Output Frequency
RXFrequencyMonitor := UINT_TO_LREAL(RX_OutputFreq)/ 100.0;
```

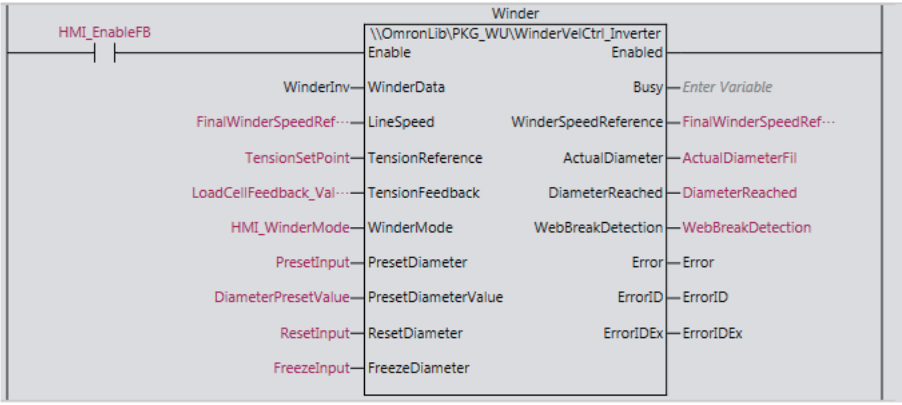
Read and Write RX PDO’s.

2

```
// WinderMode Setting
HMI_WinderMode := FALSE;

// Actual Motor Speed
ActualMotorSpeed := UINT_TO_LREAL(RX_OutputFreq) / 100;
ActualMotorSpeedPer:=(ActualMotorSpeed/WinderInv.BasicData
.MaxMotorSpeed)*100;
FinalWinderSpeedRefPer:=(FinalWinderSpeedReference/WinderInv
.BasicData.MaxMotorSpeed)*100 ;

// Actual Line Speed
ActualLineSpeedPer:=(ActualLineSpeed/WinderInv.BasicData
.MaxLineSpeed)*100 ;

// Encoder Line Speed conversion from mm/sec to meters/minute
EncoderLineSpeed := (ABS(EncoderLine.Act.Vel) * 60.0 ) / 1000.0;

//Load Cell Feedback value. Scaled from 0 to 100%.
LoadCellFeedback_Value := ((INT_TO_LREAL ( Analog_LoadCell -
MinLoadCellSensor_Value)*100.0) / INT_TO_LREAL
(MaxLoadCellSensor_Value - MinLoadCellSensor_Value));

// Diameter Sensor Value
ActualDiameterSensorValue := INT_TO_LREAL (Analog_DiameterSensor);
```

Calculate actual motor speed.

3 Winder FB with RX inverter


Winder
HMI_EnableFB
\\OmronLib\PKG_WU\WinderVelCtrl_Inverter
Enable
Enabled
WinderInv—WinderData
Busy—Enter Variable
FinalWinderSpeedRef···—LineSpeed
WinderSpeedReference—FinalWinderSpeedRef···
TensionSetPoint—TensionReference
ActualDiameter—ActualDiameterFil
LoadCellFeedback_Val···—TensionFeedback
DiameterReached—DiameterReached
HMI_WinderMode—WinderMode
WebBreakDetection—WebBreakDetection
PresetInput—PresetDiameter
Error—Error
DiameterPresetValue—PresetDiameterValue
ErrorID—ErrorID
ResetInput—ResetDiameter
ErrorIDEx—ErrorIDEx
FreezeInput—FreezeDiameter

WinderVelCtrl_Inverter execution.

Operation procedure

The operation procedure to execute WinderVelCtrl_Inverter

● From Power ON to Running

1.	Power on the system	Turn on the system.
2.	Enable ResetECError	In case of E-CAT error, reset the error.
3.	Enable ResetMCError	In case of Motion error, reset the error.
4.	Reset Inverter error	In case of Inverter error, reset the error.
5.	Configure winder application	Configure winder application. Configure setting items.
6.	Stop line speed	Confirm the Line speed is stopped.
7.	Set *Enable* TRUE	Set *Enable* to TRUE.
8.	Start Line Speed	Start Line Speed

● From Running (Winding/Unwinding) to Error stop

1.	An Error occurs	When an error occurs in WinderVelCtrl_Inverter, the winding/unwinding operation must be stopped (Error Stop).
2.	Reset the error.	Check cause of the error. After checking and confirming safety, reset errors. For troubleshooting, refer to the user’s manuals.

● From Running (Winding/Unwinding) to Emergency stop

1.	Emergency stop occurs	When an emergency stop or fatal error occurs in the machine, the winding operation stops immediately (Emergency Stop).
	▼	
2.	Reset the error.	Check cause of the emergency stop. After checking and confirming safety, reset the error. For troubleshooting, refer to the user’s manuals.

● From Emergency/Error stop to Restart

1.	Reset the error	Reset the error(s).
	▼	
2.	Stop line speed	Confirm the line speed is stopped before enabling the function block.
	▼	
3.	Set *Enable* TRUE	Set *Enable* to TRUE.

PackML Support

Function block name	Name	Page
PackMLModeStateMachine	PackML Mode/State Machine	P.217
PackMLModeStateTimer	Dwell Time Measure	P.230
PMLCtrlCmd_**	Transition Command Display	P.233
PMLState_Is**	State Output	P.237
PMLTransitionCmd_ResetAll	Transition Command All Reset	P.239
PMLTransitionCmd_ResetAllCmdSetAllSC	Transition Command Reset State Set	P.242
PMLTransitionCmd_Summarize	Transition Command Summarize	P.245
PMLTransitionCmd_SummarizePackTagCtrlCmd	Pack Tag Transition Command	P.248
Alarm	Alarm	P.251
Alarm2	Alarm 2	P.255
AlarmStatus_Update	EM Alarm Status Update	P.259
AlarmStatus_Update2	EM Alarm Status Update 2	P.262
AlarmSummation_Add	UN Alarm Status Add	P.266
AlarmSummation_Add2	UN Alarm Status Add 2	P.270
AlarmSummation_SortFilter	Alarm Sort and Filter	P.275
AlarmSummation_SortFilter2	Alarm Sort and Filter 2	P.279
DT_TO_PackTagDINTarray	DATE_AND_TIME Pack Tag Array Conversion	P.284

Background Information on PackML Support

Prior to explanation of individual specifications for PackML support FB, definition of PackML, mode and state of packaging machines specified by PackML are explained as background information.

Definition of PackML

PackML (Packaging Machine Language) is the standard for packaging machines which was standardized by OMAC (Organization for Machine Automation and Control). Mode and state of devices, and interface with peripheral devices are defined with the standard. FB and FUN which are supported by PackML conform to PackML V3.0. For information on PackML, refer to PackML specifications.

Mode and State of Packaging Machine

Packaging Machine has two types of concept, mode and state.

Mode is usage type of packaging machine. For PackML, three types of mode are defined; “PRODUCTION MODE,” “MAINTENANCE MODE,” and “MANUAL MODE.”

State is type of current state of packaging machine. Seventeen types of state are defined for PackML.

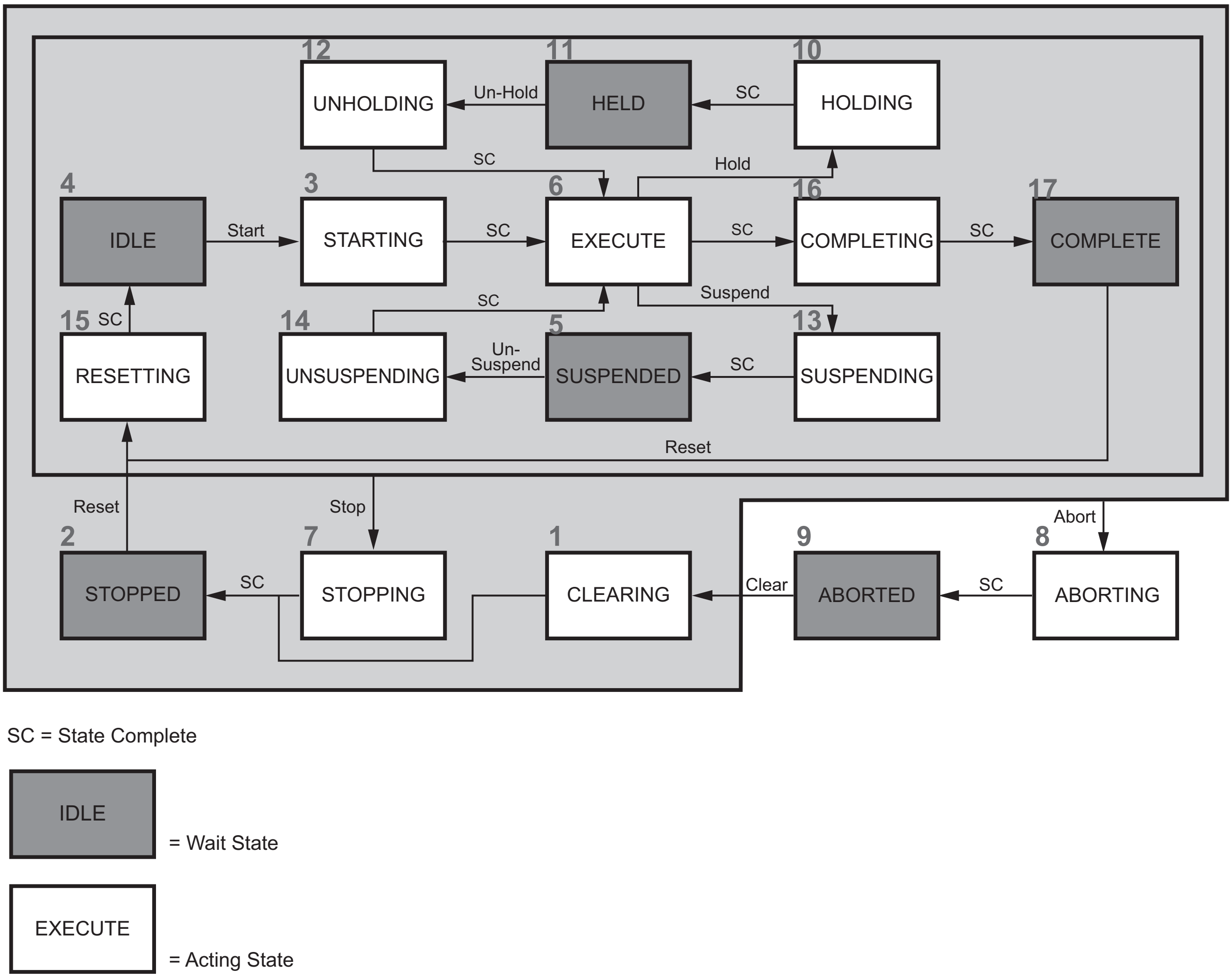
Mode

Packaging Machine has three types of mode; “PRODUCTION MODE,” “MAINTENANCE MODE,” and “MANUAL MODE.” Depending on the mode, operator of the machine and operation method changes. Individual modes of packaging machine are defined by ISA TR88.00.02 Rev.2 (2015) as follows.

- PRODUCTION MODE
  This represents the mode which is utilized for routine production. The machine executes relevant logic in response to commands which are either entered directly by the operator or issued by another supervisory system.
- MAINTENANCE MODE
  This mode allows, may allow suitably authorised personnel, the ability to run an individual machine independent of other machines in a production line. This mode would typically be used for faultfinding, machine trials or testing operational improvements. This mode would also allow the speed of the machine to be adjusted (where this feature is available).
- MANUAL MODE
  This provides direct control of individual machine axes. This feature is available depending upon the mechanical constraints of the mechanisms being exercised. This feature would be typically used for the commissioning of individual drives, verifying the operation of synchronised drives, testing the drive as a result of modifying parameters etc.

State

Packaging machine has seventeen types of status. Machine operation changes depending on the each state. Also, transition of state is defined depending on the operation which is executed in each state. The definition of the state and transition is common for each mode. Relation between each state and transition are shown as follows. For details of the state, refer to the specifications for PackML.

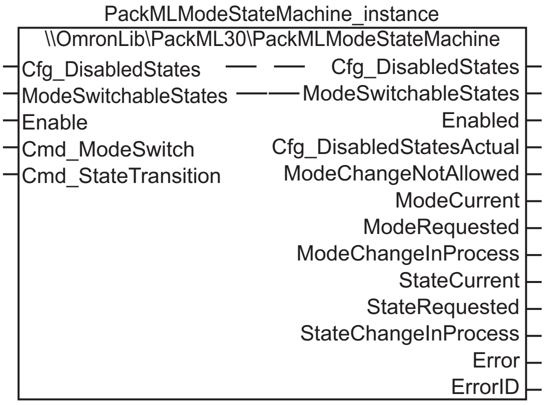


No.	State Name	State Type	Description
1	CLEARING	Acting	Transitioned to this state when AbortClear command is input in ABORTED state.
2	STOPPED	Wait	Transitioned to this state when StateComplete command is input in STOPPING sate. Example: machine stop state when power is supplied.
3	STARTING	Acting	Transitioned to this state when Start command is input in IDLE state.
4	IDLE	Wait	Transitioned to this state when StateComplete command is input in RESETTING state. Example: preparation for operation complete state
5	SUSPENDED	Acting	Transitioned to this state when StateComplete command is input in SUSPENDING state. Example: automatic stop state depending on the state at upstream process or downstream process.
6	EXECUTE	Dual State	Transitioned to this state by following methods. • Input StateComplete command in STATING state. • Input StateComplete command in UNHOLDING state. • Input StateComlete command in UNSUSPENDING state. Example: machine is in production state. Operation varies depending on the current mode.
7	STOPPING	Acting	Transitioned to this state when Stop command is input.
8	ABORTING	Acting	Transitioned to this state when Abort command is input.

No.	State Name	State Type	Description
9	ABORTED	Wait	Transitioned to this state when StateComplete command is input in ABORTING state. Example: emergency stop complete state (Power supply is turned OFF. Safe stop state after emergency stop.)
10	HOLDING	Acting	Transitioned to this state when Hold command is input in EXECUTE state.
11	HELD	Wait	Transitioned to this state when StateComplete command is input in HOLDING state. Example: pause of machine when trial operation and problem solving by operator
12	UNHOLDING	Acting	Transitioned to this state when UnHold command is input in HELD state.
13	SUSPENDING	Wait	Transitioned to this state when Suspend command is input in EXECUTE state.
14	UNSUSPENDING	Wait	Transitioned to this state when Unsuspend command is input in SUSPEND state.
15	RESETTING	Acting	Transitioned to this state when Reset command is input in COMPLETE state.
16	COMPLETING	Acting	Transitioned to this state when StateComplete command is input in EXECUTE state.
17	COMPLETE	Wait	Transitioned to this state when StateComplete command is input in COMPLETING state. Example: cycle stop state

PackMLModeStateMachine

Based on the mode/state machine stipulated by PackML, the function block outputs the current mode and state according to the mode change/state transition command.

Function block name	Name	FB/ FUN	Graphic expression	STexpression
PackML ModeState Machine	PackML Mode/ State Machine	FB	 PackMLModeStateMachine_instance \\OmronLib\PackML30\PackMLModeStateMachine Cfg_DisabledStates ModeSwitchableStates Enable Cmd_ModeSwitch Cmd_StateTransition Cfg_DisabledStates ModeSwitchableStates Enabled Cfg_DisabledStatesActual ModeChangeNotAllowed ModeCurrent ModeRequested ModeChangeInProcess StateCurrent StateRequested StateChangeInProcess Error ErrorID	PackMLModeStateMachine _instance(Cfg_DisabledStates, ModeSwitchableStates, Enable, Cmd_ModeSwitch, Cmd_StateTransition, Enabled, Cfg_DisabledStatesActual, ModeChangeNotAllowed, ModeCurrent, ModeRequested, ModeChangeInProcess, StateCurrent, StateRequested, StateChangeInProcess, Error, ErrorID);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_Vx_x.slr[*1]
Namespace	OmronLib\PackML30
Function block and function number	00101
Publish/Do not publish source code	Not published.
Function block and function version	1.00

*1. Vx_x indicates version.

Mode/State Machine Settings

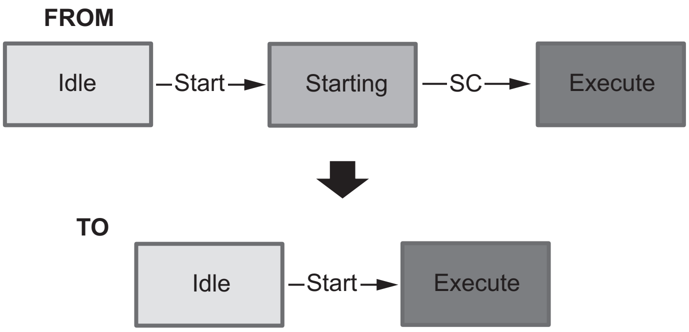
Mode/State Machine Settings

- Configure the PackML mode/State machine based on the specifications of the packaging machine application you create. Specify it by using the In/Out variables as follows:
  Cfg_ModeStateConfiguration: ARRAY[1..31] OF
  OmronLib\PackML30\sPACKML_MODE_STATE_CONFIG
  The above array index represents the mode number.
  a) Per mode, set the unused states.
  b) Per mode, set the states in which the mode is permitted to be switched.
- The In/Out variable, Cfg_ModeStateConfiguration, of setting the mode/state machine is evaluated and set only in the rising edge of *Enable* Input. Afterwards, the mode/state machine settings are retained regardless of the In/Out variable value while the cyclic execution keeps ongoing. To change the settings, you need to reset the *Enable* Input of the function block to FALSE, and then execute it again after setting TRUE for *Enable*.
- When the specified settings are not adequate, the settings are modified to be adequate inside the function block, and then the modified settings are updated into the mode/state machine. The modification result is written back to the In/Out variable, Cfg_ModeStateConfiguration.

Unused State Settings

To configure the state to be used per mode, set TRUE for the BOOL member of the unused state by using *Cmd_StateTransition*[modeNumber]. StatesDisabled.

- Because the following states are essential for PackML v3.0, you cannot set them the unused state. The In/Out variables are modified automatically.
  a) Stopped
  b) Idle
  c) Execute
  d) Aborted
- When a state is not used, the command that transitions a state to the unused state will transition the state to the next state unconditionally. The following example shows the state transition when the Starting state is not used.



- When the Wait states except for Stopped, Idle, Execute, and Aborted are set as the unused states, the related Acting (…ing) states are automatically set as the unused states accordingly.

State set as “DisabelState”	State as “unused State” automatically
Resetting	-
Starting	-
Suspending	-
Unsuspending	-
Holding	-
UnHolding	-
Completing	-
Aborting	-
Cleaning	-
Idle	-
Held	Holding,UnHolding
Suspend	Suspending,Unsuspending
Complete	Completing
Stopped	-
Aborted	-
Execute	-

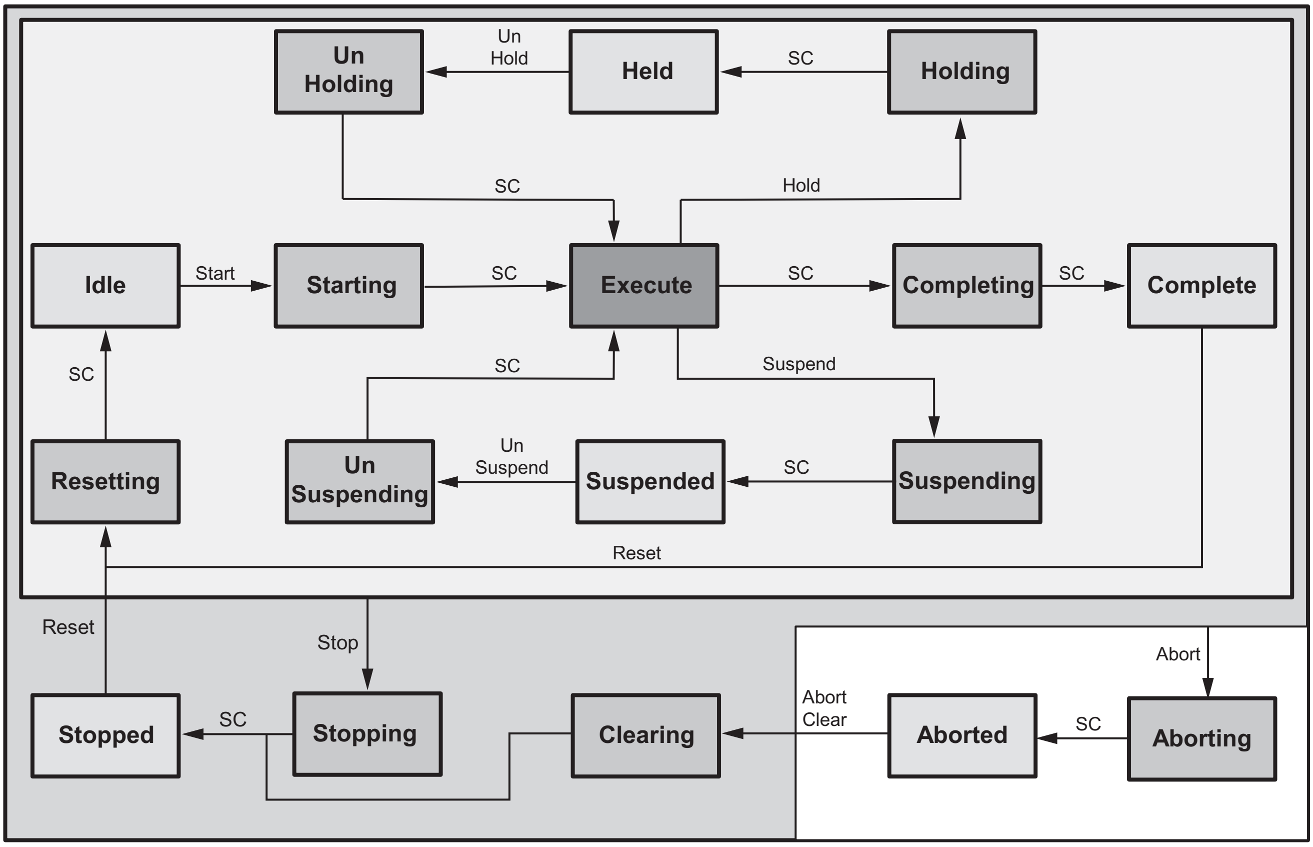
Here describes the typical modes such as “Production Mode”, “Manual Mode”, “Maintenance Mode” defined by Pack_ML.

● Example of Mode Settings

Production Mode (Mode Number: 1)

Production Mode is for production that is repeated by a machine. In this mode, the command that is directly entered by an operator or output by other monitoring-system starts up a machine.
(e.g., Running/Normal Operating)

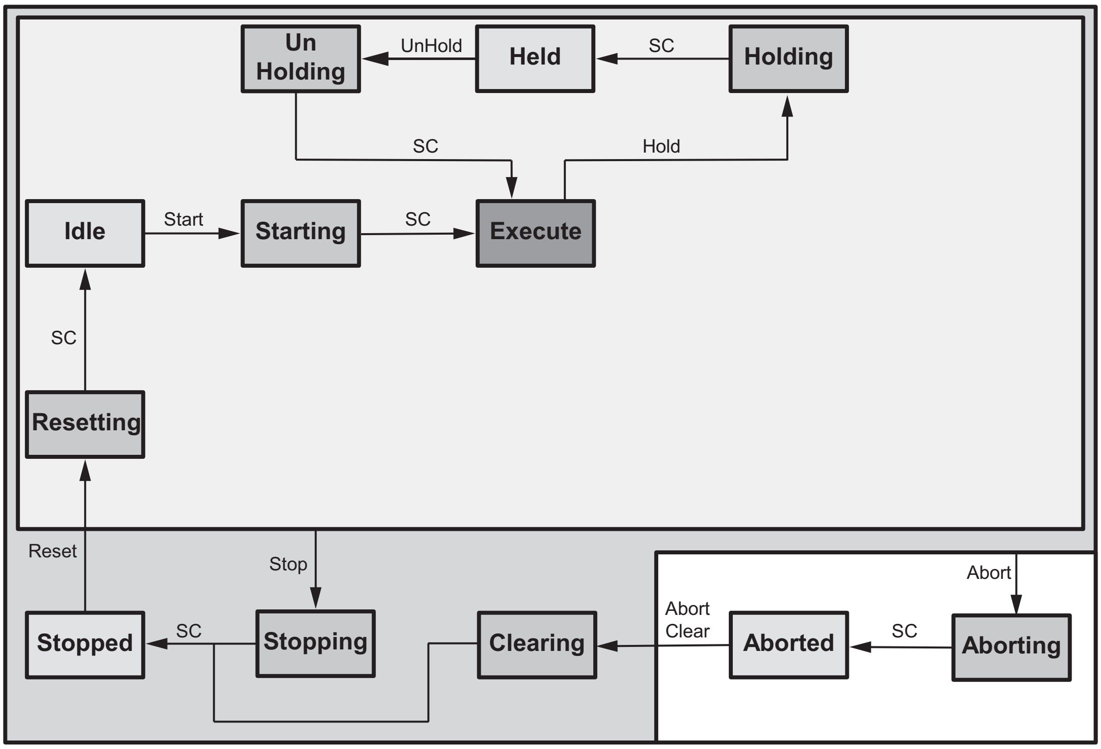
The following diagram shows the state transitions in the Production mode.



	State Commands								
Current State	Start	Reset	Hold	UnHold	Sus-pend	UnSuspend	Abort-Clear	Stop	Abort	State Complete
Idle	Staring	-	-	-	-	-	-	Stopping	Aborting	-
Starting	-	-	-	-	-	-	-	Stopping	Aborting	Execute
Execute	-	-	Holding	-	Sus-pending	-	-	Stopping	Aborting	Completing
Completing	-	-	-	-	-	-	-	Stopping	Aborting	Complete
Complete	-	Resetting	-	-	-	-	-	Stopping	Aborting	-
Resetting	-	-	-	-	-	-	-	Stopping	Aborting	Idle
Holding	-	-	-	-	-	-	-	Stopping	Aborting	Held
Held	-	-	-	UnHold-ing	-	-	-	Stopping	Aborting	-
UnHolding	-	-	-	-	-	-	-	Stopping	Aborting	Execute
Suspending	-	-	-	-	-	-	-	Stopping	Aborting	Suspended
Suspended	-	-	-	-	-	UnSuspend-ing	-	Stopping	Aborting	-
UnSuspend-ing	-	-	-	-	-	-	-	Stopping	Aborting	Execute
Stopping	-	-	-	-	-	-	-	-	Aborting	Stopped
Stopped	-	Resetting	-	-	-	-	-	-	Aborting	-
Aborting	-	-	-	-	-	-	-	-	-	Aborted
Aborted	-	-	-	-	-	-	Clearing	-	-	-
Clearing	-	-	-	-	-	-	-	-	Aborting	Stopped

*“-” should be ignored.

Maintenance Mode



In this mode, only permitted persons are allowed to operate individual machines in the production line. This mode is effective to detect machine defects, make a trial run, and test a machine.

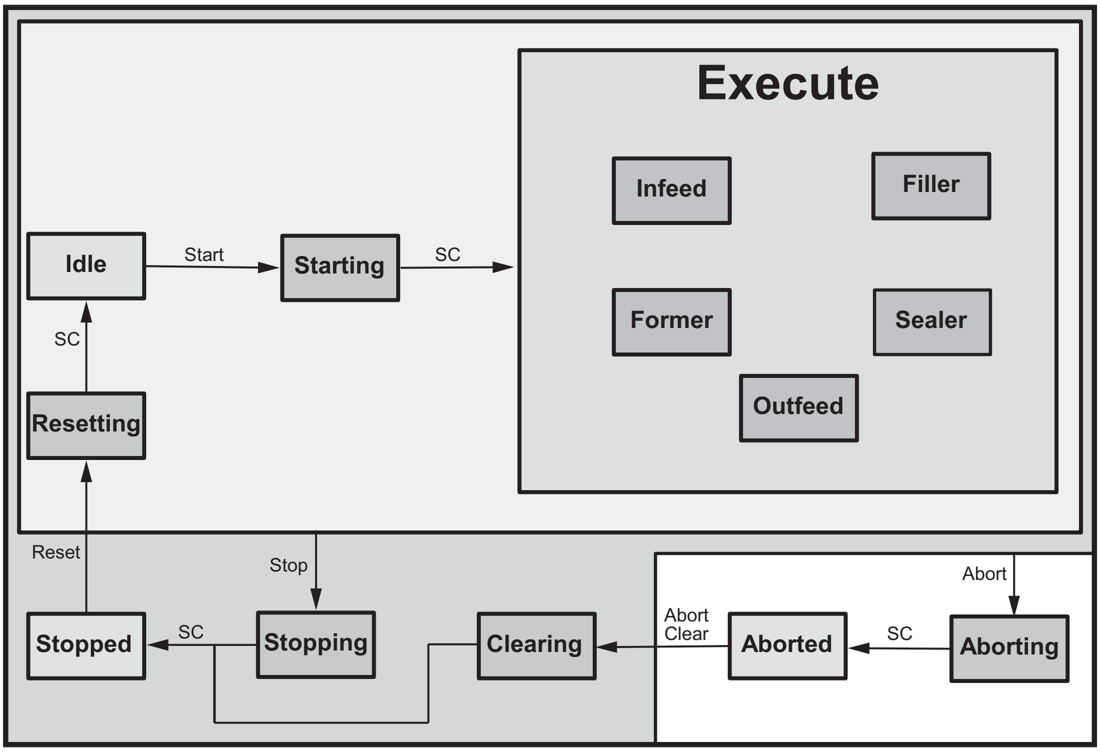
The following table shows the state transitions in the Maintenance Mode.

	State Commands									
Current State	Start	Reset	Hold	UnHold	Sus-pend	UnSuspend	Abort-Clear	Stop	Abort	State Complete
Idle	Staring	-	-	-	-	-	-	Stopping	Aborting	-
Starting	-	-	-	-	-	-	-	Stopping	Aborting	Execute
Execute	-	-	Holding	-	-	-	-	Stopping	Aborting	Completing
Completing	-	-	-	-	-	-	-	-	-	-
Complete	-	-	-	-	-	-	-	-	-	-
Resetting	-	-	-	-	-	-	-	Stopping	Aborting	Idle
Holding	-	-	-	-	-	-	-	Stopping	Aborting	Held
Held	-	-	-	UnHolding	-	-	-	Stopping	Aborting	
UnHolding	-	-	-	-	-	-	-	Stopping	Aborting	Execute
Suspending	-	-	-	-	-	-	-	-	-	-
Suspended	-	-	-	-	-	-	-	-	-	-
UnSuspending	-	-	-	-	-	-	-	-	-	-
Stopping	-	-	-	-	-	-	-	-	Aborting	Stopped
Stopped	-	Resetting	-	-	-	-	-	-	Aborting	-
Aborting	-	-	-	-	-	-	-	-	-	Aborted
Aborted	-	-	-	-	-	-	Clearing	-	-	-
Clearing	-	-	-	-	-	-	-	-	Aborting	Stopped

*“-” should be ignored.

In this mode, only permitted persons are allowed to operate a machine directly (e.g., jogging).

The following diagram describes the state transition in the Manual Mode.



*The states in *Execute* are defined by the user. They are not processed by this function block.

	State Commands									
Current State	Start	Reset	Hold	UnHold	Sus-pend	UnSuspend	Abort-Clear	Stop	Abort	State Complete
Idle	Staring	-	-	-	-	-	-	Stopping	Aborting	-
Starting	-	-	-	-	-	-	-	Stopping	Aborting	Execute
Execute	-	-	-	-	-	-	-	Stopping	Aborting	-
Completing	-	-	-	-	-	-	-	-	-	-
Complete	-	-	-	-	-	-	-	-	-	-
Resetting	-	-	-	-	-	-	-	Stopping	Aborting	Idle
Holding	-	-	-	-	-	-	-	-	-	-
Held	-	-	-	-	-	-	-	-	-	-
UnHolding	-	-	-	-	-	-	-	-	-	-
Suspending	-	-	-	-	-	-	-	-	-	-
Suspended	-	-	-	-	-	-	-	-	-	-
UnSuspending	-	-	-	-	-	-	-	-	-	-
Stopping	-	-	-	-	-	-	-	-	Aborting	Stopped
Stopped	-	Resetting	-	-	-	-	-	-	Aborting	-
Aborting	-	-	-	-	-	-	-	-	-	Aborted
Aborted	-	-	-	-	-	-	Clearing	-	-	-
Clearing	-	-	-	-	-	-	-	-	Aborting	Stopped

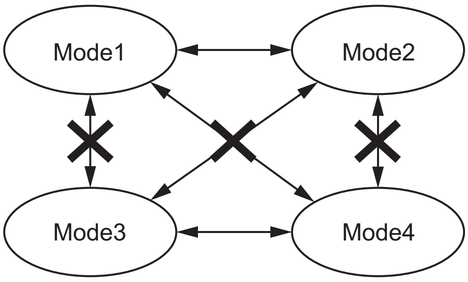
*“-” should be ignored.

Permit Mode Switch Setting

To specify the permit mode switch, set TRUE for the BOOL member of the state, which is permitted to switch, by using *Cmd_StateTransition*[modeNumber].StatesModeSwitchable

- When this flag is in the TRUE state in the current mode, the mode is switchable to the mode which flag is TRUE.
- With this function block, the mode is switchable to all of the modes in which the permit mode switch flag is set. A function that allows a mode to be switched between specific modes is not supported.

Add required interlock logic outside of this function block if you need an application that allows a mode to be switched between specific modes in a specific state, and to control a mode to be switched between the specific modes as the following diagram shows.



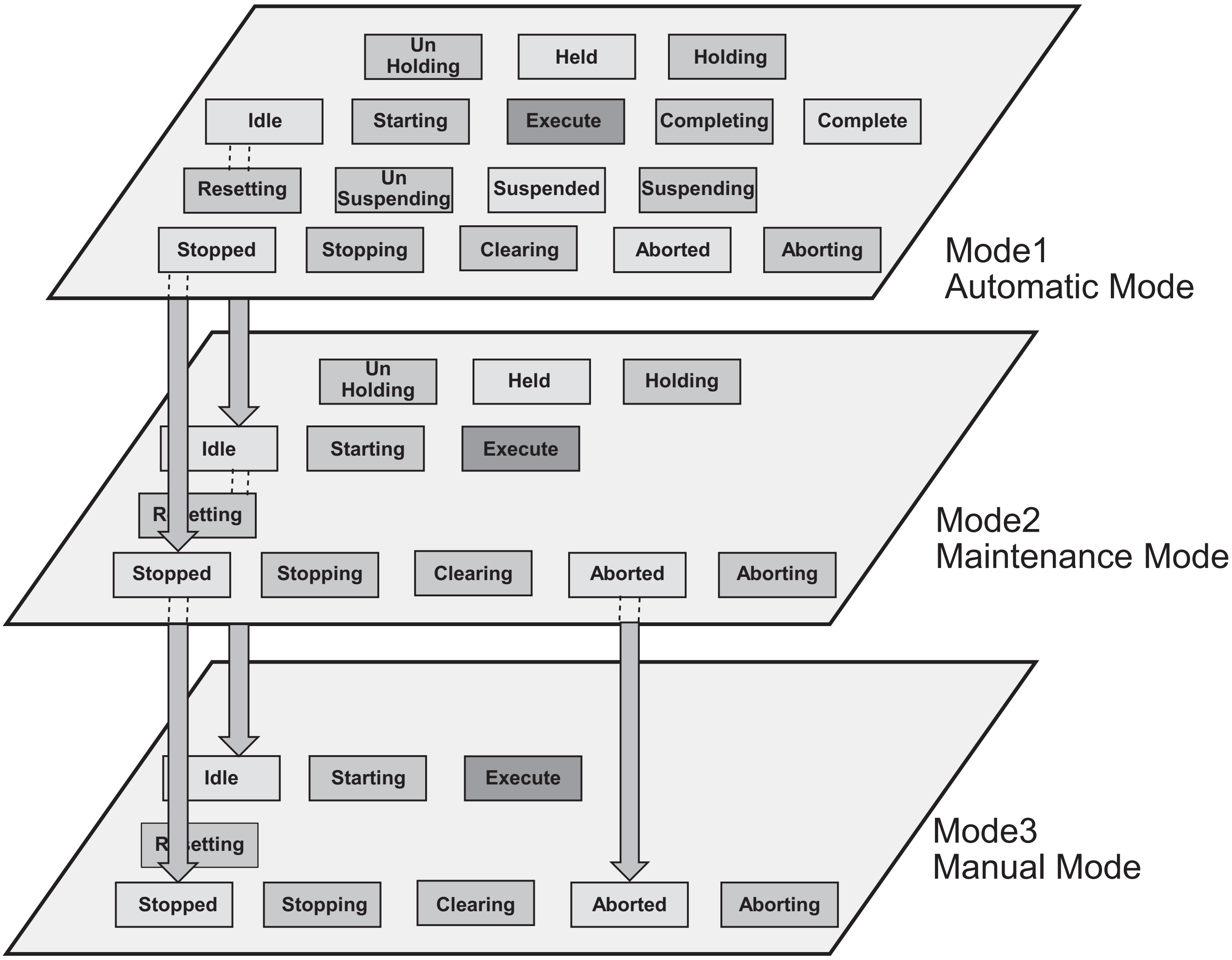
There are two ways of interlocking.

(a) Based on the current mode, change dynamically the settings of the In/Out variable, *ModeSwitchableStates*.

(b) Based on a specific mode, limit the values to be set for the In/Out variable, *Cmd_StateTransition*.

The following diagram shows an example of mode switch settings.

Mode1 to 3 support the foregoing “*Production*”, “*Maintenance*”, and “*Manual*” modes respectively. The states in which the mode is switched between Mode 1 and 2 are set for “*Idle*” and “*Stopped*”. The states in which the mode is switched between Mode2 and 3 are set for “*Idle*”, “*Stopped*”, and “*Aborted*”.



ModeSwitchableStates, which sets the above, is set as follows:

(The initial value of each member of sPACKML_STATES_FLAG should be FALSE.)

```
ModeSwitchableStates[1].StoppedState := TRUE;
ModeSwitchableStates[2].StoppedState := TRUE;
ModeSwitchableStates[3].StoppedState := TRUE;

ModeSwitchableStates[1].IdleState := TRUE;
ModeSwitchableStates[2].IdleState := TRUE;
ModeSwitchableStates[3].IdleState := TRUE;

ModeSwitchableStates[2].AbortedState := TRUE;
ModeSwitchableStates[3].AbortedState := TRUE;
```

State Transition

- In the state transition command, *Cmd_StateTransition*, more than one transaction request flags can be set to TRUE. However, transition requests which is not executable for the current state is ignored. Even if no executable transaction request is included, it is simply ignored without any error.
- When the state transition command, *Cmd_StateTransition*, includes more than one executable transaction request, only one state transition is executed in the order of Cmd_Abort, Cmd_Stop, Cmd_xxx, and Sts_xxx_SC.
- When a valid mode switch command and a valid state transaction command are entered at the same time, the state transition command is executed first.

Mode Switch Function

- When the value of the mode switch command, *Cmd_ModeSwitch*, is not the valid value (1 to 31), it is considered as a request of switching the mode.
- When 0 or a mode number out of the range (32 or larger) is specified, the number is ignored, but it is not an error.
- When the mode switch command and a valid state transition command are given at the same time, the state transition is executed first.

Variables

Name	In/Out	Data Type	Default	Description
Enable	Input	BOOL	FALSE	FB-enabled flag. Enables the function block function. Executing this in FALSE will clear the settings entered in the following.
Cfg_DisabledStates	In/Out	ARRAY[1..31] OF OmronLib\PackML30\sPACKML_STATES_-FLAG	-	Disabled-state settings. The unused states are specified per mode. The array index represents the mode number. When the set value is not adequate, the value is overwritten and corrected automatically.
ModeSwitchableStates	In/Out	ARRAY[1..31] OF OmronLib\PackML30\sPACKML_STATES_-FLAG	-	Mode Switch Permit Settings. The states where switching mode is permitted are specified per mode. The array index represents the mode number.
Cmd_ModeSwitch	Input	DINT	0	Mode Switch Command. The mode number is specified in order to change the mode. Nothing is executed when 0, the same number as the current mode, or the number out of range is specified.
Cmd_StateTransition	Input	OmronLib\PackML30\sPACKML_TRANSITION_COMMAND	FALSE (All member)	Transition Command. The transition request to the state machine is specified. More than one transaction is specifiable. However, only an executable transaction for the current state is executed. When more than one transaction is executable, one transaction is executed according to the existing priority.
Enabled	Output	BOOL	-	FB-enabled Flag Output. When TRUE, the effective values for the output variable of this function block are output.
Cfg_DisabledStatesActual	Output	ARRAY[1..31] OF OmronLib\PackML30\sPACKML_STATES_-FLAG	-	Non-use StateSetting Output. Indicate the Non-use State setting in FB. TRUE is output in the Non-use state bits.
ModeChangeNotAllowed	Output	BOOL	-	Mode Change Prohibited Flag. The flag becomes TRUE when the condition to switch the mode is not met although the valid-mode-change command between 1 and 31 is entered.
ModeCurrent	Output	DINT	-	The current mode number is output.

Name	In/Out	Data Type	Default	Description
ModeRequested	Output	DINT	-	The mode number that is currently requested to switch is output.
ModeChangeInProcess	Output	BOOL	-	It becomes TRUE only for 250ms after the mode is switched.
StateCurrent	Output	DINT		The current state number is output.
StateRequested	Output	DINT	-	The state number that is currently requested to execute the state transition is output.
StateChangeInProceess	Output	BOOL	-	It becomes TRUE only for 250ms after the state transition.
Error	Output	BOOL	-	Error flag. It is always FALSE because internal error never occurs in this function block.
ErrorID	Output	DWORD	-	Error ID It is always 0 (normal) because internal error never occurs in this function block.

Structures

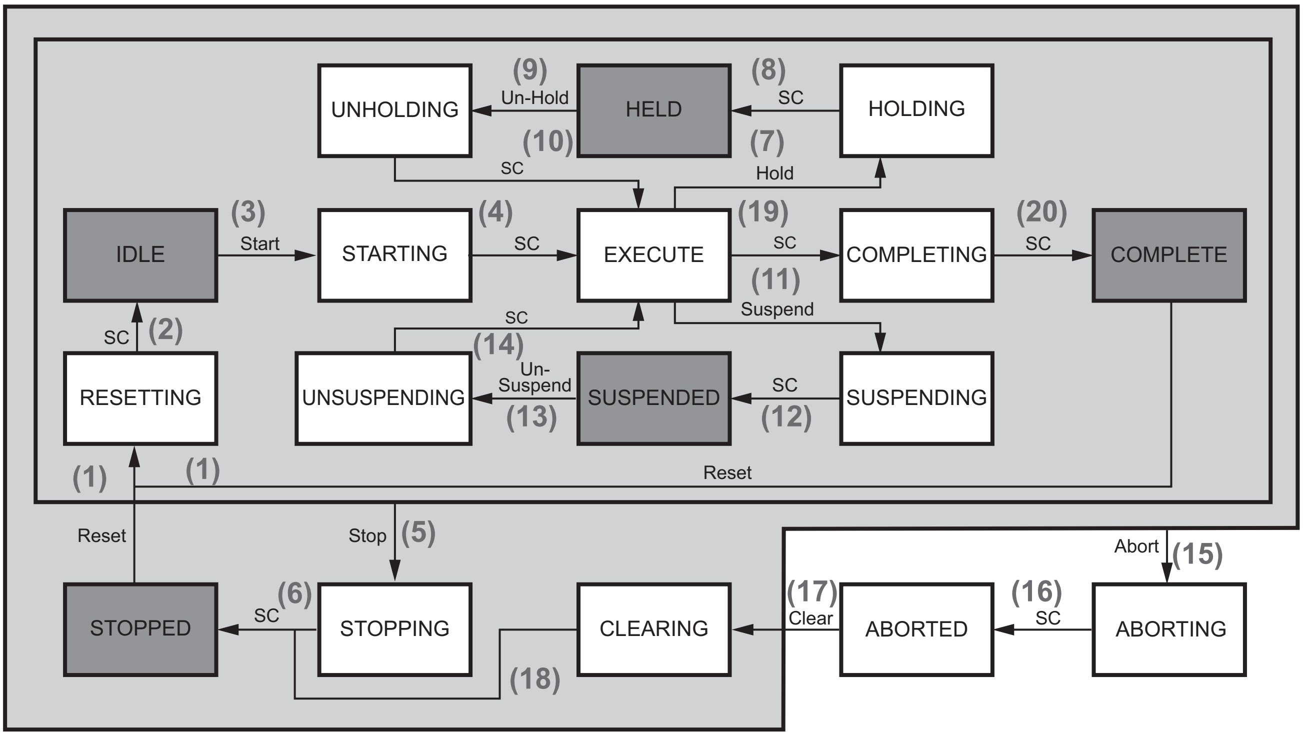
● sPACKML_STATES_FLAG

BOOL type variable group that shows the state set by PackML.

Name	Data Type	Description
sPACKML_STATES_FLAG	STRUCT	The structure that retains a setting flag per state.
CleaningState	BOOL	
StoppedState	BOOL	
StartingState	BOOL	
IdleState	BOOL	
SuspendedState	BOOL	
ExecuteState	BOOL	
StoppingState	BOOL	
AbortingState	BOOL	
AbortedState	BOOL	
HoldingState	BOOL	
HeldState	BOOL	
UnholdingState	BOOL	
SuspendingState	BOOL	
UnssupendingState	BOOL	
ResettingState	BOOL	
CompletingState	BOOL	
CompleteState	BOOL	

● sPACKML_TRANSITION_COMMAND

Name	Data Type	Description
sPACKML_TRANSITION_COMMAND	STRUCT	The structure that indicates a transition for the PackML state machine.
Cmd_Reset	BOOL	The command to execute the state transition from *Stopped* or *Complete* to *Resetting*.(1)
Sts_Resetting_SC	BOOL	The request to execute the state transition from *Resetting* to *Idle*. (2)
Cmd_Start	BOOL	The command to execute the state transition from *Idle* to *Starting*. (3)
Sts_Starting_SC	BOOL	The request to execute the state transition from *Starting* to *Execute*. (4)
Cmd_Stop	BOOL	The command to execute the state transition from *Idle*, *Resetting*, *Starting*, *Execute*, *Completing*, *Complete*, *Holding*, *Held*, *Unholding*, *Suspending*, *Suspended*, or *Unssupending* to *Stopping*. (5)
Sts_Stopping_SC	BOOL	The request to execute the state transition from *Stopping* to *Stopped*. (6)
Cmd_Hold	BOOL	The command to execute the state transition from *Execute* to *Holding*. (7)
Sts_Holding_SC	BOOL	The request to execute the state transition from *Holding* to *Held*. (8)
Cmd_UnHold	BOOL	The command to execute the state transition from *Held* to *UnHolding*. (9)
Sts_UnHolding_SC	BOOL	The request to execute the state transition from *UnHolding* to *Execute*. (10)
Cmd_Suspend	BOOL	The command to execute the state transition to *Execute* to *Suspending*. (11)
Sts_Suspending_SC	BOOL	The request to execute the state transition from *Suspending* to *Suspended*. (12)
Cmd_UnSuspend	BOOL	The command to execute the state transition from *Suspended* to *UnSuspending*. (13)
Sts_UnSuspending_SC	BOOL	The request to execute the state transition from *UnSuspending* to *Execute*. (14)
Cmd_Abort	BOOL	The command to execute the state transition from the state except Aborting and *Aborted* state, to *Aborting*. (15)
Sts_Aborting_SC	BOOL	The request to execute the state transition from *Aborting* to *Aborted*. (16)
Cmd_Clear	BOOL	The command to execute the state transition from *Aborted* to *Clearing*. (17)
Sts_Clearing_SC	BOOL	The request to execute the state transition from *Clearing* to *Stopped*. (18)
Sts_Execute_SC	BOOL	The command to execute the state transition from *Execute* to *Completing*. (19)
Sts_Completing_SC	BOOL	The request to execute the state transition from *Completing* to *Complete*. (20)



Function

Based on the mode/state machine stipulated by PackML, the function block outputs the current mode and state according to the mode change/state transition command.

- Accepts unused state setting per mode specified.
- Sets the state in which the mode is switched.
- Switches the mode to the specified one if the internal state allows.
- Executes the specified state transition if the current state allows it. When more than one state transition commands are received at the same time, the state transitions are executed according to the proper priority.

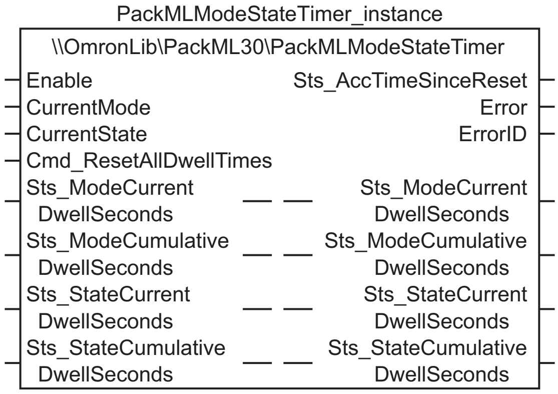
Troubleshooting

This function block does not output error.

Even if invalid values are given to the input variables with *Enable* input TRUE, this function block ignores such inputs, always outputs the valid values and always set *Enabled* output TRUE.

PackMLModeStateTimer

The function block measures and outputs a dwell time (second) in each state and mode of the mode/state machine stipulated by PackML.

Function block name	Name	FB/ FUN	Graphic expression	STexpression
PackML ModeState Timer	Dwell Time Measure	FB	 PackMLModeStateTimer_instance \\OmronLib\PackML30\PackMLModeStateTimer Enable CurrentMode CurrentState Cmd_ResetAllDwellTimes Sts_ModeCurrent DwellSeconds Sts_ModeCumulative DwellSeconds Sts_StateCurrent DwellSeconds Sts_StateCumulative DwellSeconds Sts_AccTimeSinceReset Error ErrorID Sts_ModeCurrent DwellSeconds Sts_ModeCumulative DwellSeconds Sts_StateCurrent DwellSeconds Sts_StateCumulative DwellSeconds	PackMLModeStateTimer _instance(Enable, CurrentMode, CurrentState, Cmd_ResetAllDwellTimes, Sts_ModeCurrentDwellSeconds, Sts_ModeCumulativeDwellSeconds, Sts_StateCurrentDwellSeconds, Sts_StateCumulativeDwellSeconds, Sts_AccTimeSinceReset, Error, ErrorID);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_Vx_x.slr[*1]
Namespace	OmronLib\PackML30
Function block and function number	00102
Publish/Do not publish source code	Not published.
Function block and function version	1.00

*1. Vx_x indicates version.

Variables

Name	In/Out	Data Type	Default	Description
Enable	Input	BOOL	FALSE	FB-enabled flag. Enables this function block.
CurrentMode	Input	DINT	1	Specifies the current mode number.
CurrentState	Input	DINT	1	Specifies the current state number.
Cmd_ResetAllDwell-Times	Input	BOOL	FALSE	The command to reset the accumulated time. When it is executed at TRUE, accumulated dwell seconds in each mode and state are reset to 0.
Sts_ModeCurrent-DwellSeconds	In/Out	ARRAY[1..31] OF DINT	-	Outputs dwell seconds in the current mode. Outputs seconds that have elapsed after the mode was switched to the current mode. Actually it is output but the variable is the In/Out variable in consideration of performance at the time of execution. For the array that has the mode number in the index, the values of the modes except for the current mode are 0.
Sts_ModeCumula-tive DwellSeconds	In/Out	ARRAY[0..31] OF DINT	-	Outputs accumulated dwell seconds in each mode. Outputs accumulated dwell seconds in the current mode after the last reset. Actually it is output but the variable is the In/Out variable in consideration of performance at the time of execution. The array that has the mode number in the index.
Sts_StateCurrent-DwellSeconds	In/Out	ARRAY [1..31,1..17] OF DINT	-	Outputs accumulated dwell seconds in the current state. Outputs seconds that have elapsed after the mode was transitioned to the current mode. Actually it is output but the variable is the In/Out variable in consideration of performance at the time of execution. For the array that has the mode number in the index, the values of the modes except for the current state are 0.
Sts_StateCumulative DwellSeconds	In/Out	ARRAY [1..31,1..17] OF DINT	-	Outputs the accumulated dwell seconds in the current state. Outputs the seconds that have elapsed after the state was transitioned to the current state. Actually it is output but the variable is the I/O variable in consideration of performance at the time of execution. For the array that has the state number in the index, the values of the modes except for the current state are 0.
Sts_AccTimeSince Reset	Output	DINT	-	The seconds that have elapsed after the last reset.
TimeRollOver Warning	Output	BOOL	-	When the seconds that have elapsed after the last reset exceed 2,147,483,647 seconds, it becomes TRUE. When the value is TRUE, no valid value is output to *Sts_AccTimeSinceReset*.
Error	Output	BOOL	-	Error flag. Refer to *Troubleshooting* on page 232

Name	In/Out	Data Type	Default	Description
ErrorID	Output	WORD	-	Error ID Refer to *Troubleshooting* on page 232

Function

The function block measures and outputs a dwell time (second) in each state and mode of the mode/state machine stipulated by PackML.

The function is mainly used for calculating the following tags of PackTag.

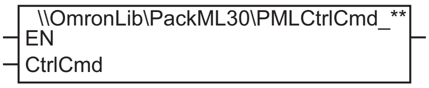
- Admin.ModeCurrentTime
- Admin.StateCurrentTime
- Admin.AccTimeSinceReset
- Admin.ModeCumulativeTime
- Admin.StateumlativeTime

Troubleshooting

If a value out of the range is input to the input variable, *CurrentMode* and *CurrentState*, this FB does not execute processing and output an error.

PMLCtrlCmd_**

The functions will check which transition command is the number of Command.CntrlCmd stipulated by PackTag.

Function name	Name	FB/ FUN	Graphic expression	ST expression
PLMCtrlCmd _**	Transition Command Display	FUN	 "**" must be a transition command.	\\OmronLib\PackML30\PMLCtrlCmd_** (CtrlCmd); "**" must be a transition command.

Function Block and Function Information

Item	Function name	Description
Library file name	-	OmronLib_PackML30_Vx_x.slr[*1]
Namespace	-	OmronLib\PackML30
Function block and function number	PMLCtlCmd_Reset	00103
	PMLCtlCmd_Start	00104
	PMLCtlCmd_Stop	00105
	PMLCtlCmd_Hold	00106
	PMLCtlCmd_Unhold	00107
	PMLCtlCmd_Suspend	00108
	PMLCtlCmd_Unsuspend	00109
	PMLCtlCmd_Abort	00110
	PMLCtlCmd_Clear	00111
Publish/Do not publish source code	-	Not published.
Function block and function version	-	1.00

[*1] Vx_x indicates version.

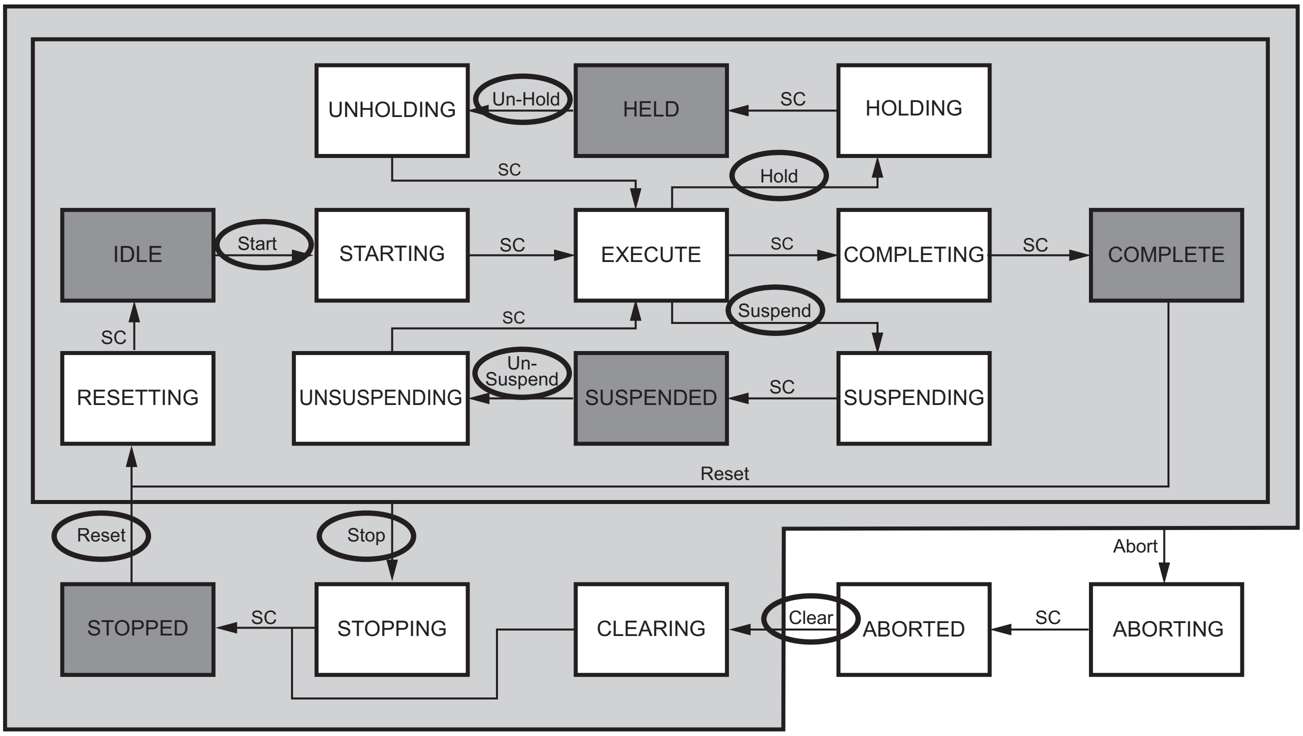
Variables

Name	In/Out	Data Type	Default	Description
EN	Input	BOOL	TRUE	Function Execution Control Flag. At FALSE, internal logic is not executed even if it is called.
CtrlCmd	Input	DINT	0	Transition Command Number. Specifies the value obtained from the Command.Cntrl-Cmd tag of PackTag. The range of input value is from 1 to 9.
<Function name>	Return value	BOOL	-	Return value. TRUE is returned only when the entered transition number represents the function name. (FALSE is returned when a value out of the range is entered.)

Function

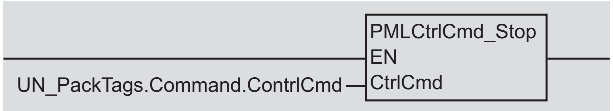
The functions check which transition command is the number of Command.CntrlCmd stipulated by PackTag. With these functions, the user no longer needs to see the specification to find out which transition number actually represents which transition.

The following diagram shows the transitions and their numbers specified by Command.CntrlCmd.



0	Undefiend
1	Reset
2	Start
3	Stop
4	Hold
5	Unhold
6	Suspend
7	Unsuspend
8	Abort
9	Clear

The example of how to see which specified transition command is the Stop command.

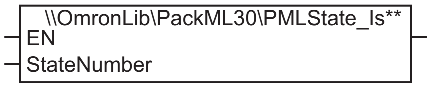


Function List

Name	Description
PMLCtrlCmd_Reset	TRUE is returned when the entered transition number is 1.
PMLCtrlCmd_Start	TRUE is returned when the entered transition number is 2.
PMLCtrlCmd_Stop	TRUE is returned when the entered transition number is 3.
PMLCtrlCmd_Hold	TRUE is returned when the entered transition number is 4.
PMLCtrlCmd_Unhold	TRUE is returned when the entered transition number is 5.
PMLCtrlCmd_Suspend	TRUE is returned when the entered transition number is 6.
PMLCtrlCmd_Unsuspend	TRUE is returned when the entered transition number is 7.
PMLCtrlCmd_Abort	TRUE is returned when the entered transition number is 8.
PMLCtrlCmd_Clear	TRUE is returned when the entered transition number is 9.

PMLState_Is**

The functions will check which state number stipulated by PackML represents which state.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
PMLState_Is**	State Output	FUN	 "**" must be a state.	\\OmronLib\PackML30 \PMLState_Is** (StateNumber); "**" must be a state.

Function Block and Function Information

Item	Function name	Description
Library file name	-	OmronLib_PackML30_Vx_x.slr[*1]
Namespace	-	OmronLib\PackML30
Function block and function number	PMLState_IsClearing	00112
	PMLState_IsStopped	00113
	PMLState_IsStarting	00114
	PMLState_IsIdle	00115
	PMLState_IsSuspended	00116
	PMLState_IsExecute	00117
	PMLState_IsStopping	00118
	PMLState_IsAborting	00119
	PMLState_IsAborted	00120
	PMLState_IsHolding	00121
	PMLState_IsHeld	00122
	PMLState_IsUnholding	00123
	PMLState_IsSuspending	00124
	PMLState_IsUnsuspending	00125
	PMLState_IsResetting	00126
	PMLState_IsCompleting	00127
	PMLState_IsComplete	00128
Publish/Do not publish source code	-	Not published.
Function block and function version	-	1.00

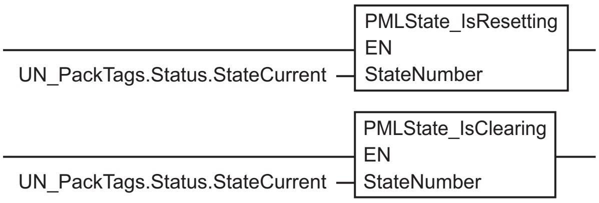
*1. Vx_x indicates version.

Variables

Name	In/Out	Data Type	Default	Description
EN	Input	BOOL	TRUE	Function Control Flag. At FALSE, internal logic does not operate even if it is called.
StateNumber	Input	DINT	0	State number. Specifies the state number to check. The range of input value is from 1 to 17.
<the same with the function name>	Return value	BOOL	-	Return value. TRUE is returned only when the entered state number represents the function name. (FALSE is returned when a state number out of the range is entered.)

Function

The functions check which state number stipulated by PackML represents which state. With these functions, the user no longer needs to see the specification to find out which state number, which is output by the PackML mode/state control function block, represents which state.



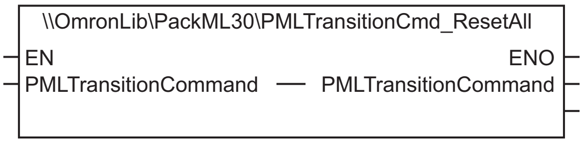
Function List

Name	Description
PMLState_IsClearing	TRUE is returned when the entered state number is 1.
PMLState_IsStopped	TRUE is returned when the entered state number is 2.
PMLState_IsStarting	TRUE is returned when the entered state number is 3.
PMLState_IsIdle	TRUE is returned when the entered state number is 4.
PMLState_IsSuspended	TRUE is returned when the entered state number is 5.
PMLState_IsExecute	TRUE is returned when the entered state number is 6.
PMLState_IsStopping	TRUE is returned when the entered state number is 7.
PMLState_IsAborting	TRUE is returned when the entered state number is 8.
PMLState_IsAborted	TRUE is returned when the entered state number is 9.
PMLState_IsHolding	TRUE is returned when the entered state number is 10.
PMLState_IsHeld	TRUE is returned when the entered state number is 11.
PMLState_IsUnholding	TRUE is returned when the entered state number is 12.
PMLState_IsSuspending	TRUE is returned when the entered state number is 13.
PMLState_IsUnsuspending	TRUE is returned when the entered state number is 14.
PMLState_IsResetting	TRUE is returned when the entered state number is 15.
PMLState_IsCompleting	TRUE is returned when the entered state number is 16.
PMLState_IsComplete	TRUE is returned when the entered state number is 17.

PMLTransitionCmd_ResetAll

For the state transition command sPACKML_TRANSITION_COMMAND structure-type variables, this function resets every BOOL member that indicates the state transition to FALSE.

This function is used for initializing the state transition request to the host module.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
PML Transition Cmd_ ResetAll	Transi- tion Com- mand All Reset	FUN	 \\OmronLib\PackML30\PMLTransitionCmd_ResetAll EN ENO PMLTransitionCommand PMLTransitionCommand	\\OmronLib\PackML30\PMLTransitionCmd_ResetAll (PMLTransitionCommand);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_Vx_x.slr[*1]
Namespace	OmronLib\PackML30
Function block and function number	00129
Publish/Do not publish source code	Not published.
Function block and function version	1.00

[*1] Vx_x indicates version.

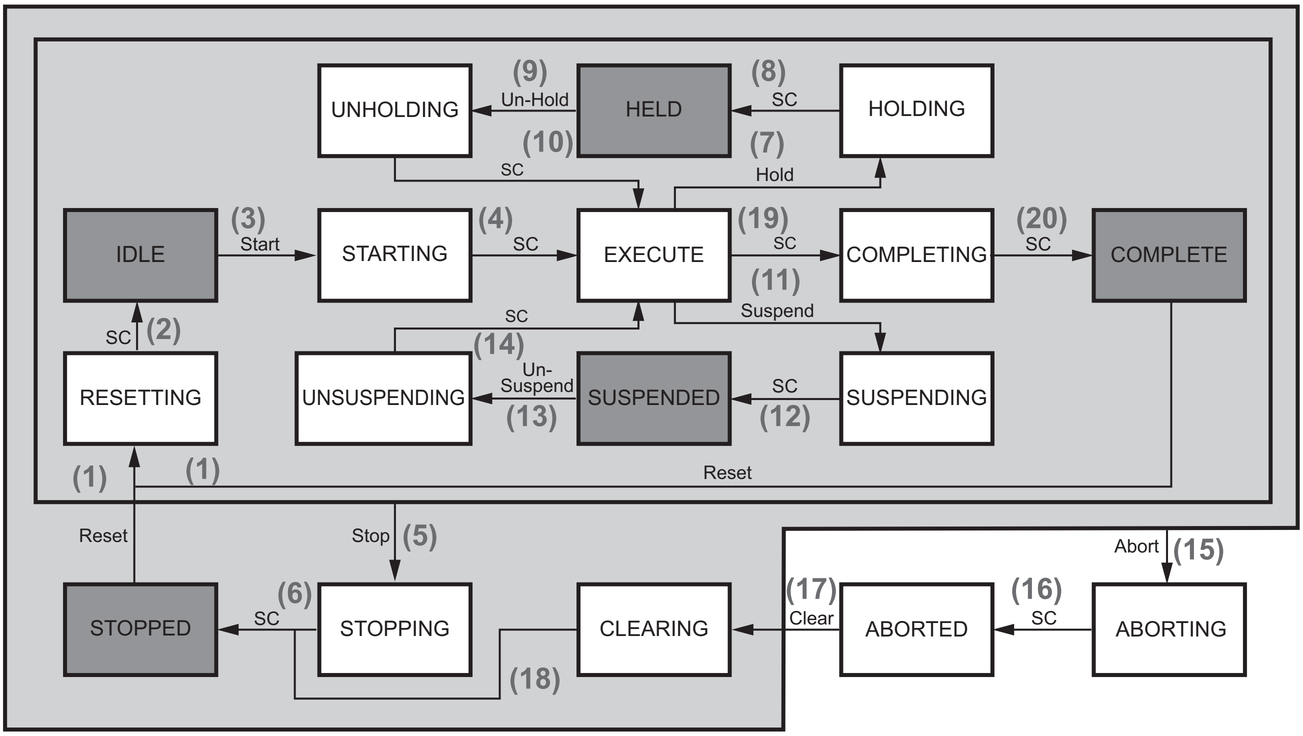
Variables

Name	In/Out	Data Type	Default	Description
EN	Input	BOOL	TRUE	Function Execution Control Flag. At FALSE, internal logic does not operate even if it is called.
ENO	Output	BOOL	-	Function Execution Control Flag Output
PMLTransitionCommand	In/Out	OmronLib\PackML30\ sPACKML _TRANSITION_COMMAND	-	PackML state transition command.

Structures

● sPACKML_TRANSITION_COMMAND

Name	Data Type	Description
sPACKML_TRANSITION_COMMAND	STRUCT	The structure that indicates a transition for the PackML state machine.
Cmd_Reset	BOOL	The command to execute the state transition from *Stopped* or *Complete* to *Resetting*. (1)
Sts_Resetting_SC	BOOL	The request to execute the state transition from *Resetting* to *Idle*. (2)
Cmd_Start	BOOL	The command to execute the state transition from *Idle* to *Starting*. (3)
Sts_Starting_SC	BOOL	The request to execute the state transition from *Starting* to *Execute*. (4)
Cmd_Stop	BOOL	The command to execute the state transition from *Idle*, *Resetting*, *Starting*, *Execute*, *Completing*, *Complete*, *Holding*, *Held*, *Unholding*, *Suspending*, *Suspended*, or *Unssupending* to *Stopping*. (5)
Sts_Stopping_SC	BOOL	The request to execute the state transition from *Stopping* to *Stopped*. (6)
Cmd_Hold	BOOL	The command to execute the state transition from *Execute* to *Holding*. (7)
Sts_Holding_SC	BOOL	The request to execute the state transition from *Holding* to *Held*. (8)
Cmd_UnHold	BOOL	The command to execute the state transition from *Held* to *UnHolding*. (9)
Sts_UnHolding_SC	BOOL	The request to execute the state transition from *UnHolding* to *Execute*. (10)
Cmd_Suspend	BOOL	The command to execute the state transition to *Execute* to *Suspending*. (11)
Sts_Suspending_SC	BOOL	The request to execute the state transition from *Suspending* to *Suspended*. (12)
Cmd_UnSuspend	BOOL	The command to execute the state transition from *Suspended* to *UnSuspending*. (13)
Sts_UnSuspending_SC	BOOL	The request to execute the state transition from *UnSuspending* to *Execute*. (14)
Cmd_Abort	BOOL	The command to execute the state transition from the state except *Aborting* and *Aborted* state, to *Aborting*. (15)
Sts_Aborting_SC	BOOL	The request to execute the state transition from *Aborting* to *Aborted*. (16)
Cmd_Clear	BOOL	The command to execute the state transition from *Aborted* to *Clearing*. (17)
Sts_Clearing_SC	BOOL	The request to execute the state transition from *Clearing* to *Stopped*. (18)
Sts_Execute_SC	BOOL	The command to execute the state transition from *Execute* to *Completing*. (19)
Sts_Completing_SC	BOOL	The request to execute the state transition from *Completing* to *Complete*. (20)



Function

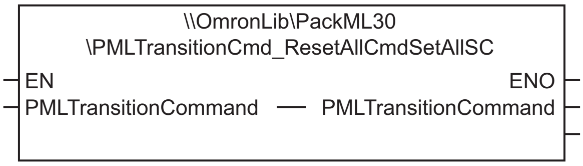
For the state transition command sPACKML_TRANSITION_COMMAND structure-type variables, this function resets every BOOL member that indicates the state transition to FALSE.

This function is used for initializing the state transition request to the host module.

PMLTransitionCmd_ResetAllCmd SetAllSC

For the state transition command sPACKML_TRANSITION_COMMAND structure variable, resets all the state transition commands (Cmd_<state name>) in the BOOL type members which indicates state transition to FALSE, and sets all the Wait state complete report (STs_<state name>_SC) to TRUE.

This function is used for initializing the state transition command to the host module.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
PML Transition Cmd_ ResetAll Cmd SetAllSC	Transition Command Reset State Set	FUN	 \\OmronLib\PackML30 \PMLTransitionCmd_ResetAllCmdSetAllSC EN ENO PMLTransitionCommand PMLTransitionCommand	\\OmronLib\PackML30\PML TransitionCmd_ResetAll CmdSetAllSC (PMLTransitionCommand);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_Vx_x.slr[*1]
Namespace	OmronLib\PackML30
Function block and function number	00130
Publish/Do not publish source code	Not published.
Function block and function version	1.00

[*1] Vx_x indicates version.

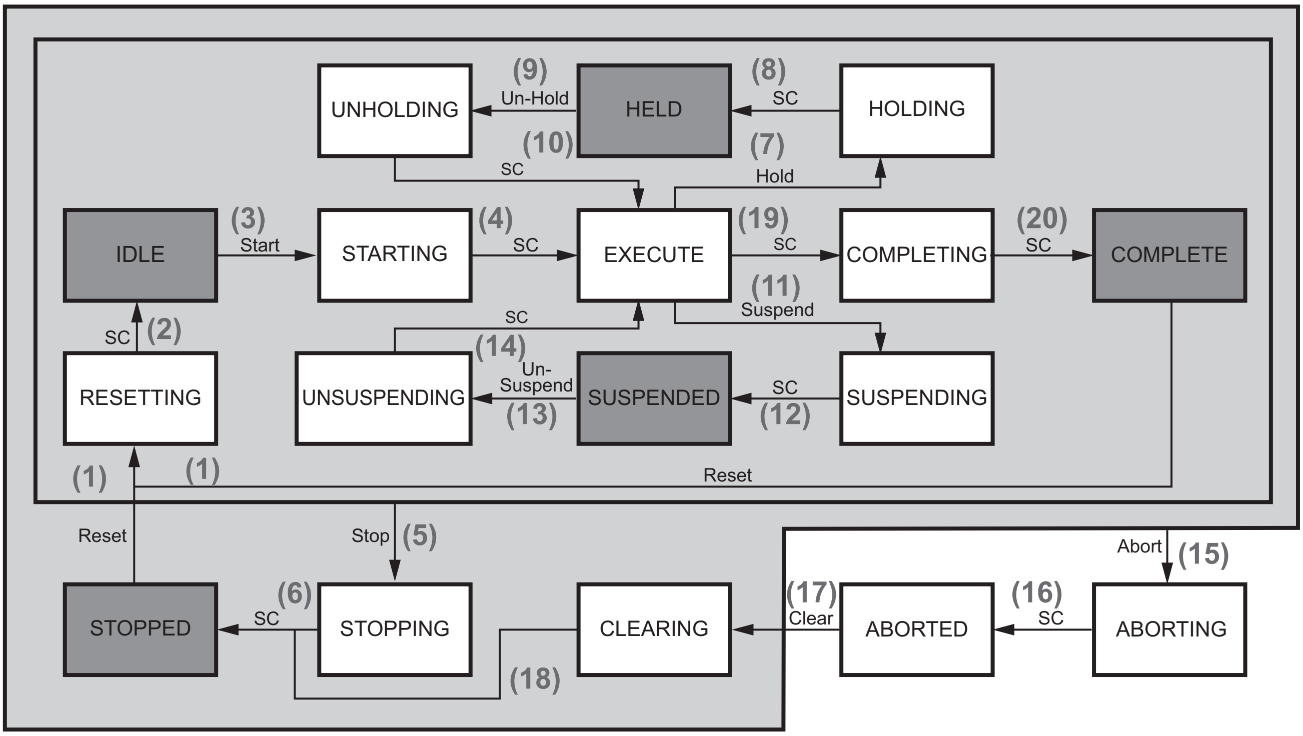
Variables

Name	In/Out	Data Type	Default	Description
EN	Input	BOOL	TRUE	Function Execution Control Flag. At FALSE, internal logic does not operate even if it is called.
ENO	Output	BOOL	-	Function Execution Control Flag Output
PMLTransitionCommand	In/Out	OmronLib\PackML30\ sPACKML _TRANSITION_COMMAND	-	PackML state transition command

Structures

● sPACKML_TRANSITION_COMMAND

Name	Data Type	Description
sPACKML_TRANSITION_COMMAND	STRUCT	The structure that indicates a transition for the PackML state machine.
Cmd_Reset	BOOL	The command to execute the state transition from *Stopped* or *Complete* to *Resetting*. (1)
Sts_Resetting_SC	BOOL	The request to execute the state transition from *Resetting* to *Idle*. (2)
Cmd_Start	BOOL	The command to execute the state transition from *Idle* to *Starting*. (3)
Sts_Starting_SC	BOOL	The request to execute the state transition from *Starting* to *Execute*. (4)
Cmd_Stop	BOOL	The command to execute the state transition from *Idle*, *Resetting*, *Starting*, *Execute*, *Completing*, *Complete*, *Holding*, *Held*, *Unholding*, *Suspending*, *Suspended*, or *Unsuspending* to *Stopping*. (5)
Sts_Stopping_SC	BOOL	The request to execute the state transition from *Stopping* to *Stopped*. (6)
Cmd_Hold	BOOL	The command to execute the state transition from *Execute* to *Holding*. (7)
Sts_Holding_SC	BOOL	The request to execute the state transition from *Holding* to *Held*. (8)
Cmd_UnHold	BOOL	The command to execute the state transition from *Held* to *UnHolding*. (9)
Sts_UnHolding_SC	BOOL	The request to execute the state transition from *UnHolding* to *Execute*. (10)
Cmd_Suspend	BOOL	The command to execute the state transition to *Execute* to *Suspending*. (11)
Sts_Suspending_SC	BOOL	The request to execute the state transition from *Suspending* to *Suspended*. (12)
Cmd_UnSuspend	BOOL	The command to execute the state transition from *Suspended* to *UnSuspending*. (13)
Sts_UnSuspending_SC	BOOL	The request to execute the state transition from *UnSuspending* to *Execute*. (14)
Cmd_Abort	BOOL	The command to execute the state transition from the state except *Aborting* and *Aborted* state, to *Aborting*. (15)
Sts_Aborting_SC	BOOL	The request to execute the state transition from *Aborting* to *Aborted*. (16)
Cmd_Clear	BOOL	The command to execute the state transition from *Aborted* to *Clearing*. (17)
Sts_Clearing_SC	BOOL	The request to execute the state transition from *Clearing* to *Stopped*. (18)
Sts_Execute_SC	BOOL	The command to execute the state transition from *Execute* to *Completing*. (19)
Sts_Completing_SC	BOOL	The request to execute the state transition from *Completing* to *Complete*. (20)



Function

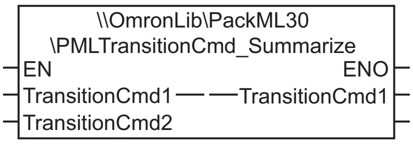
For the state transition command sPACKML_TRANSITION_COMMAND structure-type variables, this function resets every BOOL-type members, which represent the state transitions, to FALSE.

This function is used for initializing the state transition request to the host module.

PMLTransitionCmd_Summarize

Execute State Transition Commands sPACKML_TRANSITION_COMMANDs which are output by each lower module as follows, and output them as the State Transition Commands for the host module,

- Execute OR evaluation on State transition commands (Cmd_<state name>)
- Execute AND evaluation on *Wait* state completion notifications (Sts_<state name>_SC)

Function block name	Name	FB/ FUN	Graphic expression	ST expression
PMLTransitionCmd _Summarize	Transition Command Summarize	FUN		\\OmronLib\PackML30\PMLTransitionCmd_Summarize (TransitionCmd1, TransitionCmd2,);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_Vx_x.slr[*1]
Namespace	OmronLib\PackML30
Function block and function number	00131
Publish/Do not publish source code	Not published.
Function block and function version	1.00

*1. Vx_x indicates version.

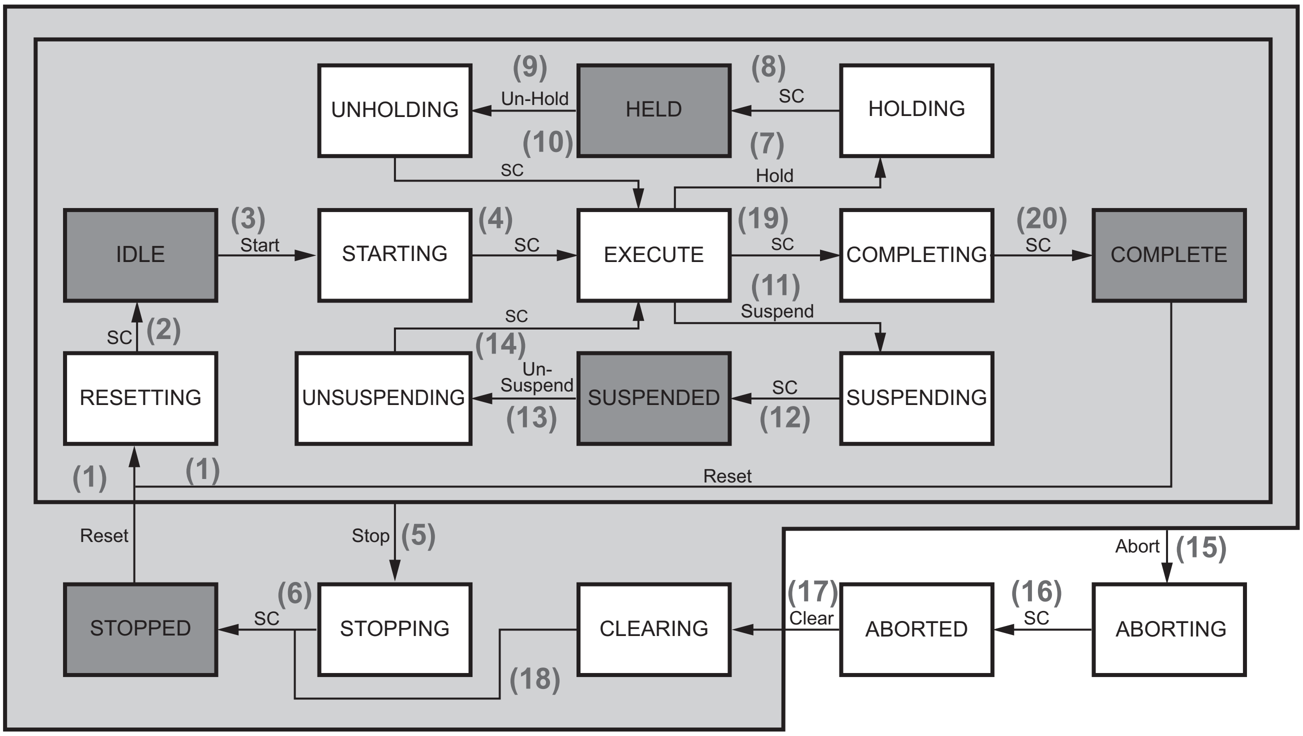
Variables

Name	In/Out	Data Type	Default	Description
EN	Input	BOOL	TRUE	Function Execution Control Flag. At FALSE, internal logic does not operate even if it is called.
ENO	Output	BOOL	-	Function Execution Control Flag Output
Transition1	In/Out	OmronLib\PackML30\ sPACKML _TRANSITINO_COMMAND	-	Specifies transition command which Transition2 are to be merged into.
Transition2	Input	OmronLib\PackML30\ sPACKML _TRANSITINO_COMMAND	FALSE (All member)	Specifies a state transition request to be merged into Transition1.

Structures

● sPACKML_TRANSITION_COMMAND

Name	Data Type	Description
sPACKML_TRANSITION_COMMAND	STRUCT	The structure that indicates a transition for the PackML state machine.
Cmd_Reset	BOOL	The command to execute the state transition from *Stopped* or *Complete* to *Resetting*. (1)
Sts_Resetting_SC	BOOL	The request to execute the state transition from *Resetting* to *Idle*. (2)
Cmd_Start	BOOL	The command to execute the state transition from *Idle* to *Starting*. (3)
Sts_Starting_SC	BOOL	The request to execute the state transition from *Starting* to *Execute*. (4)
Cmd_Stop	BOOL	The command to execute the state transition from *Idle*, *Resetting*, *Starting*, *Execute*, *Completing*, *Complete*, *Holding*, *Held*, *Unholding*, *Suspending*, *Suspended*, or *Unsuspending* to *Stopping*. (5)
Sts_Stopping_SC	BOOL	The request to execute the state transition from *Stopping* to *Stopped*. (6)
Cmd_Hold	BOOL	The command to execute the state transition from *Execute* to *Holding*. (7)
Sts_Holding_SC	BOOL	The request to execute the state transition from *Holding* to *Held*. (8)
Cmd_UnHold	BOOL	The command to execute the state transition from *Held* to *UnHolding*. (9)
Sts_UnHolding_SC	BOOL	The request to execute the state transition from *UnHolding* to *Execute*. (10)
Cmd_Suspend	BOOL	The command to execute the state transition to *Execute* to *Suspending*. (11)
Sts_Suspending_SC	BOOL	The request to execute the state transition from *Suspending* to *Suspended*. (12)
Cmd_UnSuspend	BOOL	The command to execute the state transition from *Suspended* to *UnSuspending*. (13)
Sts_UnSuspending_SC	BOOL	The request to execute the state transition from *UnSuspending* to *Execute*. (14)
Cmd_Abort	BOOL	The command to execute the state transition from the state except *Aborting* and *Aborted* state, to *Aborting*. (15)
Sts_Aborting_SC	BOOL	The request to execute the state transition from *Aborting* to *Aborted*. (16)
Cmd_Clear	BOOL	The command to execute the state transition from *Aborted* to *Clearing*. (17)
Sts_Clearing_SC	BOOL	The request to execute the state transition from *Clearing* to *Stopped*. (18)
Sts_Execute_SC	BOOL	The command to execute the state transition from *Execute* to *Completing*. (19)
Sts_Completing_SC	BOOL	The request to execute the state transition from *Completing* to *Complete*. (20)



Function

State transition requests are merged for the host module by processing the state transition requests (sPACKML_TRANSITION_COMMAND structure-type variables) arisen from the lower modules as described below:

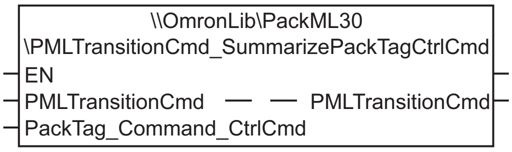
- Execute OR evaluation on State transition commands (Cmd_<state name>)
- Execute AND evaluation on *Wait* state completion notifications (Sts_<state name>_SC)

This function is used for merging each of state transition requests of CM below EM into the state transitions of EM, and for merging each of state transition requests of EM into the state transition requests of UN.

PMLTransitionCmd_Summarize PackTagCtrlCmd

This function processes and reflects the state transition requests coming from outside of the machine through the Command.CtrlCmd tag of PackTag to the summarize destination as follows.

- Execute OR evaluation on State transition commands (Cmd_<state name>)
- Execute AND evaluation on Waite state completion notifications (Sts_<state name>_SC)

Function block name	Name	FB/ FUN	Graphic expression	ST expression
PMLTransitionCmd_SummarizePackTagCtrlCmd	Pack Tag Transition Command	FUN	 \\OmronLib\PackML30 \PMLTransitionCmd_SummarizePackTagCtrlCmd EN PMLTransitionCmd PMLTransitionCmd PackTag_Command_CtrlCmd	\\OmronLib\PackML30\PMLTransitionCmd_ SummarizePackTagCtrlCmd (PMLTransitionCmd, PackTag_Command_CtrlCmd);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_Vx_x.slr[*1]
Namespace	OmronLib\PackML30
Function block and function number	00132
Publish/Do not publish source code	Not published.
Function block and function version	1.00

*1. Vx_x indicates version.

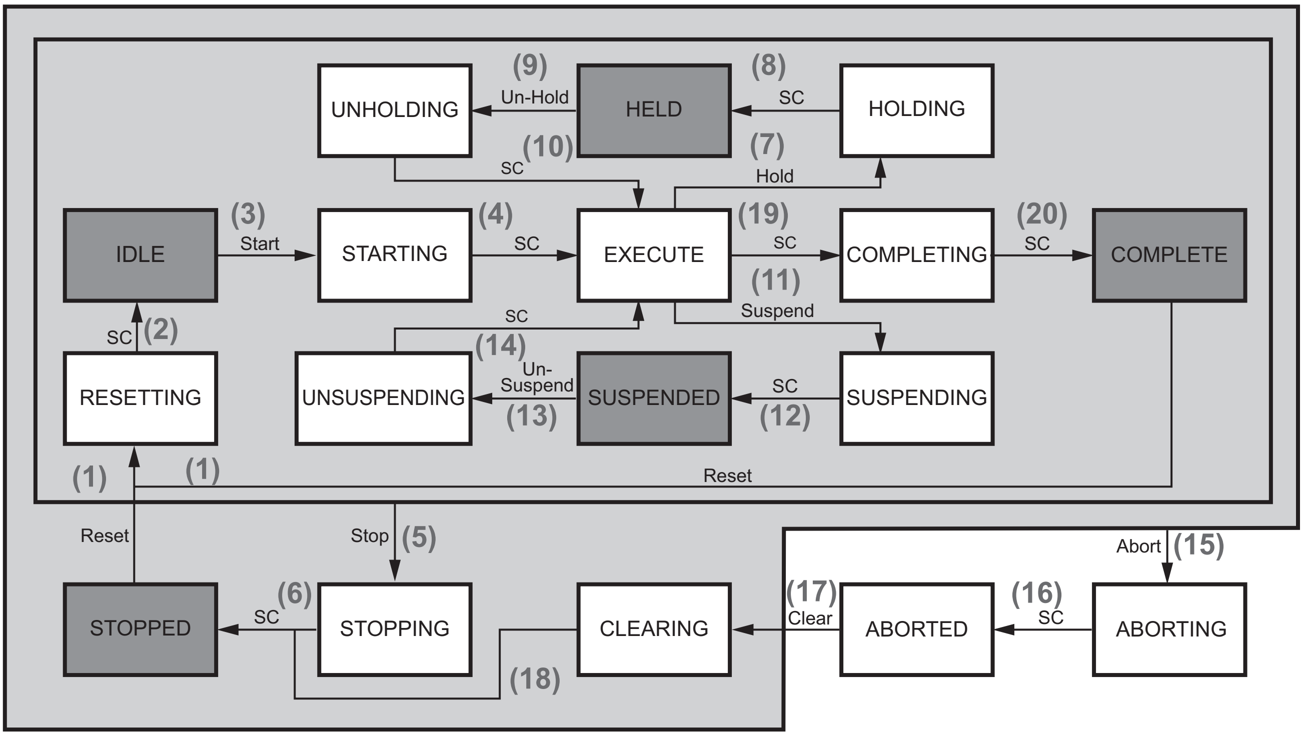
Variables

Name	In/Out	Data Type	Default	Description
EN	Input	BOOL	TRUE	Function Execution Control Flag. At FALSE, internal logic does not operate even if it is called.
ENO	Output	BOOL	-	Function Execution Control Flag Output
PMLTransitionCommand	In/Out	OmronLib\PackML30\sPACKML_TRANSITINO_COMMAND	-	Transition commands from outside of the machine are to be merged.
PackTag_Command_CtrlCmd	Input	DINT	0	The transition requests gained by PackTag Command from outside of the machine. These gained state transitions should be merged into *PMLTransitionCommand*.

Structures

● sPACKML_TRANSITION_COMMAND

Name	Data Type	Description
sPACKML_TRANSITION_COMMAND	STRUCT	The structure that indicates a transition for the PackML state machine.
Cmd_Reset	BOOL	The command to execute the state transition from *Stopped* or *Complete* to *Resetting*. (1)
Sts_Resetting_SC	BOOL	The request to execute the state transition from *Resetting* to *Idle*. (2)
Cmd_Start	BOOL	The command to execute the state transition from *Idle* to *Starting*. (3)
Sts_Starting_SC	BOOL	The request to execute the state transition from *Starting* to *Execute*. (4)
Cmd_Stop	BOOL	The command to execute the state transition from *Idle*, *Resetting*, *Starting*, *Execute*, *Completing*, *Complete*, *Holding*, *Held*, *Unholding*, *Suspending*, *Suspended*, or *Unsuspending* to *Stopping*. (5)
Sts_Stopping_SC	BOOL	The request to execute the state transition from *Stopping* to *Stopped*. (6)
Cmd_Hold	BOOL	The command to execute the state transition from *Execute* to *Holding*. (7)
Sts_Holding_SC	BOOL	The request to execute the state transition from *Holding* to *Held*. (8)
Cmd_UnHold	BOOL	The command to execute the state transition from *Held* to *UnHolding*. (9)
Sts_UnHolding_SC	BOOL	The request to execute the state transition from *UnHolding* to *Execute*. (10)
Cmd_Suspend	BOOL	The command to execute the state transition to *Execute* to *Suspending*. (11)
Sts_Suspending_SC	BOOL	The request to execute the state transition from *Suspending* to *Suspended*. (12)
Cmd_UnSuspend	BOOL	The command to execute the state transition from *Suspended* to *UnSuspending*. (13)
Sts_UnSuspending_SC	BOOL	The request to execute the state transition from *UnSuspending* to *Execute*. (14)
Cmd_Abort	BOOL	The command to execute the state transition from the state except *Aborting* and *Aborted* state, to *Aborting*. (15)
Sts_Aborting_SC	BOOL	The request to execute the state transition from *Aborting* to *Aborted*. (16)
Cmd_Clear	BOOL	The command to execute the state transition from *Aborted* to *Clearing*. (17)
Sts_Clearing_SC	BOOL	The request to execute the state transition from *Clearing* to *Stopped*. (18)
Sts_Execute_SC	BOOL	The command to execute the state transition from *Execute* to *Completing*. (19)
Sts_Completing_SC	BOOL	The request to execute the state transition from *Completing* to *Complete*. (20)

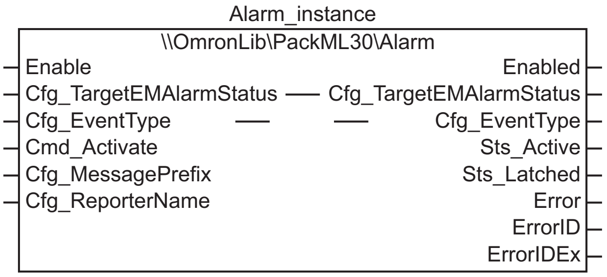


Function

The function merges the state transition requests from outside of the machine through the Command.CntrlCmd tag of PackTag, and the state transition requests gained in the machine by merging the state transition requests from EM and CM below UN.

Alarm

This function defines “*Alarm*” to support events. This function reports the state of the defined *Alarm* to *sALARM_STATUS* structure-type variables under the host module control.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
Alarm	Alarm	FB	 Alarm_instance \\OmronLib\PackML30\Alarm Enable Cfg_TargetEMAlarmStatus Cfg_EventType Cmd_Activate Cfg_MessagePrefix Cfg_ReporterName Enabled Cfg_TargetEMAlarmStatus Cfg_EventType Sts_Active Sts_Latched Error ErrorID ErrorIDEx	Alarm_instance (Enable, Cfg_TargetEMAlarmStatus, Cfg_EventType, Cmd_Activate, Cfg_MessagePrefix, Cfg_ReporterName, Enabled, Sts_Active, Sts_Latched, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_Vx_x.slr*1
Namespace	OmronLib\PackML30
Function block and function number	00133
Publish/Do not publish source code	Not published.
Function block and function version	1.00

*1. Vx_x indicates version.

Variables

Name	In/Out	Data Type	Default	Description
Enable	Input	BOOL	FALSE	FB-enabled flag. Enables this function block. At FALSE, nothing executes.
Cfg_TargetEMAlarmStatus	In/Out	OmronLib\PackML30\sALARM_STATUS	-	Report destination alarm status. Specifies *sAlarmStatus* variables to which this Alarm status is reported. Do not change the value while *Enable* is TRUE.
Cfg_EventType	In/Out	OmronLib\PackML30\sEVENT_CFG	-	Event type. Specifies the event type to be supported as *Alarm*. Do not change the value while *Enable* is TRUE.
Cmd_Activate	Input	BOOL	FALSE	Alarm activation flag input. Sets TRUE when *Alarm* is activated after the Event occurs. To reset, sets FALSE.
Cfg_MessagePrefix	Input	STRING[10]	‘’	Alarm message prefix. When reporting *Alarm*, specifies a prefix that should be attached to the message specified by *Cfg_EventType*.
Cfg_ReporterName	Input	STRING[100]	‘’	Report source name. Specifies the necessary name in order to identify the *Alarm* report source. (for debugging)
Enabled	Output	BOOL	-	FB-enabled flag output. It becomes TRUE when *Enable* becomes TRUE and this function block is operating normally.
Sts_Active	Output	BOOL	-	Alarm Activation Flag Output. It becomes TRUE when this *Alarm* is activated.
Sts_Latched	Output	BOOL	-	Alarm Latch Flag Output. When this Alarm is activated, it becomes TRUE. Even after being reset, it retains TRUE. When it is reset by *AlarmStatus_Update* function, it goes back to FALSE.
Error	Output	BOOL	-	Output Error It is always 0 (normal) because internal error never occurs in this function block.
ErrorID	Output	WORD	-	Output ErrorID It is always 0 (normal) because internal error never occurs in this function block.
ErrorIDEx	Output	DWORD	-	Output ErrorIDEx It is always 0 (normal) because internal error never occurs in this function block.

Structures

● sEVENT_CFG

This is the structure that retains detail information of events that Alarm supports.

Name	Data Type	Description
sEVENT_CFG	STRUCT	The structure that defines the events to be used for sALARM.
ID	DINT	An identifier of event type
Value	DINT	Additional information of event type
Message	STRING[80]	Message to be indicated for event
Description	STRING[256]	Detailed description of event type
Category	USINT	Event category number. Range (0 to 9)

● sALARM_STATUS

This is the structure that merges the states of *Alarm* collected per EM (equipment module).

This structure is available only with OmronLib_PackML30_V1_0.slr.

Name	Data Type	Description
sALARM_STATUS	STRUCT	The structure that shows the states of Alarm collected per equipment module (EM)
Sts_FirstOutAlarm	OmronLib\PackML30\sALARM	The snapshot of the first active Alarm
Sts_FirstOutAlarmByCategory	OmronLib\PackML30\sALARM	The snapshot of the first active Alarm in each category
Sts_Alarms	ARRAY[0..29] OF OmronLib\PackML30\sALARM	The array of Alarm collected by the equipment module
Sts_NumOfAlarms	UINT	The actual size of the above Alarm array
Sts_CategoryActiveFlag	ARRAY[0..9] OF BOOL	The array of the flag that shows whether each category includes active Alarm or not. The array index represents the category number.
Sts_CategoryLatchedFlag	ARRAY[0..9] OF BOOL	The array of the flag that shows whether each category includes Alarm has an evidence that it used to be active or not. The array index represents the category number.
NeedToBeUpdated	BOOL	The flag that shows the necessity of updating data by AlarmStatus_Update function block because the state of Sts_Alarm is updated by Alarm function block.
NeedToBeSummarized	BOOL	The flag that shows the necessity of updating AlarmSummation

Function

This function defines “*Alarm*” to support events. This function reports the state of the defined *Alarm* to *sALARM_STATUS* structure-type variables under the host module control.

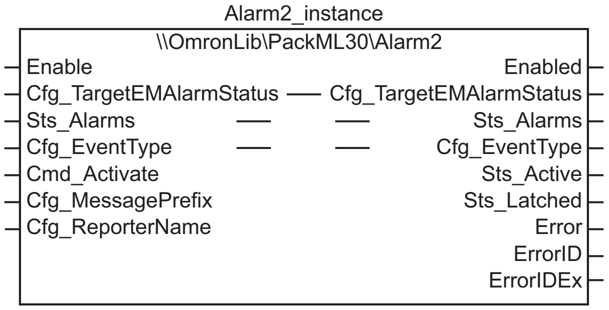
Operation Specification

- At first execution, this function block checks Sts_NumOfAlarms of the internal variable, *Cfg_TargetEMAlarmStatus*; retains the index number of the unused element of the Sts_Alarms array in the internal variable; and increments *Sts_NumOfAlarms*.

- Afterwards, the function block writes the contents based on other input variables for the *sALARM* structure-type variable of the index number of internal input *Cfg_TargetEMAlarmStatus* member *Sts_Alarms* (*sALARM* structure array), and then sets *NeedToBeUpdated* for TRUE.
- The function block outputs the member corresponding to *sALARM* structure of its index number.

Alarm2

This function defines “*Alarm*” to support events. This function reports the state of the defined *Alarm* to *sALARM_STATUS2* structure-type variables under host module control. *Sts_Alarms* is a variable-length array.

FB/FUN name	Name	FB/ FUN	Graphic expression	ST expression
Alarm2	Alarm 2	FB	 Alarm2_instance \\OmronLib\PackML30\Alarm2 Enable Cfg_TargetEMAlarmStatus Sts_Alarms Cfg_EventType Cmd_Activate Cfg_MessagePrefix Cfg_ReporterName Enabled Cfg_TargetEMAlarmStatus Sts_Alarms Cfg_EventType Sts_Active Sts_Latched Error ErrorID ErrorIDEx	Alarm2_instance(Enable, Cfg_TargetEMAlarmStatus, Sts_Alarms, Cfg_EventType, Cmd_Activate, Cfg_MessagePrefix, Cfg_ReporterName, Enabled, Sts_Active, Sts_Latched, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_V2_0.slr
Namespace	OmronLib\PackML30
FUN/FB number	00219
Source code	Not published.

Variables

Input Variables

Name	Data type	Description	Valid range	Unit	Default
Enable	BOOL	FB-enabled flag. Enables the function block function. At FALSE, nothing executes.	TRUE, FALSE	-	FALSE
Cmd_Activate	BOOL	Alarm activation flag input. Sets TRUE when *Alarm* is activated after the Event occurs. To reset, sets FALSE.	TRUE, FALSE	-	FALSE
Cfg_Message Prefix	STRING[10]	Alarm message prefix. When reporting *Alarm*, specifies a prefix that should be attached to the message specified by *Cfg_EventType*.	10 bytes max. (9 single-byte alphanumeric characters plus the final NULL character)	-	‘’
Cfg_Reporter-Name	STRING[100]	Report source name. Specifies the necessary name in order to identify the *Alarm* report source.	100 bytes max. (99 single-byte alphanumeric characters plus the final NULL character)	-	‘’

Output Variables

Name	Data type	Description	Valid range	Unit	Default
Enabled	BOOL	FB-enabled flag output. It becomes TRUE when *Enable* becomes TRUE and this function block is operating normally.	TRUE, FALSE	-	-
Sts_Active	BOOL	Alarm Activation Flag Output. It becomes TRUE when this *Alarm* is activated.	TRUE, FALSE	-	-
Sts_Latched	BOOL	Alarm Latch Flag Output. When this Alarm is activated, it becomes TRUE. Even after being reset, it retains TRUE. When it is reset by *AlarmStatus_Update2* function, it goes back to FALSE.	TRUE, FALSE	-	-
Error	BOOL	Output Error.	TRUE, FALSE	-	-
ErrorID	WORD	The value is 16#3CC8 in the event of an error. The value is 16#0000 for a normal end.	16#0000 or 16#3CC8	-	-
ErrorIDEx	DWORD	The value is an expansion error code for an error end. The value is 16#00000000 for a normal end.	(*1)	-	-

*1. Refer to *Function* on page 258 for details.

In-Out Variables

Name	Data type	Description	Valid range	Unit	Default
Cfg_TargetEMAlarmStatus	OmronLib\PackML30\sALARM_STATUS2	Report destination alarm status. Specifies *sALARM_STATUS2* type variables to which this Alarm status is reported. * Do not change the value while *Enable* is TRUE.	Depends on data type.	-	-
Sts_Alarms[][*1]	ARRAY[*] OF OmronLib\PackML30\sALARM	Array of Alarms collected by the equipment module.	Depends on data type.	-	-
Cfg_EventType	OmronLib\PackML30\sEVENT_CFG	Event type. Specifies the event type to be supported as *Alarm*. * Do not change the value while *Enable* is TRUE.	Depends on data type.	-	-

[*1] The maximum number of array elements is 500. The first number of array element should be 0.

Structures

● OmronLib\PackML30\sALARM_STATUS2

This is the structure that merges the states of *Alarm* collected per EM (equipment module).

This structure is available only with OmronLib_PackML30_V2_0.slr or later.

Member name	Data type	Description
Sts_FirstOutAlarm	OmronLib\PackML30\sALARM	The snapshot of the first active Alarm.
Sts_FirstOutAlarmByCategory	ARRAY[0..9] OF OmronLib\PackML30\sALARM	The snapshot of the first active Alarm in each category.
Sts_NumOfAlarms	UINT	Number of data stored in Alarm array.
Sts_CategoryActiveFlag	ARRAY[0..9] OF BOOL	The array of the flag that shows whether each category includes active Alarm or not. The array element number represents the category number.
Sts_CategoryLatchedFlag	ARRAY[0..9] OF BOOL	Array of flag indicating whether each category includes any Latched Alarms (Alarms with evidence of having been active). The array element number represents the category number.
NeedToBeUpdated	BOOL	The flag that shows the necessity of updating data by AlarmStatus_Update2 because the state of Sts_Alarms[] is updated.
NeedToBeSummarized	BOOL	The flag that shows the necessity of updating AlarmSummation.

OmronLib\PackML30\sALARM

This is the structure that represents a single Alarm.

Member name	Data type	Description
EventType	OmronLib\PackML30 \sEVENT_CFG	Event type supported by this Alarm.
OccuredTime	DATE_AND_TIME	Event occurrence time.
Active	BOOL	The flag that indicates whether this Alarm is active or not (not acknowledged yet).
Latched	BOOL	Flag indicating this Alarm has been active (i.e. a related event has occurred) since the last reset.
AcknowledgedTime	DATE_AND_TIME	Time when this Alarm became inactive.
ReporterName	STRING[256]	Information, which shows a source of Alarm, for debugging.

OmronLib\PackML30\sEVENT_CFG

This is the structure that retains detail information of events that Alarm supports.

Member name	Data type	Description
ID	DINT	Identifier of event type
Value	DINT	Additional information of event type
Message	STRING[80]	Message to be indicated for event
Description	STRING[256]	Detailed description of event type
Category	USINT	Event category number. Range (0 to 9)

Function

This function defines "*Alarm*" to support events. This function reports the state of the defined *Alarm* to *sALARM_STATUS2* structure-type variables under host module control.

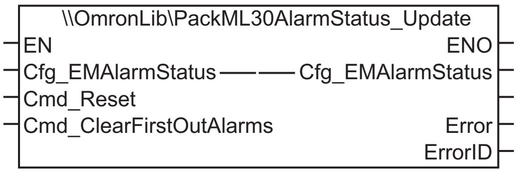
Operation Specification

- At first execution, this function block checks Sts_NumOfAlarms of *Cfg_TargetEMAlarmStatus*, retains the unused element numbers of *Sts_Alarms[]* as an internal variable; and increments *Sts_NumOfAlarms*.
- Afterwards, the function block writes content based on other input variables to a *sALARM* structure-type variable of *Sts_Alarms[]*, of the relevant array element number. It also changes *NeedToBeUpdated* to TRUE.
- The function block outputs to its output variable the corresponding member of the *sALARM* structure with its array element number.
- In the following cases, the function block changes the output variable *Error* to TRUE and outputs 16#3CC8 to *ErrorID*.
  - A number other than 0 is assigned to the first number of *Sts_Alarms* array element, or the number of *Sts_Alarms[]* array elements exceeds 500 (ErrorIDEx=16#00000001).
  - An instance of this function block has been executed more times than the number of *Sts_Alarms[]* array elements (ErrorIDEx=16#00000002).

AlarmStatus_Update

This function checks whether each alarm status changed against *Cfg_EMAlarmStatus* that indicates the status of Alarms collected to EM as In/Out variables, and then updates each member of *sALARM_STATUS*.

Also, the function block resets *Cfg_EMAlarmStatus* based on the instruction given as In/Out variables.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
AlarmStatus_ Update	EM Alarm Status Update	FUN	 \\OmronLib\PackML30AlarmStatus_Update EN ENO Cfg_EMAlarmStatus Cfg_EMAlarmStatus Cmd_Reset Cmd_ClearFirstOutAlarms Error ErrorID	\\OmronLib\PackML30\AlarmStatus_Update (Cfg_EMAlarmStatus, Cmd_Reset, Cmd_ClearFirstOutAlarms, Error, ErrorID);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_Vx_x.slr[*1]
Namespace	OmronLib\PackML30
Function block and function number	00134
Publish/Do not publish source code	Not published.
Function block and function version	1.00

*1. Vx_x indicates version.

Variables

Name	In/Out	Data Type	Default	Description
EN	Input	BOOL	TRUE	Execution Control Flag. At TRUE, the internal code in this function is executed. At FALSE, nothing executes.
Cfg_EMAlarmStatus	In/Out	OmronLib\PackML30\sALARM_STATUS	-	Update Alarm States. Specifies the Alarm status variable to be updated by this function.
Cmd_Reset	Input	BOOL	FALSE	Reset Command. At TRUE, all information except *FirstOutAlarm* of the target alarm status is reset.[*1]
Cmd_ClearFirstOutAlarms	Input	BOOL	FALSE	First Alarm Clear Command. At TRUE, *FirstOutAlarm* of the target Alarm status is cleared.
ENO	Output	BOOL	-	Execution Control Flag Output. EN is reflected as it is.
Error	Output	BOOL	-	Error Output. In this function block, FALSE is always output for Error.
ErrorID	Output	WORD	-	ErrorID Output. In this function block, 0 is always output for ErrorID.

*1. The data of *sALARM_STATUS* thar *Cmd_Reset* reset are as follows.
Sts_Alarms (except for *EventType*)
Sts_CategoryActive
Sts_CategoryLatchedFlag

Structures

● sALARM_STATUS

This is the structure that merges the states of *Alarm* collected per EM (equipment module).

Name	Data Type	Description
sALARM_STATUS	STRUCT	The structure that shows the states of Alarm collected per equipment module (EM).
Sts_FirstOutAlarm	OmronLib\PackML30\sALARM	The snapshot of the first active Alarm.
Sts_FirstOutAlarmByCategory	OmronLib\PackML30\sALARM	The snapshot of the first active Alarm in each category.
Sts_Alarms	ARRAY[0..29] OF OmronLib\PackML30\sALARM	The array of Alarm collected by the equipment module.
Sts_NumOfAlarms	UINT	The actual size of the above Alarm array.
Sts_CategoryActiveFlag	ARRAY[0..9] OF BOOL	The array of the flag that shows whether each category includes active Alarm or not. The array index represents the category number.
Sts_CategoryLatchedFlag	ARRAY[0..9] OF BOOL	The array of the flag that shows whether each category includes Latched Alarm (Alarm has an evidence that it used to be active) or not. The array index represents the category number.

Name	Data Type	Description
NeedToBeUpdated	BOOL	This flag is not used by user. The flag that shows the necessity of updating data by AlarmStatus_Update function block because the state of Sts_Alarm is updated by Alarm function block.
NeedToBeSummarized	BOOL	The flag that shows the necessity of updating AlarmSummation.

Function

This function checks whether each alarm status changed against *Cfg_EMAlarmStatus* that indicates the status of Alarms collected to EM as In/Out variables, and then updates each member of *sALARM_STATUS*.

Also, the function block resets *Cfg_EMAlarmStatus* based on the instruction given as In/Out variables.

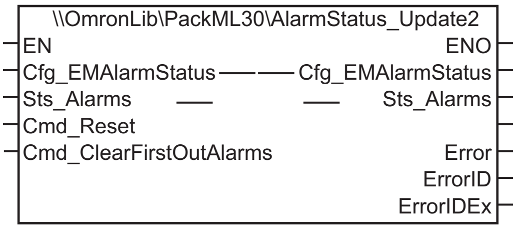
Operation Specification

- When *Cmd_Reset* is TRUE, the function block resets *Active* and *Latched*, which are elements of *Cfg_EMAlarmStatus.Sts_Alarms*, to FALSE.
- When *Cmd_ClearFirstOutAlarm* is TRUE, the function block clears *sALARM_STATUS.Sts_FirstOutAlarm*.

AlarmStatus_Update2

This function checks *Cfg_EMAlarmStatus*, which indicates the status of Alarms collected to EM as in-out variables, to see whether the status of each Alarm has changed. The function then updates each member of *sALARM_STATUS2*.

Also, the function resets *Cfg_EMAlarmStatus* based on instructions given as input variables. *Sts_Alarms* is a variable-length array.

FB/FUN name	Name	FB/ FUN	Graphic expression	ST expression
AlarmStatus_ Update2	EM Alarm Status Update 2	FUN	 \\OmronLib\PackML30\AlarmStatus_Update2 EN ENO Cfg_EMAlarmStatus Cfg_EMAlarmStatus Sts_Alarms Sts_Alarms Cmd_Reset Cmd_ClearFirstOutAlarms Error ErrorID ErrorIDEx	\\OmronLib\PackML30\AlarmStatus_Update2(Cfg_EMAlarmStatus, Sts_Alarms, Cmd_Reset, Cmd_ClearFirstOutAlarms, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_V2_0.slr
Namespace	OmronLib\PackML30
FUN/FB number	00220
Source code	Not published.

Variables

Input Variables

Name	Data type	Description	Valid range	Unit	Default
EN	BOOL	Execution control flag. At TRUE, the internal code in this function is executed. At FALSE, nothing executes.	TRUE, FALSE	-	FALSE
Cmd_Reset	BOOL	Reset command. At TRUE, all information except *FirstOutAlarm* of the target Alarm status is reset.[*1]	TRUE, FALSE	-	FALSE
Cmd_ClearFirstOutAlarms	BOOL	First Alarm Clear Command. At TRUE, *FirstOutAlarm* of the target Alarm status is cleared.	TRUE, FALSE	-	FALSE

*1. The following data is reset by *Cmd_Reset*.

- *Sts_Alarms[]* (except EvenType)
- *Cmd_ClearFirstOutAlarms .Sts_CategoryAcriveFlag*
- *Cmd_ClearFirstOutAlarms .Sts_CategoryLatchedFlag*

Output Variables

Name	Data type	Description	Valid range	Unit	Default
ENO	BOOL	Execution control flag output. EN is reflected as it is.	TRUE, FALSE	-	-
Error	BOOL	Output error.	TRUE, FALSE	-	-
ErrorID	WORD	The value is 16#3CC9 in the event of an error. The value is 16#0000 for a normal end.	16#0000 or 16#3CC9	-	-
ErrorIDEx	DWORD	The value is an expansion error code for an error end. The value is 16#00000000 for a normal end.	(*1)	-	-

*1. Refer to *Function* on page 265 for details.

In-Out Variables

Name	Data type	Description	Valid range	Unit	Default
Cfg_EMAlarm Status	OmronLib\PackML30\sALARM_STATUS2	Update Alarm States Specifies the Alarm status variable to be updated by this function.	Depends on data type.	-	-
Sts_Alarms[][*1]	ARRAY[*] OF OmronLib\PackML30\sALARM	Array of Alarms collected by the equipment module.	Depends on data type.	-	-

*1. The maximum number of array elements is 500. The first number of array element should be 0.

Structures

● OmronLib\PackML30\sALARM_STATUS2

This is the structure that merges the states of *Alarm* collected per EM (equipment module).

This structure is available only with OmronLib_PackML30_V2_0.slr or later.

Member name	Data type	Description
Sts_FirstOutAlarm	OmronLib\PackML30 \sALARM	The snapshot of the first active Alarm.
Sts_FirstOutAlarmByCategory	ARRAY[0..9] OF OmronLib\PackML30 \sALARM	The snapshot of the first active Alarm in each category.
Sts_NumOfAlarms	UINT	Number of data stored in Alarm array.
Sts_CategoryActiveFlag	ARRAY[0..9] OF BOOL	The array of the flag that shows whether each category includes active Alarm or not. The array element number represents the category number.
Sts_CategoryLatchedFlag	ARRAY[0..9] OF BOOL	Array of flag indicating whether each category includes any Latched Alarms (Alarms with evidence of having been active). The array element number represents the category number.
NeedToBeUpdated	BOOL	The flag that shows the necessity of updating data by AlarmStatus_Update2 because the state of Sts_Alarms[] is updated.
NeedToBeSummarized	BOOL	The flag that shows the necessity of updating AlarmSummation.

● OmronLib\PackML30\sALARM

This is the structure that represents a single Alarm.

Member name	Data type	Description
EventType	OmronLib\PackML30 \sEVENT_CFG	Event type supported by this Alarm.
OccuredTime	DATE_AND_TIME	Event occurrence time.
Active	BOOL	The flag that indicates whether this Alarm is active or not (not acknowledged yet).
Latched	BOOL	Flag indicating this Alarm has been active (i.e. a related event has occurred) since the last reset.
AcknowledgedTime	DATE_AND_TIME	Time when this Alarm became inactive.
ReporterName	STRING[256]	Information, which shows a source of Alarm, for debugging.

● OmronLib\PackML30\sEVENT_CFG

This is the structure that retains detail information of events that Alarm supports.

Member name	Data type	Description
ID	DINT	Identifier of event type
Value	DINT	Additional information of event type
Message	STRING[80]	Message to be indicated for event
Description	STRING[256]	Detailed description of event type
Category	USINT	Event category number. Range (0 to 9)

Function

This function checks *Cfg_EMAlarmStatus*, which indicates the status of Alarms collected to EM as in-out variables, to see whether the status of each Alarm has changed. The function then updates each member of *sALARM_STATUS2*.

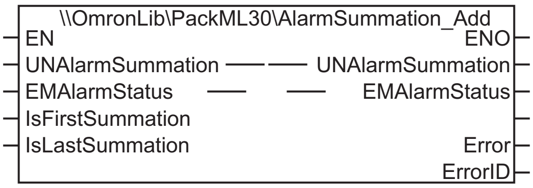
Also, the function resets *Cfg_EMAlarmStatus* based on instructions given as input variables.

Operation Specification

- When *Cmd_Reset* is TRUE, the function block changes *Active* and *Latched* of each *Cfg_EMAlarmStatus.Sts_Alarms* element to FALSE.
- When *Cmd_ClearFirstOutAlarm* is TRUE, the function block clears *sALARM_STATUS2.Sts_FirstOutAlarm*.
- When *Cfg_EMAlarmStatus.NeedToBeUpdated* is TRUE, the function block updates each Sts_Alarms[] member.
- In the following cases, the function block changes the output variable *Error* to TRUE and outputs 16#3CC9 to *ErrorID*.
  - A number other than 0 is assigned to the first number of *Sts_Alarms[]* array element, or the number of *Sts_Alarms[]* array elements exceeds 500 (ErrorIDEx=16#00000001).

AlarmSummation_Add

This function adds the specific EM *Alarm* status given by In/Out variable *EMAlarmStatus* for the In/Out variable that retains the *Alarm* statuses merged to *UNAlarmSummation* UN (unit/machine).

Function block name	Name	FB/ FUN	Graphic expression	ST expression
Alarm Summation _Add	UN Alarm Status Add	FUN	 \\OmronLib\PackML30\AlarmSummation_Add EN ENO UNAlarmSummation UNAlarmSummation EMAlarmStatus EMAlarmStatus IsFirstSummation IsLastSummation Error ErrorID	\\OmronLib\PackML30\AlarmSummation_Add (UNAlarmSummation, EMAlarmStatus, IsFirstSummation, IsLastSummation, Error, ErrorID);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_Vx_x.slr[*1]
Namespace	OmronLib\PackML30
Function block and function number	00135
Publish/Do not publish source code	Not published.
Function block and function version	1.00

*1. Vx_x indicates version.

Variables

Name	In/Out	Data Type	Default	Description
EN	Input	BOOL	TRUE	Execution Control Flag. At TRUE, the internal code in this function is executed. At FALSE, nothing executes.
UNAlarmSummation	In/Out	OmronLib\PackML30 \sALARM _SUMMMATION	-	Update Machine-level Alarm Status. Specifies the *sALARM_SUMMATION* variable to be updated by this function.
EMAlarmStatus	In/Out	OmronLib\PackML30 \sALARM_STATUS	-	Added EM Alarm Status. The EM-level alarm status that should be added to the machine-level alarm status.
IsFirstSummation	Input	BOOL	FALSE	Sets TRUE when the first EM-level status is added to the machine-level alarm status. At TRUE, *UNAlarmSummation* is cleared and then *EMAlarmStatus* is added on top. At FALSE, it is added to the tail of the existing valid array size.
IsLastSummation	Input	BOOL	TRUE	Sets TRUE when the last EM-level status is added to the machine-level alarm status. At TRUE, the necessary members for *UNAlarmSummation* are updated after *EMAlarmStatus* is added to *UNAlarmSummation*. While multiple *EMAlarmStatus* are added, addition of this Input FALSE can skip unneeded update operation to speed up production.
ENO	Output	BOOL	-	Execution Control Flag. EN is reflected as it is.
Error	Output	BOOL	-	Error Output. In this function block, FALSE is always output for Error.
ErrorID	Output	WORD	-	ErrorID Output. In this function block, 0 is always output for ErrorID.

Structures

● sALARM_STATUS

This is the structure that merges the states of *Alarm* collected per EM (equipment module).

Name		Data Type	Description
sALARM_STATUS		STRUCT	The structure that shows the states of Alarm collected per equipment module (EM).
	Sts_FirstOutAlarm	OmronLib\PackML30\sALARM	The snapshot of the first active Alarm.
	Sts_FirstOutAlarmByCategory	OmronLib\PackML30\sALARM	The snapshot of the first active Alarm in each category.
	Sts_Alarms	ARRAY[0..29] OF OmronLib\PackML30\sALARM	The array of Alarm collected by the equipment module.
	Sts_NumOfAlarms	UINT	The actual size of the above Alarm array.
	Sts_CategoryActiveFlag	ARRAY[0..9] OF BOOL	The array of the flag that shows whether each category includes active Alarm or not. The array index represents the category number.
	Sts_CategoryLatchedFlag	ARRAY[0..9] OF BOOL	The array of the flag that shows whether each category includes Alarm has an evidence that it used to be active or not. The array index represents the category number.
	NeedToBeUpdated	BOOL	The flag that shows the necessity of updating data by AlarmStatus_Update function block because the state of Sts_Alarm is updated by Alarm function block.
	NeedToBeSummarized	BOOL	The flag that shows the necessity of updating AlarmSummation.

● sALARM_SUMMATION

The structure that merges *Alarms* collected from all EM below UN (unit/machine).

Name		Data Type	Description
sALARM_SUMMATION		STRUCT	Alarm status information collected by UN (machines).
	Sts_FirstOutAlarm	OmronLib\PackML30\sALARM	The first active Alarm.
	Sts_Alarms	ARRAY[0..100] OF OmronLib\PackML30\sALARM	The array of collected Alarms.
	Sts_NumOfAlarms	UINT	The actual size of the above Alarm array.
	ActiveOneExists	BOOL	The array of the flag that shows whether each category includes active Alarm or not. The array index represents the category number.
	LatchedOneExists	BOOL	The array of the flag that shows whether each category includes Alarm has an evidence that it used to be active or not. The array index represents the category number.
	Sts_CategoryActiveFlag	ARRAY[0..9] OF BOOL	The first active Alarm.
	Sts_CategoryLatchedFlag	ARRAY[0..9] OF BOOL	The array of Alarms collected by the equipment module.

Function

This function adds the specific EM *Alarm* status given by In/Out variable *EMAlarmStatus* for the In/Out variable that retains the *Alarm* statuses merged to *UNAlarmSummation* UN (unit/machine).

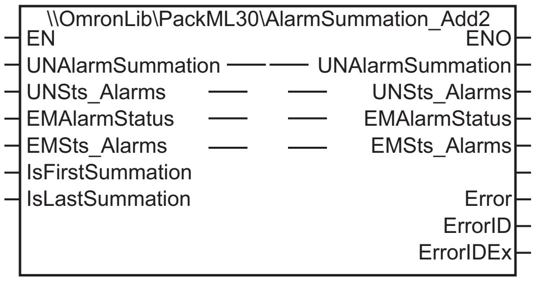
Operation Specification

This function performs following operations when EN = TRUE.

- When *IsFirstSummation* is TRUE, clear each value of *UNAlarmSummation*, and store Alarms, which are retained by *EMAlarmStatus*, from the head of the same member *Alarm* array.
  Also, when *IsFirstSummation* is FALSE, without clearing the value, store Alarms, which are retained by *EMAlarmStatus*, from the end of the Alarm array.
- When *IsLastSummation* is TRUE, add *AlarmStatus* to *AlarmSummation* to update value of each member for *AlarmSummation*.

AlarmSummation_Add2

This function adds the specific EM *Alarm* status given by in-out variable *EMAlarmStatus* to the in-out variable *UNAlarmSummation* retaining the *Alarm* statuses merged to UN (unit/machine). *UNSts_Alarms* and *EMSts_Alarms* are variable-length arrays.

FB/FUN name	Name	FB/ FUN	Graphic expression	ST expression
AlarmSum-mation_Add2	UN Alarm Status Add 2	FUN	 \\OmronLib\PackML30\AlarmSummation_Add2 EN ENO UNAlarmSummation UNAlarmSummation UNSts_Alarms UNSts_Alarms EMAlarmStatus EMAlarmStatus EMSts_Alarms EMSts_Alarms IsFirstSummation IsLastSummation Error ErrorID ErrorIDEx	\\OmronLib\PackML30\Alarm Summation_Add2(UNAlarmSummation, UNSts_Alarms, EMAlarmStatus, EMSts_Alarms, IsFirstSummation, IsLastSummation, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_V2_0.slr
Namespace	OmronLib\PackML30
FUN/FB number	00221
Source code	Not published.

Variables

Input Variables

Name	Data type	Description	Valid range	Unit	Default
EN	BOOL	Execution control flag. At TRUE, the internal code in this function is executed. At FALSE, nothing executes.	TRUE, FALSE	-	FALSE
IsFirstSummation	BOOL	Set to TRUE when the first EM-level status is added to the machine-level alarm status. When this input variable is TRUE, *UNAlarmSummation* is cleared and then *EMAlarmStatus* is added on the top. When FALSE, it is added to the end of the existing valid array size.	TRUE, FALSE	-	FALSE
IsLastSummation	BOOL	Set to TRUE when the last EM-level status is added to the machine-level alarm status. When TRUE, *UNAlarmSummation* members are updated as necessary after *EMAlarmStatus* is added to *UNAlarmSummation*. When adding multiple *EMAlarmStatus*, setting this input variable to FALSE allows you to speed up the process by skipping unnecessary update procedures.	TRUE, FALSE	-	FALSE

Output Variables

Name	Data type	Description	Valid range	Unit	Default
ENO	BOOL	Execution control flag output. EN is reflected as it is.	TRUE, FALSE	-	-
Error	BOOL	Output error.	TRUE, FALSE	-	-
ErrorID	WORD	The value is 16#3CCA in the event of an error. The value is 16#0000 for a normal end.	16#0000 or 16#3CCA	-	-
ErrorIDEx	DWORD	The value is an expansion error code for an error end. The value is 16#00000000 for a normal end.	(*1)	-	-

*1. Refer to *Function* on page 274 for details.

In-Out Variables

Name	Data type	Description	Valid range	Unit	Default
UNAlarmSummation	OmronLib\PackML30\sALARM_SUMMATION2	Update Alarm Status Specifies the Alarm status variable to be updated by this function.	Depends on data type.	-	-
UNSts_Alarms[][*1]	ARRAY[*] OF OmronLib\PackML30\sALARM	Array of Update Machine-level Alarm Status	Depends on data type.	-	-
EMAlarmStatus	OmronLib\PackML30\sALARM_STATUS2	The EM-level Alarm Status to add to the machine-level alarm status	Depends on data type.	-	-
EMSts_Alarms[][*1]	ARRAY[*] OF OmronLib\PackML30\sALARM	Array of EM Alarm Status to add	Depends on data type.	-	-

[*1] The maximum number of array elements is 500. The first number of array element should be 0.

Structures

● OmronLib\PackML30\sALARM_SUMMATION2

The structure that merges Alarms collected from all EM below UN (unit/machine).

This structure is available only with OmronLib_PackML30_V2_0.slr or later.

Member name	Data type	Description
Sts_FirstOutAlarm	OmronLib\PackML30\sALARM	The first active Alarm.
Sts_NumOfAlarms	UINT	The number of data stored in the array of update machine-level alarm status.
ActiveOneExists	BOOL	Flag indicating whether each category includes an active Alarm.
LatchedOneExists	BOOL	Flag indicating whether each category includes any Latched Alarms (Alarms with evidence of having been active).
Sts_CategoryActiveFlag	ARRAY[0..9] OF BOOL	The array of the flag that shows whether each category includes active Alarm or not. The array element number represents the category number.
Sts_CategoryLatchedFlag	ARRAY[0..9] OF BOOL	Array of flag indicating whether each category includes any Latched Alarms (Alarms with evidence of having been active). The array element number represents the category number.

● OmronLib\PackML30\sALARM_STATUS2

This is the structure that merges the states of *Alarm* collected per EM (equipment module).

This structure is available only with OmronLib_PackML30_V2_0.slr or later.

Member name	Data type	Description
Sts_FirstOutAlarm	OmronLib\PackML30 \sALARM	The snapshot of the first active Alarm.
Sts_FirstOutAlarmByCategory	ARRAY[0..9] OF OmronLib\PackML30 \sALARM	The snapshot of the first active Alarm in each category.
Sts_NumOfAlarms	UINT	Number of data stored in Alarm array.
Sts_CategoryActiveFlag	ARRAY[0..9] OF BOOL	The array of the flag that shows whether each category includes active Alarm or not. The array element number represents the category number.
Sts_CategoryLatchedFlag	ARRAY[0..9] OF BOOL	Array of flag indicating whether each category includes any Latched Alarms (Alarms with evidence of having been active). The array element number represents the category number.
NeedToBeUpdated	BOOL	The flag that shows the necessity of updating data by AlarmStatus_Update2 because the state of Sts_Alarms[] is updated.
NeedToBeSummarized	BOOL	The flag that shows the necessity of updating AlarmSummation.

● OmronLib\PackML30\sALARM

This is the structure that represents a single Alarm.

Member name	Data type	Description
EventType	OmronLib\PackML30 \sEVENT_CFG	Event type supported by this Alarm.
OccuredTime	DATE_AND_TIME	Event occurrence time.
Active	BOOL	The flag that indicates whether this Alarm is active or not (not acknowledged yet).
Latched	BOOL	Flag indicating this Alarm has been active (i.e. a related event has occurred) since the last reset.
AcknowledgedTime	DATE_AND_TIME	Time when this Alarm became inactive.
ReporterName	STRING[256]	Information, which shows a source of Alarm, for debugging.

● OmronLib\PackML30\sEVENT_CFG

This is the structure that retains detail information of events that Alarm supports.

Member name	Data type	Description
ID	DINT	Identifier of event type
Value	DINT	Additional information of event type
Message	STRING[80]	Message to be indicated for event
Description	STRING[256]	Detailed description of event type
Category	USINT	Event category number. Range (0 to 9)

Function

This function adds the specific EM *Alarm* status given by in-out variable *EMAlarmStatus* to the in-out variable *UNAlarmSummation* retaining the *Alarm* statuses merged to UN (unit/machine).

Operation Specification

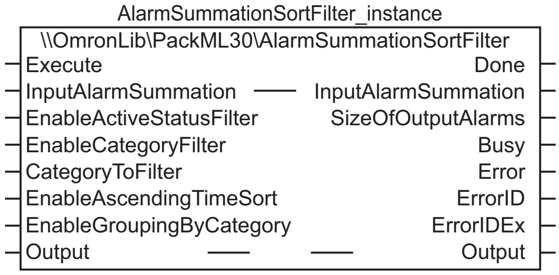
This function performs following operations when EN = TRUE.

- When *IsFirstSummation* is TRUE, the function clears each *UNAlarmSummation* value and stores Alarms retained by *EMAlarmStatus* to *UNSts_Alarms[]*, beginning with its first element.

  When *IsFirstSummation* is FALSE, the function does not clear *UNAlarmSummation* and stores Alarms retained by *EMAlarmStatus* to *UNSts_Alarms[]*, beginning with its last element.
- When *IsLastSummation* is TRUE, the function adds *EMAlarmStatus* to *UNAlarmSummation*, and then updates the value of each *UNAlarmSummation* member.
- In the following cases, the function changes the output variable *Error* to TRUE and outputs 16#3CCA to *ErrorID*.
  - A number other than 0 is assigned to the first number of *UNSts_Alarms[]* array element, or the number of *UNSts_Alarms[]* array elements exceeds 500 (ErrorIDEx=16#00000001).
  - A number other than 0 is assigned to the first number of *EMSts_Alarms[]* array element, or the number of *EMSts_Alarms[]* array elements exceeds 500 (ErrorIDEx=16#00000002).

AlarmSummation_SortFilter

This function reflects the results of filtering and sorting that are conducted with the conditions specified by the In/Out variable InputAlarmSummation that retains the Alarm statuses merged into UN (unit/machine), to sALARM array variable Output.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
Alarm Summation _SortFilter	Alarm Sort and Filter	FB	 AlarmSummationSortFilter_instance \\OmronLib\PackML30\AlarmSummationSortFilter Execute Done InputAlarmSummation InputAlarmSummation EnableActiveStatusFilter SizeOfOutputAlarms EnableCategoryFilter Busy CategoryToFilter Error EnableAscendingTimeSort ErrorID EnableGroupingByCategory ErrorIDEx Output Output	AlarmSummation_SortFilter _instance (Execute, InputAlarmSummation, EnableActiveStatusFilter, EnableCategoryFilter, CategoryToFilter, EnableAscendingTimeSort, EnableGroupingByCategory, Output, Done SizeOfOutputAlarms, Busy, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_Vx_x.slr[*1]
Namespace	OmronLib\PackML30
Function block and function number	00136
Publish/Do not publish source code	Not published.
Function block and function version	1.00

[*1] Vx_x indicates version.

Variables

Name	In/Out	Data Type	Default	Description
Execute	Input	BOOL	TRUE	Execution Control Flag. At TRUE, the internal code in this function is executed. At FALSE, nothing executes.
InputAlarmSummation	In/Out	OmronLib\PackML30 \sALARM _SUMMMATION	-	Source Alarm Information. The machine-level alarm status structure variable that includes the alarm array source of filtering and sorting.
EnableActiveStatusFilter	Input	BOOL	FALSE	Active Alarm Filter Enabled Flag. At TRUE, only the *Alarms* whose ACTIVE = TRUE is output.
EnableCategoryFilter	Input	BOOL	FALSE	Category Filter Enabled Flag. At TRUE, only the categories that are specified by the following *CategoryFilter* are output.
CategoryToFilter	Input	USINT	0	Category Number for Filter. The category number to be output by the category filter is specified. Range (0 to 9)
EnableAscendingTimeSort	Input	BOOL	FALSE	Time Ascending Flag. At TRUE, sorted alarm occurrence time in ascending order is updated to Output.
EnableGroupingByCategory	Input	BOOL	FALSE	Categorized Groping Flag. At TRUE, the sorted group numbers in ascending order are updated to Output.
Output	In/Out	ARRAY[0..100] OF OmronLib\PackML30 \sALARM	-	The Alarm array to which the sorted/filtered results are output.
Done	Output	BOOL	-	This flag turns on after having executed the function.
SizeOfOutputAlarms	Output	INT	-	Output the number of alarm that is output to *Output*.
Busy	Output	BOOL	-	This flag turns on during executing the function.
Error	Output	BOOL	-	It is always 0 (normal) because internal error never occurs in this function block.
ErrorID	Output	WORD	-	It is always 0 (normal) because internal error never occurs in this function block.
ErrorIDEx	Output	DWORD	-	It is always 0 (normal) because internal error never occurs in this function block.

Structures

● sALARM

This is the structure that represents a single Alarm.

Name		Data Type	Description
sALARM		STRUCT	The structure that represents a single Alarm.
	EventType	OmronLib\PackML30\sEVENT_CFG	Event type supported by this Alarm
	OccuredTime	DATE_AND_TIME	Event occurrence time
	Active	BOOL	The flag that indicates whether this Alarm is active or not (not acknowledged yet).
	Latched	BOOL	The flag indicates that this Alarm used to be active after the last reset. (An related event fired.)
	AcknowledgedTime	DATE_AND_TIME	Date and Time when this Alarm became inactive.
	ReporterName	STRING[256]	Information, which shows a source of Alarm, for debugging.

● sALARM_SUMMATION

The structure that merges Alarms collected from all EM below UN (unit/machine).

Name		Data Type	Description
sALARM_SUMMATION		STRUCT	Alarm status information collected by UN (machines).
	Sts_FirstOutAlarm	OmronLib\PackML30\sALARM	The first active Alarm.
	Sts_Alarms	ARRAY[0..100] OF OmronLib\PackML30\sALARM	The array of collected Alarms.
	Sts_NumOfAlarms	UINT	The actual size of the above Alarm array.
	ActiveOneExists	BOOL	The flag that shows whether each category includes active Alarm or not.
	LatchedOneExists	BOOL	The flag that shows whether each category includes Alarm has an evidence that it used to be active or not.
	Sts_CategoryActiveFlag	ARRAY[0..9] OF BOOL	The array of the flag that shows whether each category includes active Alarm or not. The array index represents the category number.
	Sts_CategoryLatchedFlag	ARRAY[0..9] OF BOOL	The array of the flag that shows whether each category includes Alarm has an evidence that it used to be active or not. The array index represents the category number.

Function

This function reflects the results of filtering and sorting that are conducted with the conditions specified by the In/Out variable *InputAlarmSummation* that retains the Alarm statuses merged into UN (unit/machine), to *sALARM* array variable Output.

Filter Function

When *Execute* turns on the FB execute as follows:

- If *EnableActiveStatusFilter* is TRUE, *Alarms* that the *Active* is TRUE are resistered *Output*.
- If *EnableCateoryFilter* is TRUE, *Alarms* that the category corresponse with the one in the *CategoryToFilter* are resistered *Output*.
- If *EnableActiveStatusFilter* and *EnableCateoryFilter* are TRUE, *Alarms* that the *Active* is TRUE and the category corresponse with the one in the *CategoryToFilter* are resistered *Output*.

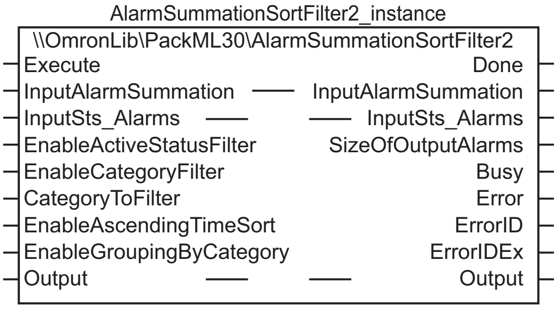
Sort Function

When *Execute* turns on the FB execute as follows:

- If *EnableAscendingTimeSort* is TRUE, *Alarms* are sorted occurrence time in ascending order is updated to *Output*.
- If *EnableGroupingByCategory* is TRUE, *Alarms* are sorted group numbers in ascending order are updated to *Output*.
- If *EnableAscendingTimeSort* and *EnableGroupingByCategory* are TRUE, first alarms are sorted group numbers in ascending order and then are sorted occurrence time in ascending order.

AlarmSummation_SortFilter2

This function reflects the results of filtering and sorting that are conducted with the conditions specified by the in-out variable *InputAlarmSummation* that retains the Alarm statuses merged into UN (unit/machine), to sALARM array variable *Output*. *InputSts_Alarms* is a variable-length array.

FB/FUN name	Name	FB/ FUN	Graphic expression	ST expression
Alarm Summation_SortFilter2	Alarm Sort and Filter 2	FB		AlarmSummation_SortFilter2_instance(Execute, InputAlarmSummation, InputSts_Alarms, EnableActiveStatusFilter, EnableCategoryFilter, CategoryToFilter, EnableAscendingTimeSort, EnableGroupingByCategory, Output, Done, SizeOfOutputAlarms, Busy, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_V2_0.slr
Namespace	OmronLib\PackML30
FUN/FB number	00222
Source code	Not published.

Variables

Input Variables

Name	Data type	Description	Valid range	Unit	Default
Execute	BOOL	Execution flag for this function block. At TRUE, the internal code in this function block is executed. At FALSE, nothing executes.	TRUE, FALSE	-	FALSE
EnableActive StatusFilter	BOOL	Active Alarm filter enabled flag. When TRUE, only the *Alarms* whose Active = TRUE are output to Output. To reset, sets FALSE.	TRUE, FALSE	-	FALSE
EnableCategory Filter	BOOL	Category filter enabled flag. When TRUE, only Alarms of categories specified by the input variable *CategoryToFilter* are output to Output.	TRUE, FALSE	-	FALSE
CategoryToFilter	USINT	Category number for filter. Specifies the category number to be output by the category filter.	0 to 9	-	0
EnableAscending TimeSort	BOOL	Time ascending flag. At TRUE, sorted Alarm occurrence time in ascending order is output to Output.	TRUE, FALSE	-	FALSE
EnableGrouping-ByCategory	BOOL	Categorized grouping flag. When TRUE, sorted group numbers in ascending order are output to Output.	TRUE, FALSE	-	FALSE

Output Variables

Name	Data type	Description	Valid range	Unit	Default
Done	BOOL	Flag indicating whether processing is complete. After processing is complete, value remains TRUE while Execute is TRUE.	TRUE, FALSE	-	-
SizeOfOutputAlarms	INT	Output the number of alarm that is stored to *Output[]*.	0 to 499	-	-
Busy	BOOL	Flag indicating whether processing is ongoing. Value is TRUE from the time Execute changes to TRUE until processing is complete.	TRUE, FALSE	-	-
Error	BOOL	Output error.	TRUE, FALSE	-	-
ErrorID	WORD	The value is 16#3CCB in the event of an error. The value is 16#0000 for a normal end.	16#0000 or 16#3CCB	-	-
ErrorIDEx	DWORD	The value is an expansion error code for an error end. The value is 16#00000000 for a normal end.	(*1)	-	-

[1] Refer to *Function* on page 283 for details.

In-Out Variables

Name	Data type	Description	Valid range	Unit	Default
InputAlarm Summation	OmronLib\PackML30\sALARM_SUMMATION2	Source Alarm Information The machine-level alarm status structure variable.	Depends on data type.	-	-
InputSts_Alarms[][1]	ARRAY[*]OF OmronLib\PackML30\sALARM	Array of source alarm information of filtering and sorting.	Depends on data type.	-	-
Output[][1]	ARRAY [*] OF OmronLib\PackML30\sALARM	Alarm array to which the sorted/filtered results are output.	Depends on data type.	-	-

[1] The maximum number of array elements is 500. The first number of array element should be 0.

Structures

● OmronLib\PackML30\sALARM_SUMMATION2

The structure that merges Alarms collected from all EM below UN (unit/machine).

This structure is available only with OmronLib_PackML30_V2_0.slr or later.

Member name	Data type	Description
Sts_FirstOutAlarm	OmronLib\PackML30 \sALARM	The first active Alarm.
Sts_NumOfAlarms	UINT	Flag indicating the number of data records stored in the array of update machine-level alarm status.
ActiveOneExists	BOOL	Flag indicating whether each category includes an active Alarm.
LatchedOneExists	BOOL	Flag indicating whether each category includes any Latched Alarms (Alarms with evidence of having been active).
Sts_CategoryActiveFlag	ARRAY[0..9] OF BOOL	The array of the flag that shows whether each category includes active Alarm or not. The array element number represents the category number.
Sts_CategoryLatchedFlag	ARRAY[0..9] OF BOOL	Array of flag indicating whether each category includes any Latched Alarms (Alarms with evidence of having been active). The array element number represents the category number.

● OmronLib\PackML30\sALARM

This is the structure that represents a single Alarm.

Member name	Data type	Description
EventType	OmronLib\PackML30 \sEVENT_CFG	Event type supported by this Alarm.
OccuredTime	DATE_AND_TIME	Event occurrence time.
Active	BOOL	The flag that indicates whether this Alarm is active or not (not acknowledged yet).
Latched	BOOL	Flag indicating this Alarm has been active (i.e. a related event has occurred) since the last reset.
AcknowledgedTime	DATE_AND_TIME	Time when this Alarm became inactive.
ReporterName	STRING[256]	Information, which shows a source of Alarm, for debugging.

● OmronLib\PackML30\sEVENT_CFG

This is the structure that retains detail information of events that Alarm supports.

Member name	Data type	Description
ID	DINT	Identifier of event type
Value	DINT	Additional information of event type
Message	STRING[80]	Message to be indicated for event
Description	STRING[256]	Detailed description of event type
Category	USINT	Event category number. Range (0 to 9)

Function

This function reflects the results of filtering and sorting that are conducted with the conditions specified by the in-out variable *InputAlarmSummation* that retains the Alarm statuses merged into UN (unit/machine), to sALARM array variable *Output[]*.

Filter Function

When *Execute* changes to TRUE, the FB execute as follows:

- When *EnableActiveStatusFilter* is TRUE, only Alarms of *InputSts_Alarms[]* whose *Active* are TRUE are stored to *Output[]*.
- When *EnableCategoryFilter* is TRUE, only Alarms of *InputSts_Alarms[]* whose categories are specified by *CategoryFilter* are stored to *Output[]* array.
- When both *EnableActiveStatusFilter* and *EnableCategoryFilter* are TRUE, Alarms whose *Active* are TRUE and whose categories are specified by *CategoryFilter* are stored to *Output[]*.

Sort Function

When *Execute* changes to TRUE, the FB execute as follows:

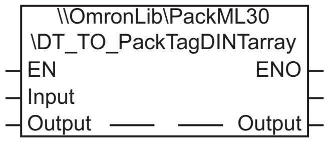
- When *EnableAscendingTimeSort* is TRUE, *InputSts_Alarms[]* is sorted in ascending order of their times of occurrence and output to *Output[]*.
- When *EnableGroupingByCategory* is TRUE, *InputSts_Alarms[]* is sorted in ascending order of their category numbers and output to *Output[]*.
- If both *EnableAscendingTimeSort* and *EnableGroupingByCategory* are TRUE, *InputSts_Alarms[]* is first sorted in ascending order of their category numbers. The *InputSts_Alarms[]* with the same category numbers are then sorted in ascending order of their times of occurrence. *InputSts_Alarms[]* is then output to *Output[]*.

In the following cases, the function changes the output variable *Error* to TRUE and outputs 16#3CCB to *ErrorID*.

- A number other than 0 is assigned to the first number of *InputSts_Alarms[]* array element, or the number of *InputSts_Alarms[]* array elements exceeds 500 (ErrorIDEx=16#00000001).
- A number other than 0 is assigned to the first number of *Output[]* array element, or the number of *Output[]* elements exceeds 500 (ErrorIDEx=16#00000002).
- The number of *InputSts_Alarms[]* array elements exceeds the number of *Output[]* array elements (ErrorIDEx=16#00000003).

DT_TO_PackTagDINTarray

This function converts the input of DATE_AND_TIME into the date-time array specified by PackTags.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
DT_TO_Pack TagDINTarray	DATE_AND_TIME Type Pack Tag Array Conversion	FUN	 \\OmronLib\PackML30 \DT_TO_PackTagDINTarray EN ENO Input Output Output	\\OmronLib\PackML30\DT_TO_PackTagDINTarray (Input, Output,);

Function Block and Function Information

Item	Description
Library file name	OmronLib_PackML30_Vx_x.slr[*1]
Namespace	OmronLib\PackML30
Function block and function number	00137
Publish/Do not publish source code	Not published.
Function block and function version	1.00

*1. Vx_x indicates version.

Variables

Name	In/Out	Data Type	Default	Description
EN	Input	BOOL	TRUE	Execution start flag. At TRUE, convert “INPUT” data to the DINT-type array variable and output it to “Output.”
Input	Input	DATE_AND_TIME	0	DATE_AND_TIME value to be converted.
Output	In/Out	ARRAY[0..6] OF DINT	-	Destination of conversion result in PackTag format. Array element 0 = Year Array element 1 = Month Array element 2 = Day Array element 3 = Hour (24hr format) Array element 4 = Min Array element 5 = Sec Array element 6 = USec (1/1,000,000 sec)

Function

This function converts the input of DATE_AND_TIME into the date-time array specified by PackTags.

Appendix

Referring to Library Information

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

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

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

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

Use the Sysmac Studio to access the library information.

Attributes of Libraries, Function Blocks and Functions

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

● Attributes of Libraries

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

[*1] These numbers correspond to the numbers shown on the screen images in the next section, *Referring to Attributes of Libraries, Function Blocks and Functions* on page 287.

[*2] It is provided in English and Japanese.

● Attributes of Function Blocks and Functions

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

[*1] These numbers correspond to the numbers shown on the screen images in the next section, *Referring to Attributes of Libraries, Function Blocks and Functions* on page 287.

[*2] It is provided in English and Japanese.

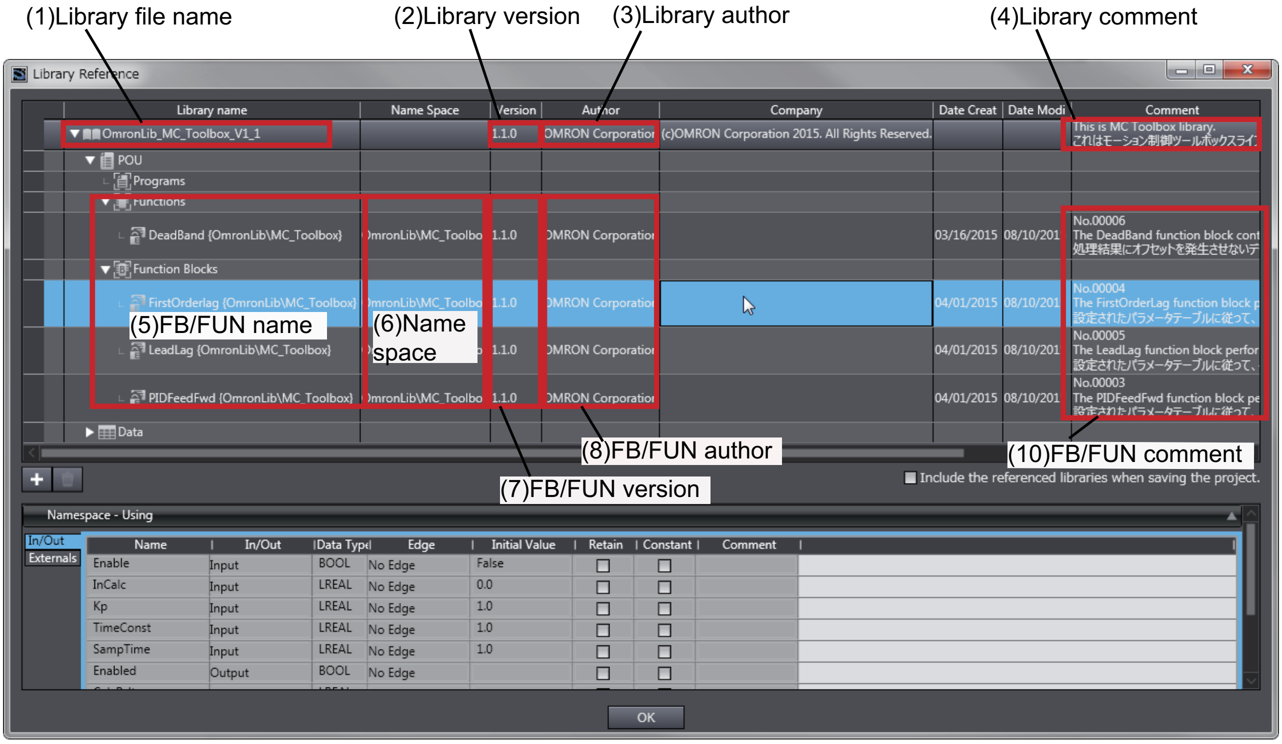
Referring to Attributes of Libraries, Function Blocks and Functions

You can refer to the attributes of libraries, function blocks and functions of the library information at the following locations on the Sysmac Studio.

- Library Reference Dialog Box
- Toolbox Pane
- Ladder Editor

(a) Library Reference Dialog Box

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



(1)Library file name

(2)Library version

(3)Library author

(4)Library comment

(5)FB/FUN name

(6)Name space

(7)FB/FUN version

(8)FB/FUN author

(10)FB/FUN comment

Library name	Name Space	Version	Author	Company	Date Creat	Date Modi	Comment
OmronLib_MC_Toolbox_V1_1		1.1.0	OMRON Corporatior	(c)OMRON Corporation 2015. All Rights Reserved.			This is MC Toolbox library. これはモーション制御ツールボックスライ
POU							
Programs							
Functions							
DeadBand {OmronLib\MC_Toolbox}	)mronLib\MC_Toolbo	1.1.0	OMRON Corporatior		03/16/2015	08/10/201	No.00006 The DeadBand function block cont 処理結果にオフセットを発生させないデ
Function Blocks							
FirstOrderlag {OmronLib\MC_Toolbox}	)mronLib\MC_Toolbo	1.1.0	OMRON Corporatior		04/01/2015	08/10/201	No.00004 The FirstOrderLag function block p 設定されたパラメータテーブルに従って、
LeadLag {OmronLib\MC_Toolbox}		1.1.0	OMRON Corporatior		04/01/2015	08/10/201	No.00005 The LeadLag function block perfor 設定されたパラメータテーブルに従って、
PIDFeedFwd {OmronLib\MC_Toolbox}	)mronLib\MC_Toolbo	1.1.0	OMRON Corporatior		04/01/2015	08/10/201	No.00003 The PIDFeedFwd function block pe 設定されたパラメータテーブルに従って、
Data							

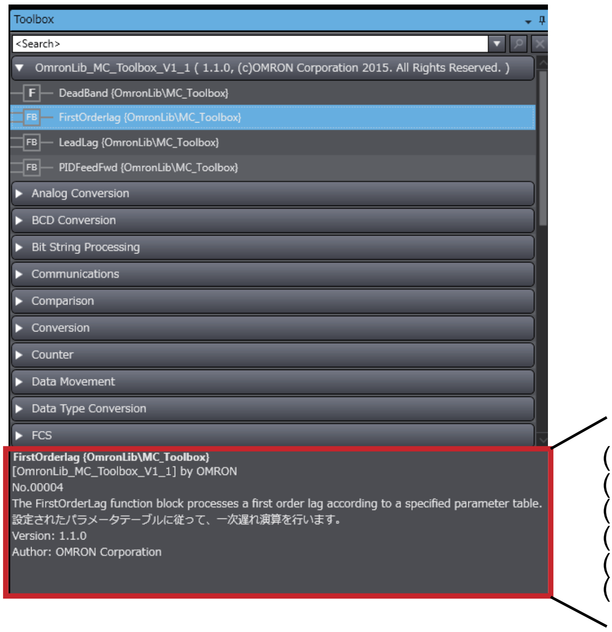
□ Include the referenced libraries when saving the project.

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

(b) Toolbox Pane

Select a function block and function to display its library information at the bottom of the Toolbox Pane.

The text “by OMRON” which is shown on the right of the library name (1) indicates that this library was provided by OMRON.

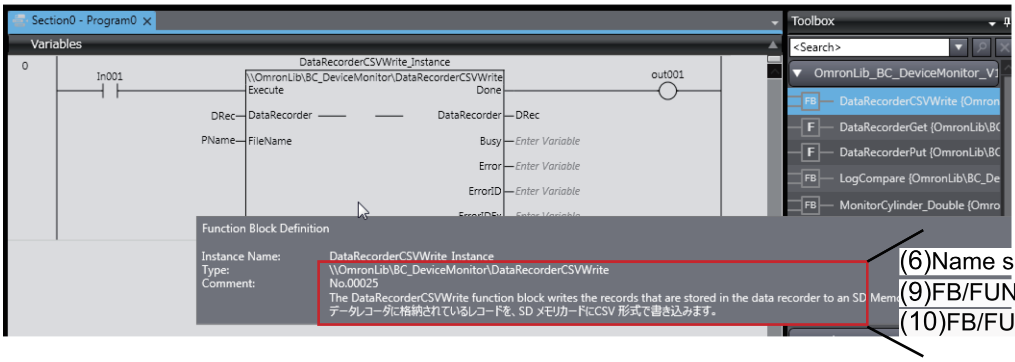

Toolbox
<Search>
OmronLib_MC_Toolbox_V1_1 (1.1.0, (c)OMRON Corporation 2015. All Rights Reserved.)
F DeadBand {OmronLib\MC_Toolbox}
FB FirstOrderlag {OmronLib\MC_Toolbox}
FB LeadLag {OmronLib\MC_Toolbox}
FB PIDFeedFwd {OmronLib\MC_Toolbox}
Analog Conversion
BCD Conversion
Bit String Processing
Communications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

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

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

(c) Ladder Editor

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


Section0 - Program0
Variables
0
In001
DataRecorderCSVWrite_Instance
\\OmronLib\BC_DeviceMonitor\DataRecorderCSVWrite
Execute
Done
out001
DRec
DataRecorder
DataRecorder
DRec
PName
FileName
Busy
Enter Variable
Error
Enter Variable
ErrorID
Enter Variable
Toolbox
<Search>
OmronLib_BC_DeviceMonitor_V1
FB DataRecorderCSVWrite {Omron
F DataRecorderGet {OmronLib\BC
F DataRecorderPut {OmronLib\BC
FB LogCompare {OmronLib\BC_De
FB MonitorCylinder_Double {Omro

Function Block Definition

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

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

Referring to Function Block and Function Source Codes

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

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

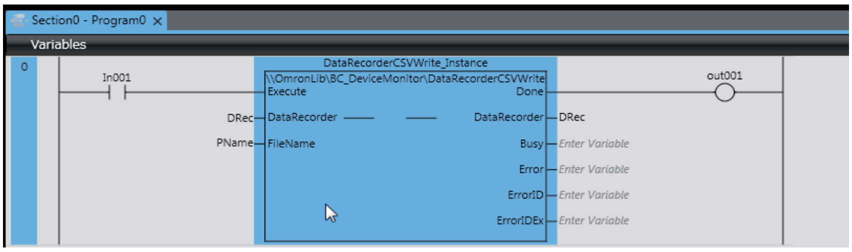
The following are the examples of items that you may need to customize.

- Customizing the size of arrays to suit the memory capacity of the user's Controller
- Customizing the data types to suit the user-defined data types

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

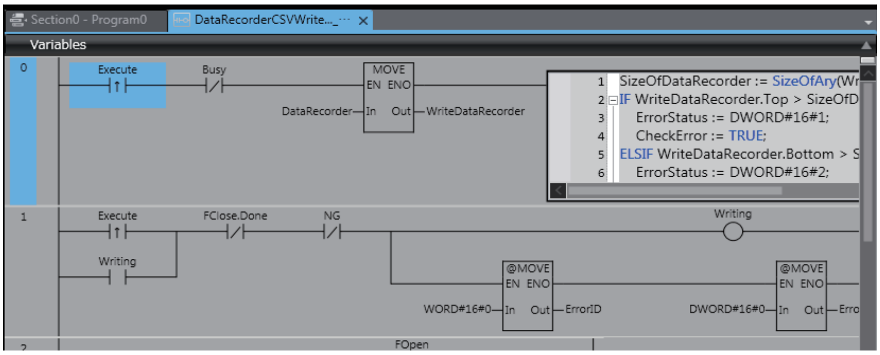
Use the following procedure to refer to the source codes of function blocks and functions.

1. Select a function block or function in the program.



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

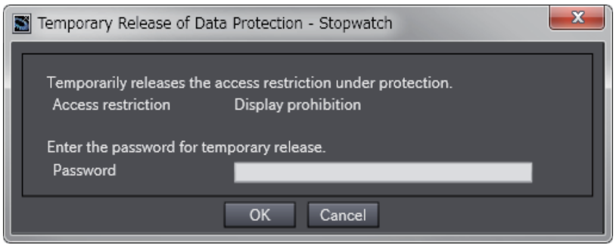
   The source code is displayed.



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

Precautions for Correct Use

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



Temporarily releases the access restriction under protection.

Access restriction Display prohibition

Enter the password for temporary release.

Password

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

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

Authorized Distributor: