

OMRON

Sysmac Library


User's Manual for Visual Feedback Alignment Library

SYSMAC-XR018

NOTE

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

Trademarks

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

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

Copyrights

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

Introduction

Thank you for purchasing an NJ/NX-series CPU Unit, PC for NY-series production.

This manual contains information that is necessary to use the Functions (sometimes abbreviated as FUN) and Function blocks (sometimes abbreviated as FB) in Visual Feedback Alignment Library.

Please read this manual and make sure you understand the functions and capabilities before you attempt to use it in a control system.

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

Make sure to read the user's manual for each product before use.

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

Features of the Library

The Visual Feedback Alignment Library is a set of software function components for alignment applications employing visual feedback.

Intended Audience

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

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

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

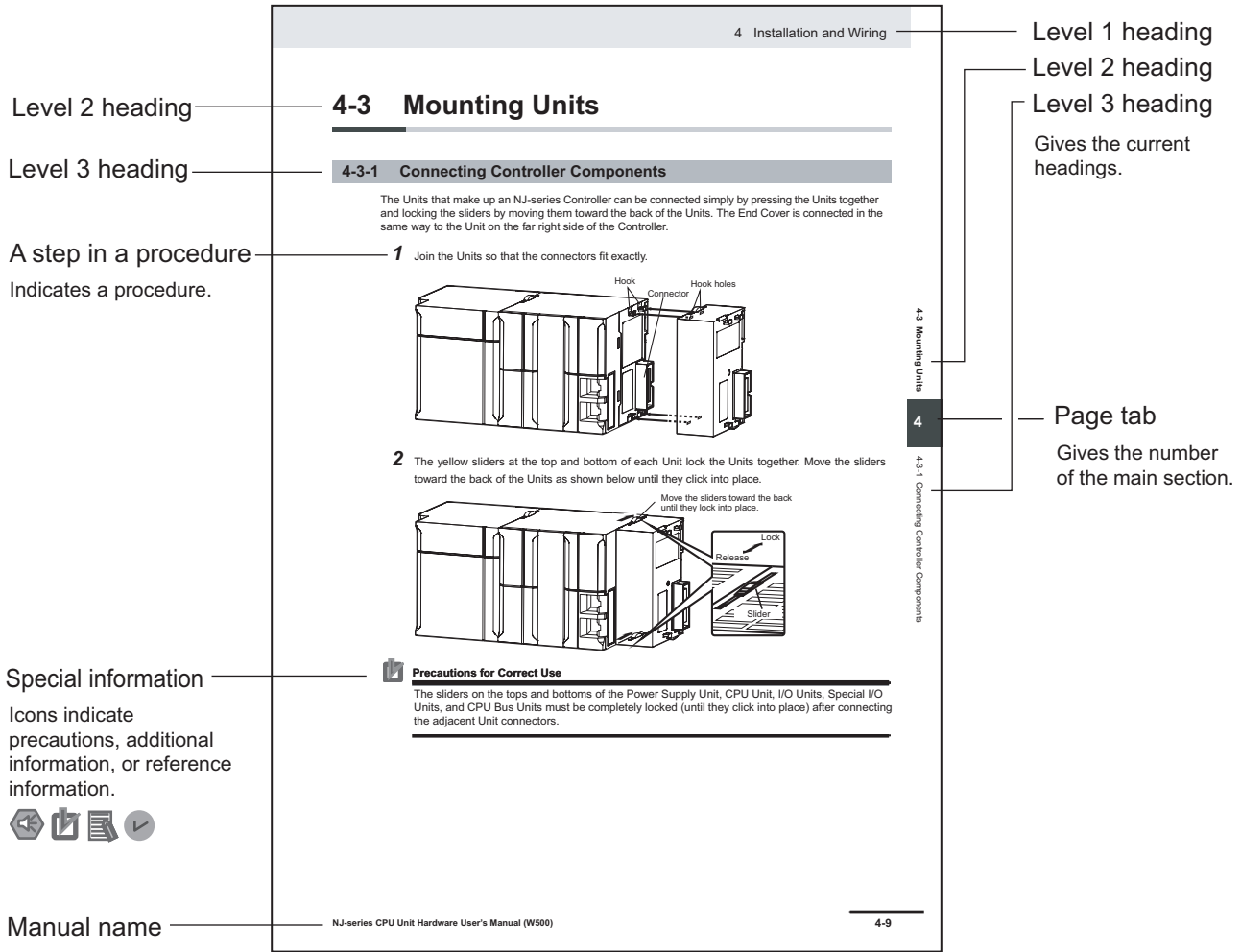
Applicable Products

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

Manual Structure

Page Structure

The following page structure is used in this manual.



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

Special Information

Special information in this manual is classified as follows:



Precautions for Safe Use

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



Precautions for Correct Use

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



Additional Information

Additional information to read as required.

This information is provided to increase understanding and make operation easier.



Version Information

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

CONTENTS

Introduction	1
Features of the Library.....	1
Intended Audience.....	1
Applicable Products	1
Manual Structure.....	2
Page Structure.....	2
Special Information	3
Sections in this Manual	21
Terms and Conditions Agreement.....	9
Warranty, Limitations of Liability	9
Application Considerations	10
Disclaimers	10
Safety Precautions.....	12
Definition of Precautionary Information.....	12
Symbols	12
WARNING.....	12
CAUTIONS	13
Precautions for Correct Use	14
Using the Library.....	14
Related Manuals.....	15
Revision History.....	19

Section 1 Sysmac Library Usage Procedure

1-1 Procedure to Use Sysmac Library Installed Using the Installer.....	1 - 2
1-1-1 Using a Newly Installed Sysmac Library	1 - 2
1-1-2 Using an Upgraded Sysmac Library	1 - 4
1-2 How to use Sysmac Library in the CPU Unit or Industrial PC.....	1 - 6

Section 2 Visual Feedback Alignment Library

2-1 Terms	2 - 2
2-2 Overview	2 - 3
2-2-1 Alignment Application.....	2 - 3
2-2-2 Visual Feedback.....	2 - 3
2-3 Restrictions and Limitations.....	2 - 6
2-4 Hardware Configuration	2 - 7
2-4-1 System Configuration.....	2 - 7
2-4-2 Supported Stages	2 - 7
2-5 Visual Feedback Alignment Library Flow Chart	2 - 9
2-5-1 Alignment Flow Chart.....	2 - 9
2-5-2 Generate Calibration Parameter Flow Chart.....	2 - 11
2-6 Visual Feedback Alignment Library Function Specification	2 - 13

2-6-1	Function Overview	2 - 13
2-6-2	Coordinate Systems	2 - 14
2-6-3	Control Blocks	2 - 15
2-6-4	Execute Measurement (Imaging of Workpiece)	2 - 17
2-6-5	When Vision Sensor NG Measurement Occurs	2 - 17

Section 3 Common Specifications of Function Blocks

3-1	Common Variables	3 - 2
3-1-1	Definition of Input Variables and Output Variables	3 - 2
3-1-2	Execute-type Function Blocks	3 - 3
3-1-3	Enable-type Function Blocks.....	3 - 5
3-2	Precautions	3 - 7
3-2-1	Nesting	3 - 7
3-2-2	Instruction Options	3 - 7
3-2-3	Re-execution of Function Blocks.....	3 - 7

Section 4 Individual Specifications of FB/FUN

Structure	4 - 2
XYθ Position (sPOSITION).....	4 - 2
Calibration Parameter (sCALIB_PARAMS)	4 - 2
Alignment Control Parameter (sALIGNMENT_PARAMS)	4 - 2
Stage Parameter (sSTAGE_PARAMS).....	4 - 3
XYθ Stage Parameter (sXYTH_STAGE_PARAMS)	4 - 4
UVWR Stage Parameter (sUVWR_STAGE_PARAMS).....	4 - 5
UVWR Axis Parameter (sUVWR_PARAMS)	4 - 6
Forward/Reverse Kinematics Calculation Parameter (sKINEMATICS_PARAMS)	4 - 9
Virtual XYθ Stage Axis Position/Velocity Calculation Parameter (sCALCVIRTUALMOVE_PARAMS)	4 - 10
Axis Motion Parameter (sAXIS_MOVE_PARAMS).....	4 - 11
Fifth Order Trajectory Calculation Parameter (sCALCURVE_PARAMS).....	4 - 13
Imaging Parameter (sIMAGING_PARAMS).....	4 - 14
Done Judgment Parameter (JUDGE_PARAMS)	4 - 14
Calibration Axis Motion Parameter (sCALIB_MOVE_PARAMS)	4 - 15
FB/FUN Structure Usage.....	4 - 17
AffineTrans	4 - 22
Function Block and Function Information	4 - 22
Variables	4 - 22
Function	4 - 23
Precautions for Correct Use	4 - 24
Sample Programming	4 - 24
Troubleshooting (Error Codes and Corrective Actions)	4 - 24
CalcPosAngle	4 - 25
Function Block and Function Information	4 - 25
Variables	4 - 25
Function	4 - 26
Precautions for Correct Use	4 - 28
Sample Programming	4 - 28
Troubleshooting (Error Codes and Corrective Actions)	4 - 28
CalcMultiPosAngle	4 - 30
Function Block and Function Information	4 - 30
Variables	4 - 30
Function	4 - 32
Precautions for Correct Use	4 - 33
Sample Programming	4 - 34
Troubleshooting (Error Codes and Corrective Actions)	4 - 34
CtrlStage	4 - 35
Function Block and Function Information	4 - 35

Variables	4 - 36
Function	4 - 38
Timing Charts.....	4 - 40
Precautions for Correct Use	4 - 42
Sample Programming	4 - 43
Troubleshooting (Error Codes and Corrective Actions)	4 - 43
CalcMovement.....	4 - 45
Function Block and Function Information	4 - 45
Variables	4 - 45
Function	4 - 46
Precautions for Correct Use	4 - 47
Sample Programming	4 - 47
Troubleshooting (Error Codes and Corrective Actions)	4 - 47
CalcInverseKinematics.....	4 - 48
Function Block and Function Information	4 - 48
Variables	4 - 48
Function	4 - 49
Precautions for Correct Use	4 - 50
Sample Programming	4 - 50
Troubleshooting (Error Codes and Corrective Actions)	4 - 50
CalcForwardKinematics	4 - 51
Function Block and Function Information	4 - 51
Variables	4 - 51
Function	4 - 52
Precautions for Correct Use	4 - 52
Sample Programming	4 - 53
Troubleshooting (Error Codes and Corrective Actions)	4 - 53
CalcAxisVelocity	4 - 54
Function Block and Function Information	4 - 54
Variables	4 - 55
Function	4 - 57
Precautions for Correct Use	4 - 63
Sample Programming	4 - 63
Troubleshooting (Error Codes and Corrective Actions)	4 - 63
JudgeAlignmentComplete.....	4 - 64
Function Block and Function Information	4 - 64
Variables	4 - 65
Function	4 - 67
Precautions for Correct Use	4 - 71
Sample Programming	4 - 71
Troubleshooting (Error Codes and Corrective Actions)	4 - 71
GenerateTrigger	4 - 73
Function Block and Function Information	4 - 73
Variables	4 - 73
Function	4 - 75
Precautions for Correct Use	4 - 77
Sample Programming	4 - 77
Troubleshooting (Error Codes and Corrective Actions)	4 - 77
GenerateCalibParams.....	4 - 78
Function Block and Function Information	4 - 78
Variables	4 - 78
Function	4 - 80
Precautions for Correct Use	4 - 87
Sample Programming	4 - 88
Troubleshooting (Error Codes and Corrective Actions)	4 - 88
Sample Programming.....	4 - 90
Overview.....	4 - 90
Auto-calibration.....	4 - 93
Alignment Control	4 - 106

Appendix

A-1 Referring to Library Information.....	A - 2
A-1-1 Library Attributes, and FB or FUN Attributes.....	A - 2
A-1-2 Referring to Attributes of Libraries, Function Blocks, and Functions	A - 3
A-2 Referring to Function Block and Function Source Codes.....	A - 5

Index

Terms and Conditions Agreement

Warranty, Limitations of Liability

Warranties

- **Exclusive Warranty**

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

- **Limitations**

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

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

- **Buyer Remedy**

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

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

Limitation on Liability; Etc

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

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

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

Application Considerations

Suitability of Use

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

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

Programmable Products

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

Disclaimers

Performance Data

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

Change in Specifications

Product specifications and accessories may be changed at any time based on improvements and other reasons. It is our practice to change part numbers when published ratings or features are changed, or when significant construction changes are made. However, some specifications of the Product may

be changed without any notice. When in doubt, special part numbers may be assigned to fix or establish key specifications for your application. Please consult with your Omron's representative at any time to confirm actual specifications of purchased Product.

Errors and Omissions

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



Safety Precautions

Definition of Precautionary Information





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

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

The following notation is used.

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

Symbols

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

WARNING

WARNING

Emergency stop circuits, interlock circuits, limit circuits, and similar safety measures must be provided in external control circuits.



Check the user program, data, and parameter settings for proper execution before you use them for actual operation.



Perform the test run by holding an emergency stop switch in hand or otherwise prepare for rapid motor operation in an application to control the motor.

Also perform the test run by using the parameters for which the motor does not rapidly accelerate or decelerate before you gradually adjust the parameters.



In the FB/FUN of this library, some values for Input variables and Input/Output variables are "System reserved" variables. For "System reserved" variables, please set them in the range, format and timing specified in this manual. If they are not set as specified, there is a possibility that the operation of the control target may be incorrect.



CAUTIONS

Caution

Read all related manuals carefully before you use this library.



The Sysmac Library and manuals are intended for and assumed to be used by the personnel indicated in the "Intended Audience" section of this manual, or those under the supervision of those personnel.



You must confirm that the user program and parameter values are appropriate to the specifications and operation methods of the devices.



The sample programming shows only the portion of a program that uses the function or function block from the library.



When you use the actual device, include user programming for device safety instructions, interlocks, I/O with other devices, and other control procedures.



Understand the contents of sample programming before you use the sample programming and create the user program.



Create a user program that will produce the intended device operation.



Precautions for Correct Use

Using the Library

- Specify the values of the input parameters within the valid ranges.
- In a function or function block with an Enabled output variable, if the value of Enabled is FALSE, do not use the processing result of the function or function block as a command value to the control target.
- For FUN with a return value of *Out*, if the value for *Out* is one that indicates an error, do not use the calculation result of FUN as the command value to the control target.
- In the function block with *Execute*, do not perform re-execution by the same instance. The values output by the FB returns to the initial values.
- For FB and FUN with output variable *Error* or return value *Out*, when *Error* is TRUE, or return value *Out* is a value that indicated an error, do not use the calculation result of FB or FUN as the command value to the control target.
- The multi-execution (buffer mode) cannot be performed in the Sysmac Library.

Related Manuals

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

Manual name	Man. No.	Model	Application	Description
NX-series CPU Unit Hardware User's Manual	W535	NX701-□□□□	Learning the basic specifications of the NX701 CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NX701 system is provided along with the following information on the CPU Unit. <ul style="list-style-type: none"> • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection
NX-series NX102 CPU Unit Hardware User's Manual	W593	NX102-□□□□	Learning the basic specifications of the NX102 CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NX102 system is provided along with the following information on the CPU Unit. <ul style="list-style-type: none"> • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection
NX-series NX1P2 CPU Unit Hardware User's Manual	W578	NX1P2-□□□□	Learning the basic specifications of the NX1P2 CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NX1P2 system is provided along with the following information on the CPU Unit. <ul style="list-style-type: none"> • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection

Manual name	Man. No.	Model	Application	Description
NJ-series CPU Unit Hardware User's Manual	W500	NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning the basic specifications of the NJ-series CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NJ-series system is provided along with the following information on the CPU Unit. <ul style="list-style-type: none"> • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection
NY-series IPC Machine Controller Industrial Panel PC Hardware User's Manual	W557	NY532-□□□□	Learning the basic specifications of the NY-series Industrial Panel PCs, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NY-series system is provided along with the following information on the Industrial Panel PC. <ul style="list-style-type: none"> • Features and system configuration • Introduction • Part names and functions • General specifications • Installation and wiring • Maintenance and inspection
NY-series IPC Machine Controller Industrial Box PC Hardware User's Manual	W556	NY512-□□□□	Learning the basic specifications of the NY-series Industrial Box PCs, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided.	An introduction to the entire NY-series system is provided along with the following information on the Industrial Box PC. <ul style="list-style-type: none"> • 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. <ul style="list-style-type: none"> • CPU Unit operation • CPU Unit features • Initial settings • Programming based on IEC 61131-3 language specifications

Manual name	Man. No.	Model	Application	Description
NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Software User's Manual	W558	NY532-□□□□ NY512-□□□□	Learning how to program and set up the Controller functions of an NY-series Industrial PC.	The following information is provided on the NY-series Controller functions. <ul style="list-style-type: none"> • Controller operation • Controller features • Controller settings • Programming based on IEC 61131-3 language specifications
NJ/NX-series Instructions Reference Manual	W502	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning detailed specifications on the basic instructions of an NJ/NX-series CPU Unit.	The instructions in the instruction set (IEC 61131-3 specifications) are described.
NY-series Instructions Reference Manual	W560	NY532-□□□□ NY512-□□□□	Learning detailed specifications on the basic instructions of an NY-series Industrial PC.	The instructions in the instruction set (IEC 61131-3 specifications) are described.
NJ/NX-series CPU Unit Motion Control User's Manual	W507	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning about motion control settings and programming concepts.	The settings and operation of the CPU Unit and programming concepts for motion control are described.
NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Motion Control User's Manual	W559	NY532-□□□□ NY512-□□□□	Learning about motion control settings and programming concepts of an NY-series Industrial PC.	The settings and operation of the Controller and programming concepts for motion control are described.
NJ/NX-series Motion Control Instructions Reference Manual	W508	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning about the specifications of the motion control instructions.	The motion control instructions are described.
NY-series Motion Control Instructions Reference Manual	W561	NY532-□□□□ NY512-□□□□	Learning about the specifications of the motion control instructions of an NY-series Industrial PC.	The motion control instructions are described.

Manual name	Man. No.	Model	Application	Description
NJ/NX-series CPU Unit Built-in EtherCAT® Port User's Manual	W505	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Using the built-in EtherCAT port on an NJ/NX-series CPU Unit.	Information on the built-in EtherCAT port is provided. This manual provides an introduction and provides information on the configuration, features, and set-up.
NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Built-in EtherCAT® Port User's Manual	W562	NY532-□□□□ NY512-□□□□	Using the built-in EtherCAT port in an NY-series Industrial PC.	Information on the built-in EtherCAT port is provided. This manual provides an introduction and provides information on the configuration, features, and set-up.
NJ/NX-series CPU Unit Built-in EtherNet/IP™ Port User's Manual	W506	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Using the built-in EtherNet/IP port on an NJ/NX-series CPU Unit.	Information on the built-in EtherNet/IP port is provided. Information is provided on the basic setup, tag data links, and other features.
NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Built-in EtherNet/IP™ Port User's Manual	W563	NY532-□□□□ NY512-□□□□	Using the built-in EtherNet/IP port in an NY-series Industrial PC.	Information on the built-in EtherNet/IP port is provided. Information is provided on the basic setup, tag data links, and other features.
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.
Vision System FH/FZ5 Series User's Manual	Z365	FH-□□□□ FH-□□□□-□□ FZ5-□□□□ FZ5-□□□□-□□ FZ5-□□□□ FZ5-□□□□-□□	Learning how to use the FH/FZ5-series Vision Systems.	Describes the software functions, setup and operating methods required for using the FH/FZ5-series system.
Vision System FH/FZ5 Series User's Manual for Communications Settings	Z342	FH-□□□□ FH-□□□□-□□ FZ5-□□□□ FZ5-□□□□-□□ FZ5-□□□□ FZ5-□□□□-□□	Learning how to connect FH/FZ5-series Vision Systems	The functions, settings, and communications methods to communicate with FH/FZ5-series Vision Systems from a PLC or other external device are described.
Vision System FH/FZ5 Series Processing Item Function Reference Manual	Z341	FH-□□□□ FH-□□□□-□□ FZ5-□□□□ FZ5-□□□□-□□ FZ5-□□□□ FZ5-□□□□-□□	Learning detailed specifications of each processing item function in the Vision System.	Describes the software functions, settings, and operations for using the FH/FZ5-series.

Revision History

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

Cat. No.	W608-E1-02
-----------------	-------------------

↑
Revision code

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

Sections in this Manual

1	Sysmac Library Usage Procedure	1
2	Visual Feedback Alignment Library	2
3	Common Specifications of Function Blocks	3
4	Individual Specifications of FB/FUN	4
A	Appendix	A
I	Index	I

1

Sysmac Library Usage Procedure

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

1-1	Procedure to Use Sysmac Library Installed Using the Installer.....	1 - 2
1-1-1	Using a Newly Installed Sysmac Library.....	1 - 2
1-1-2	Using an Upgraded Sysmac Library	1 - 4
1-2	How to use Sysmac Library in the CPU Unit or Industrial PC	1 - 6

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

This section describes the procedure to use Sysmac Library installed using the installer. There are two ways to use libraries.

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

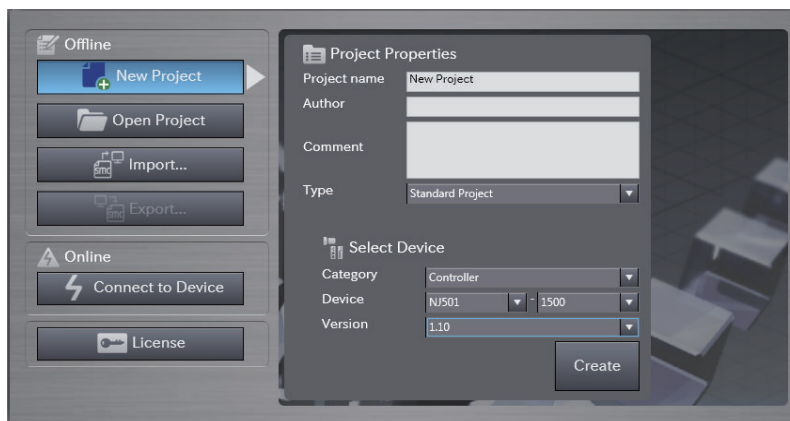


Version Information

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

1-1-1 Using a Newly Installed Sysmac Library

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

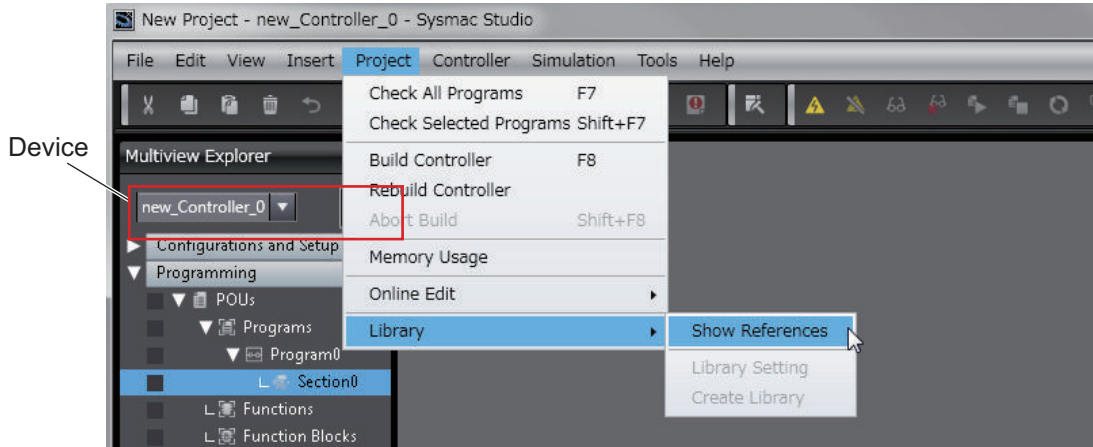


Precautions for Correct Use

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

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

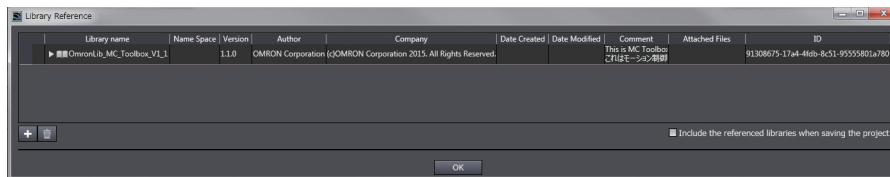
- 2 Select **Project - Library - Show References**.



Precautions for Correct Use

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

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



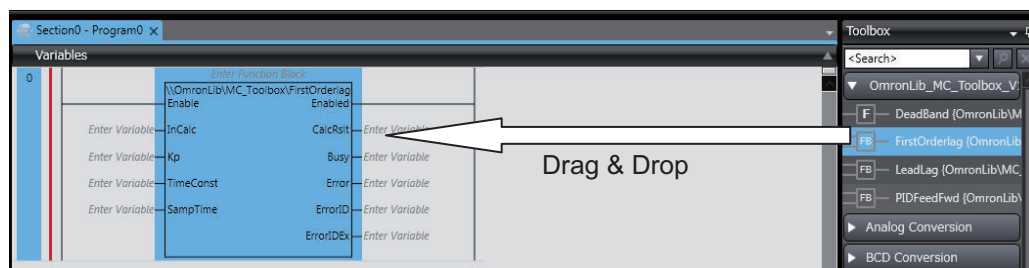
Sysmac Library is read into the project.

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

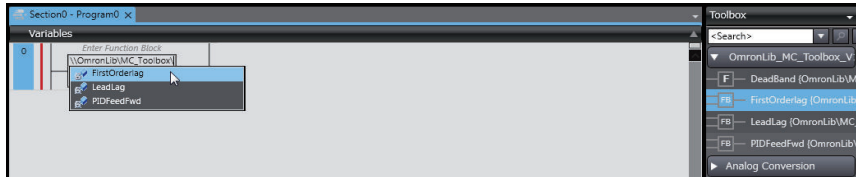
For the procedure for adding and setting libraries in the above screen, refer to *Sysmac Studio Version 1 Operation Manual (W504)*.

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

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

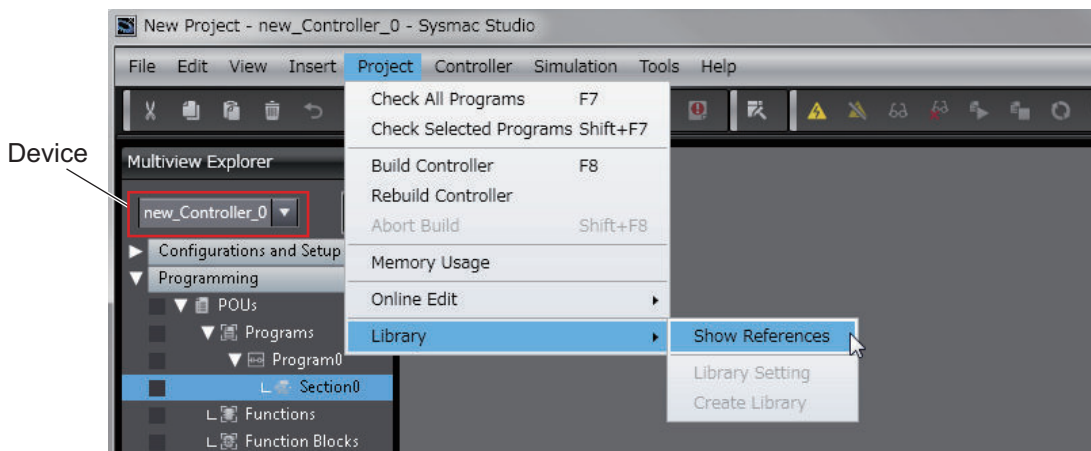


- Right-click the Ladder Editor, select **Insert Function Block** in the menu, and enter the fully qualified name (¥¥namespace¥¥FBname).




1-1-2 Using an Upgraded Sysmac Library

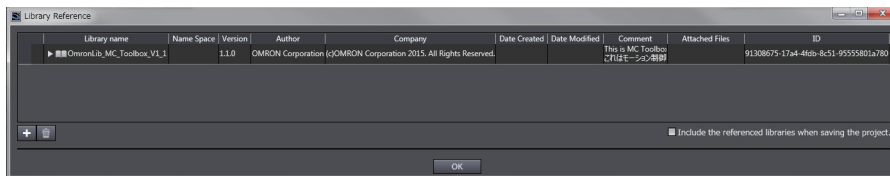
- 1 Start Sysmac Studio and open a project in which any old-version Sysmac Library is included.
- 2 Select **Project - Library - Show References**.



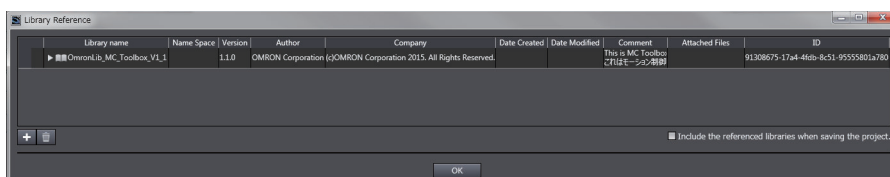
Precautions for Correct Use

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

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



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





Precautions for Correct Use

Upgrade the Sysmac Library version, and then execute All Program Check, and confirm that there are no errors in the Build Window Program Check results.

From the Main Menu, select **Project - All Program Check**.

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

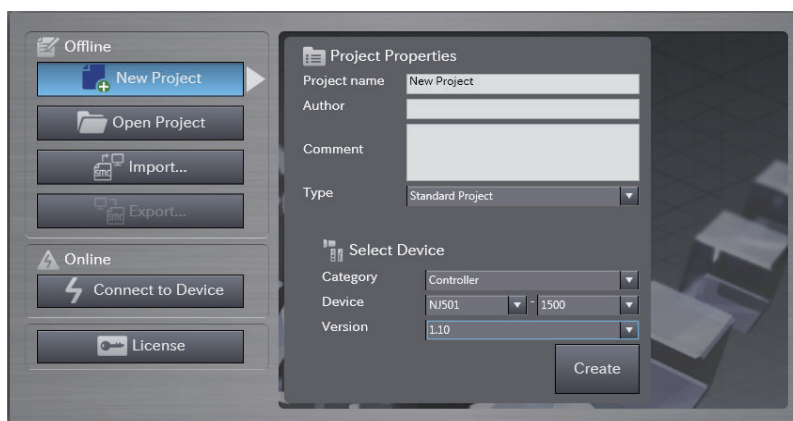
Even when Sysmac Library is not installed on your computer, you can use Sysmac Library by uploading it from the CPU Unit or Industrial PC to your computer.

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

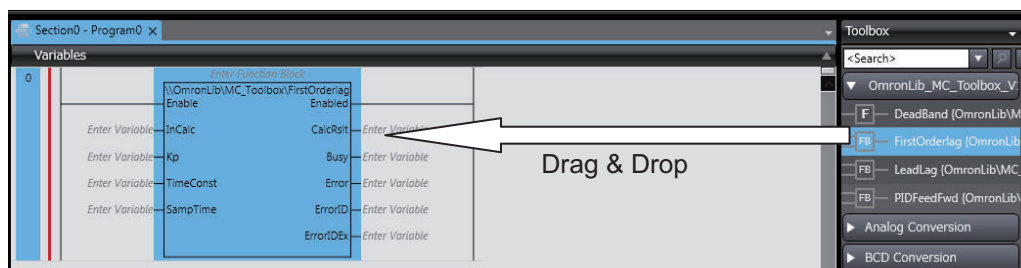
Version Information

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

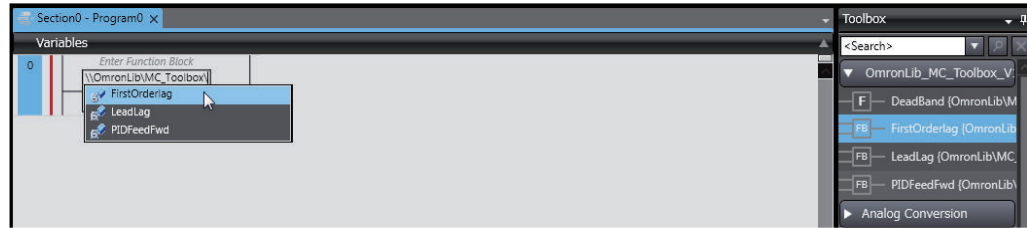
- 1 Start the Sysmac Studio and create a new project in which you want to use Sysmac Library.



- 2 Connect online to the CPU Unit or Industrial PC.
- 3 Upload the POUs in which Sysmac Library is used.
Now, when you select the Ladder Editor or ST Editor, the function blocks and functions included in the Sysmac Library used in the uploaded POUs appear in the Toolbox.
- 4 Insert the Sysmac Library's function blocks and functions into the circuit using one of the following two methods.
 - Select the desired function block or function in the Toolbox and drag and drop it onto the Ladder Editor.



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



Precautions for Correct Use

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

2

Visual Feedback Alignment Library

An explanation of the common specifications for each Function (FUN) and Function Block (FB) in the Visual Feedback Alignment Library.

2-1	Terms	2 - 2
2-2	Overview	2 - 3
2-2-1	Alignment Application	2 - 3
2-2-2	Visual Feedback	2 - 3
2-3	Restrictions and Limitations	2 - 6
2-4	Hardware Configuration	2 - 7
2-4-1	System Configuration	2 - 7
2-4-2	Supported Stages	2 - 7
2-5	Visual Feedback Alignment Library Flow Chart	2 - 9
2-5-1	Alignment Flow Chart	2 - 9
2-5-2	Generate Calibration Parameter Flow Chart	2 - 11
2-6	Visual Feedback Alignment Library Function Specification	2 - 13
2-6-1	Function Overview	2 - 13
2-6-2	Coordinate Systems	2 - 14
2-6-3	Control Blocks.....	2 - 15
2-6-4	Execute Measurement (Imaging of Workpiece).....	2 - 17
2-6-5	When Vision Sensor NG Measurement Occurs.....	2 - 17

2-1 Terms

Terms used in this document and their meanings are as follows.

Term	Meaning
Camera coordinates	X and Y coordinates on an image from a camera, in units of pixels, where the X coordinate is in the horizontal direction and the Y coordinate is in the vertical direction starting from the upper left of the image. The unit of rotation direction on the XY plane is degree (°).
Reference mark position	Coordinates for a mark on a reference workpiece object when performing alignment control.
Calibration	Control and calculations for calculating calibration parameters.
Calibration parameter	The conversion coefficient used for coordinate conversion (converting camera coordinates to stage coordinates).
Measurement	The vision sensor capturing an image of an object and outputting the measurement position/angle of the object.
Measurement mark position	The coordinates for a mark on a workpiece object to be measured for the purpose of alignment control.
Coordinate conversion	Conversion of camera coordinates to stage coordinates using calibration parameters.
Axis coordinates	Rotation coordinates specific to each axis, or linear motion coordinates. These are in units called Axis Instruction Units.
Axis Instruction Unit	Units representing the position and distance which are used by the motion control function of the controller. Value converted from position units and electronic gears.
Stage coordinates	Stage-specific Cartesian coordinates. The target position for moving the stage is calculated by the position of the stage coordinates. These are in units called Axis Instruction Units. The unit of rotation direction on the Cartesian plane is degree (°).
Visual Feedback	A system that performs feedback control using position information acquired from the vision sensor as its input.

2-2 Overview

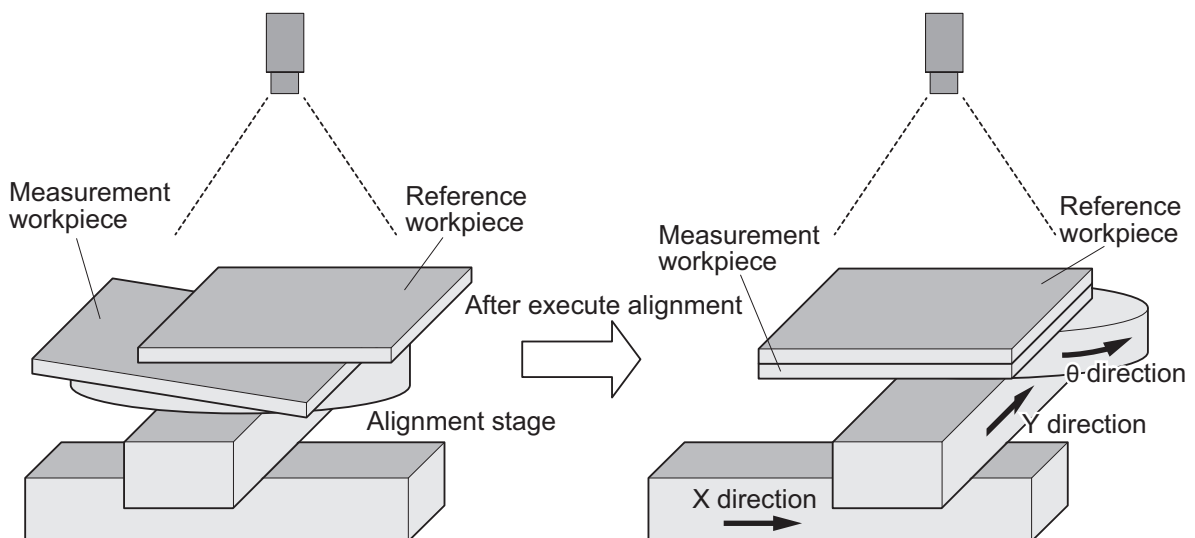
The Visual Feedback Alignment Library is a set of software function components for alignment applications employing visual feedback.

2-2-1 Alignment Application

The alignment application is a method of detecting the position/angle of a workpiece object to be measured (hereinafter, referred to as "measurement workpiece") with a vision sensor in reference to an object placed at an arbitrary position/angle on the alignment stage (hereinafter, referred to as "stage"). By moving the alignment stage based on the above information, it controls the positioning so that the measurement position/angle matches with the target position/angle (hereinafter, referred to as "reference position/angle").

The image capture on which to detect the reference position/angle is referred to as the reference workpiece. The following illustration is an example of Position Alignment of two workpieces. In this example, the reference workpiece, which becomes the reference for alignment, does not move even when the stage is moved, while the measurement workpiece is moved and Position Alignment is performed.

In the case of the $XY\theta$, θXY stage, there are three control axes in the θ direction corresponding to the rotation direction on the XY plane in addition to the X direction and the Y direction which are the perpendicular direction in the XY plane.



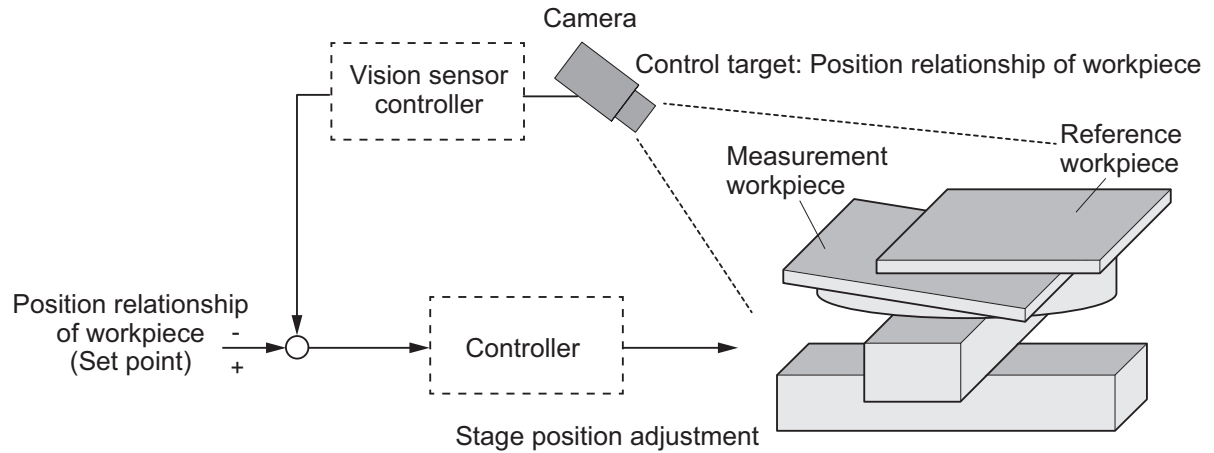
With this library, the stage is controlled using visual feedback of the measurement position/angle of the workpiece object as measured by the vision sensor.

2-2-2 Visual Feedback

When you want the control target to be in a target state, you can apply a feedback control that takes the difference between the value representing the current state of the control target and the value representing its target state and perform the operation according to the difference.

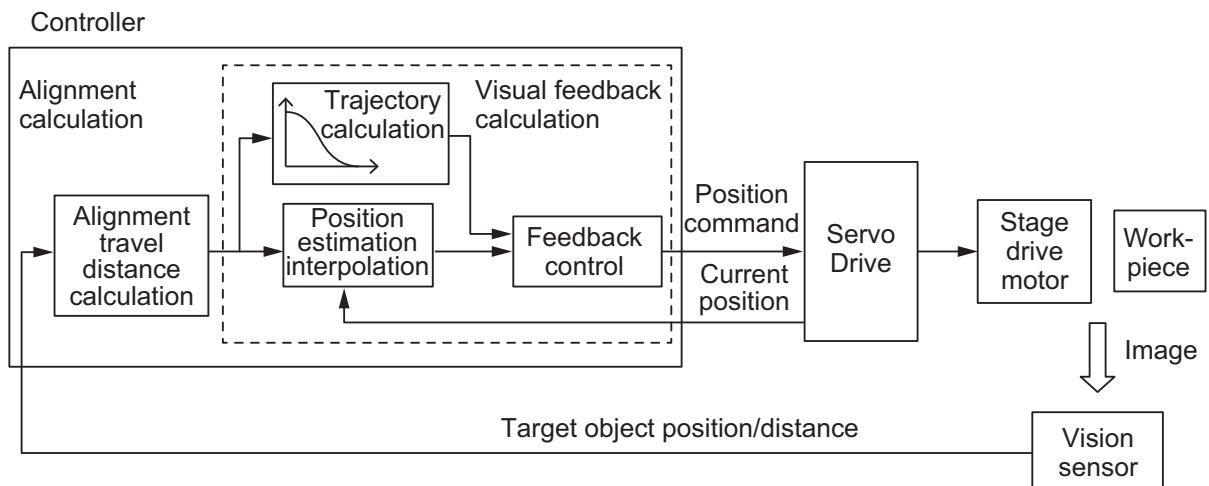
Visual feedback means to obtain the state of the control target from the visual information in feedback control. It uses the vision sensor to acquire and measure the image of the control target, and performs

feedback control based on that information. By using visual information, it is possible to have control according to the state of the factory/actual workpiece.

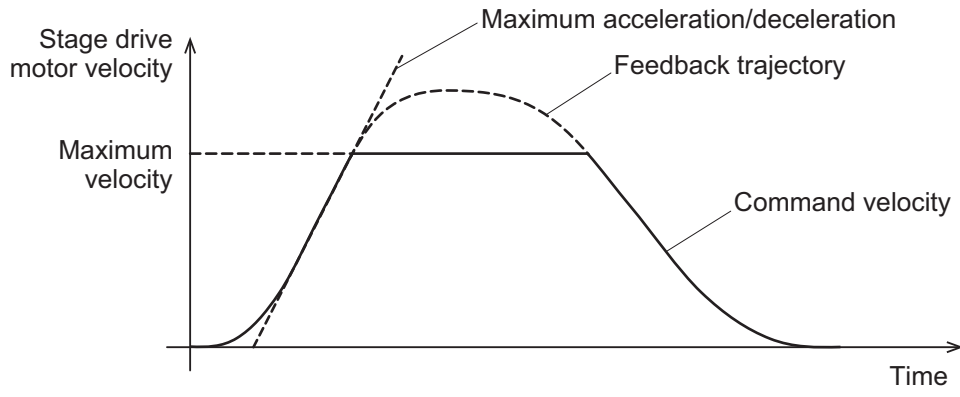


This library is a set of software components (Function Blocks (FB) / Functions (FUN)) that realizes the alignment application with visual feedback. By combining FB / FUN provided with this library, it is possible to build alignment control that applies visual feedback with the user program on the controller.

With this library, calculations to determine the stage travel distance required for alignment are made based on the position/distance of the object measured with the vision sensor. Then it performs visual feedback control by giving control that is proportional to the travel distance.



With the visual feedback control provided by this library, a trajectory of feedback control is generated for the calculated alignment travel distance. The generated feedback trajectory is given as a command to drive the stage with limits defined by the specified maximum acceleration/deceleration and maximum velocity.



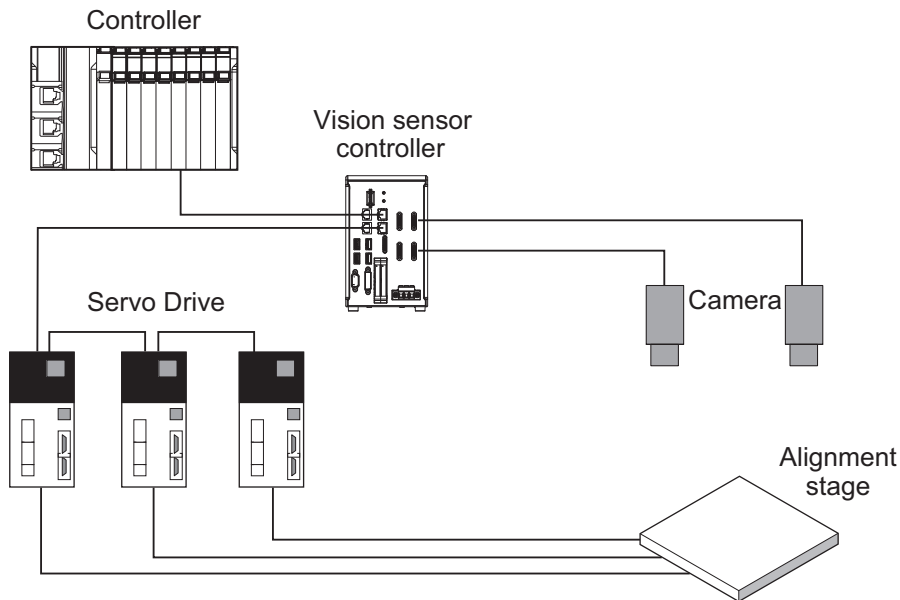
2-3 Restrictions and Limitations

- Because the alignment control provided by this library uses visual feedback control, it is necessary that the measurement position and reference position are always both within the measurable range of the vision sensor.
- Regarding the Image mode of the vision sensor: Alignment cannot be performed using this library if either Camera Through image is selected for the camera image mode or the Multi-input function is used.
- This library does not do any correction for camera tilt or image distortion caused by the camera lens. To perform this correction if needed, please use in combination with the Precise Calibration function provided in the vision sensor.

2-4 Hardware Configuration

2-4-1 System Configuration

The following is an example of the typical system configuration for alignment control using this library.

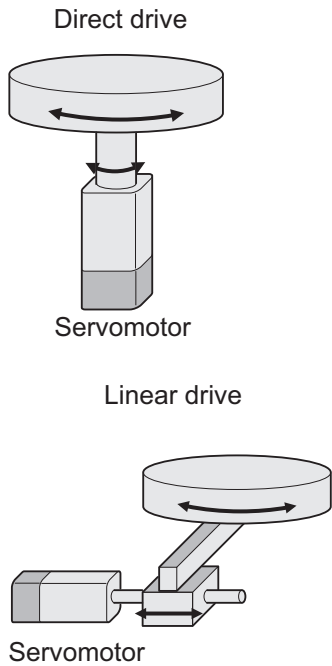


2-4-2 Supported Stages

This library supports the following types of stages.

XY Stage	XY θ Stage	θ XY Stage
X(Y) Stage	X θ (Y θ) Stage	θ X(θ Y) Stage
UVW Stage	UVWR Stage	

Regarding the control of θ axis used in XY θ , θ XY, X θ (Y θ), θ X(θ Y) stage types, both direct drive (the drive system by which the θ axis rotation is aligned to the rotary axis of a Servomotor) and Linear drive (the drive system by which rotation control of the θ axis is done by linear movement) are supported.



Precautions for Correct Use

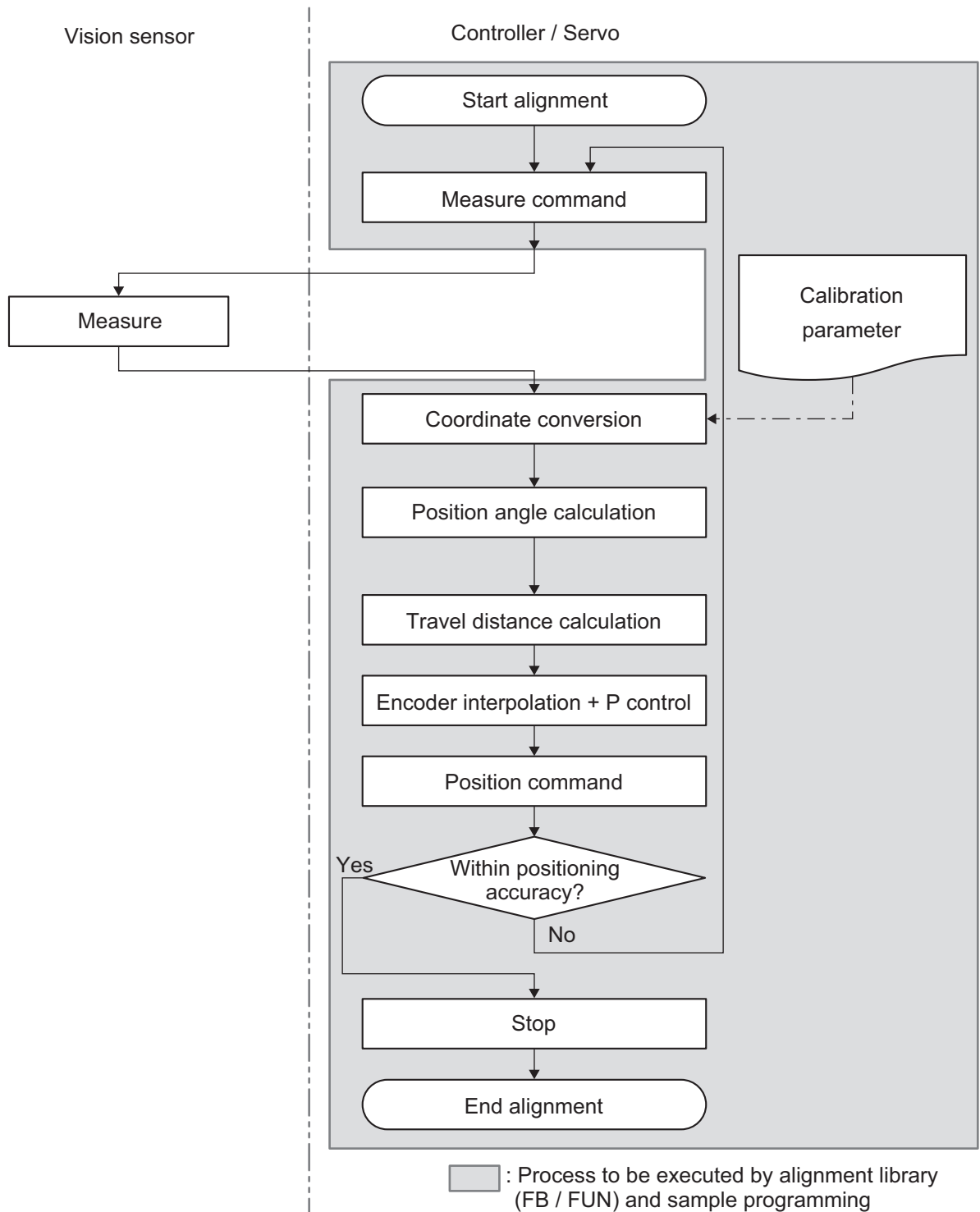
The θ axis is the rotary axis, but it does not support multi-rotation. Therefore, when using it in Linear Mode or Rotary Mode, it is necessary to set the Count Mode of the axis parameter so that it does not rotate multiple times. This can be done using the various software limits. The UVW stage and the UVWR stages correspond to two fulcrum types: direct movement and rotation. Refer to *UVWR Stage Parameter (sUVWR_STAGE_PARAMS)* on page 4 - 5 for detail.

2-5 Visual Feedback Alignment Library Flow Chart

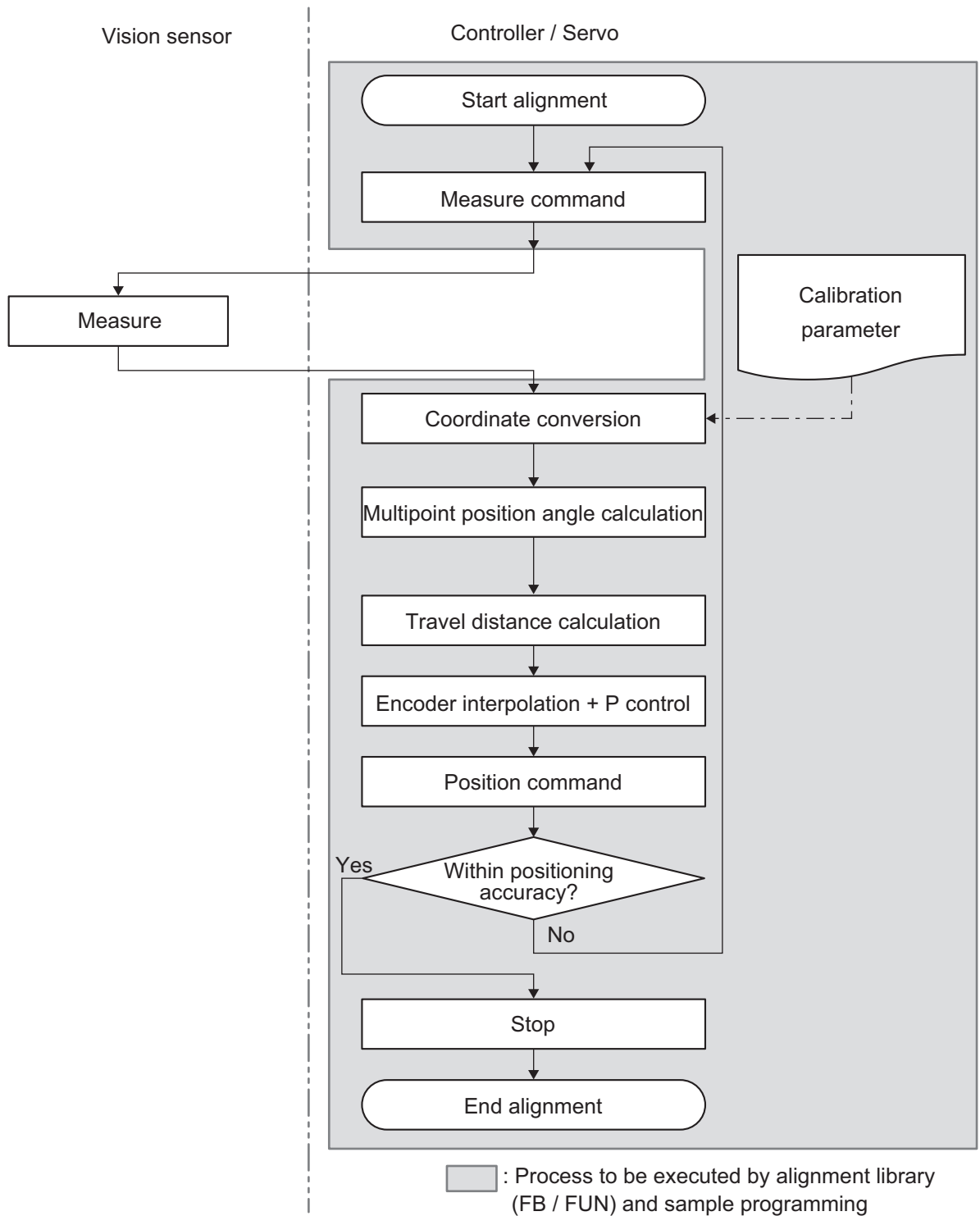
2-5-1 Alignment Flow Chart

The following is a flow chart showing how alignment is done with the use of this library.

- Alignment whereby one measurement position/angle is aligned to a corresponding reference position/angle

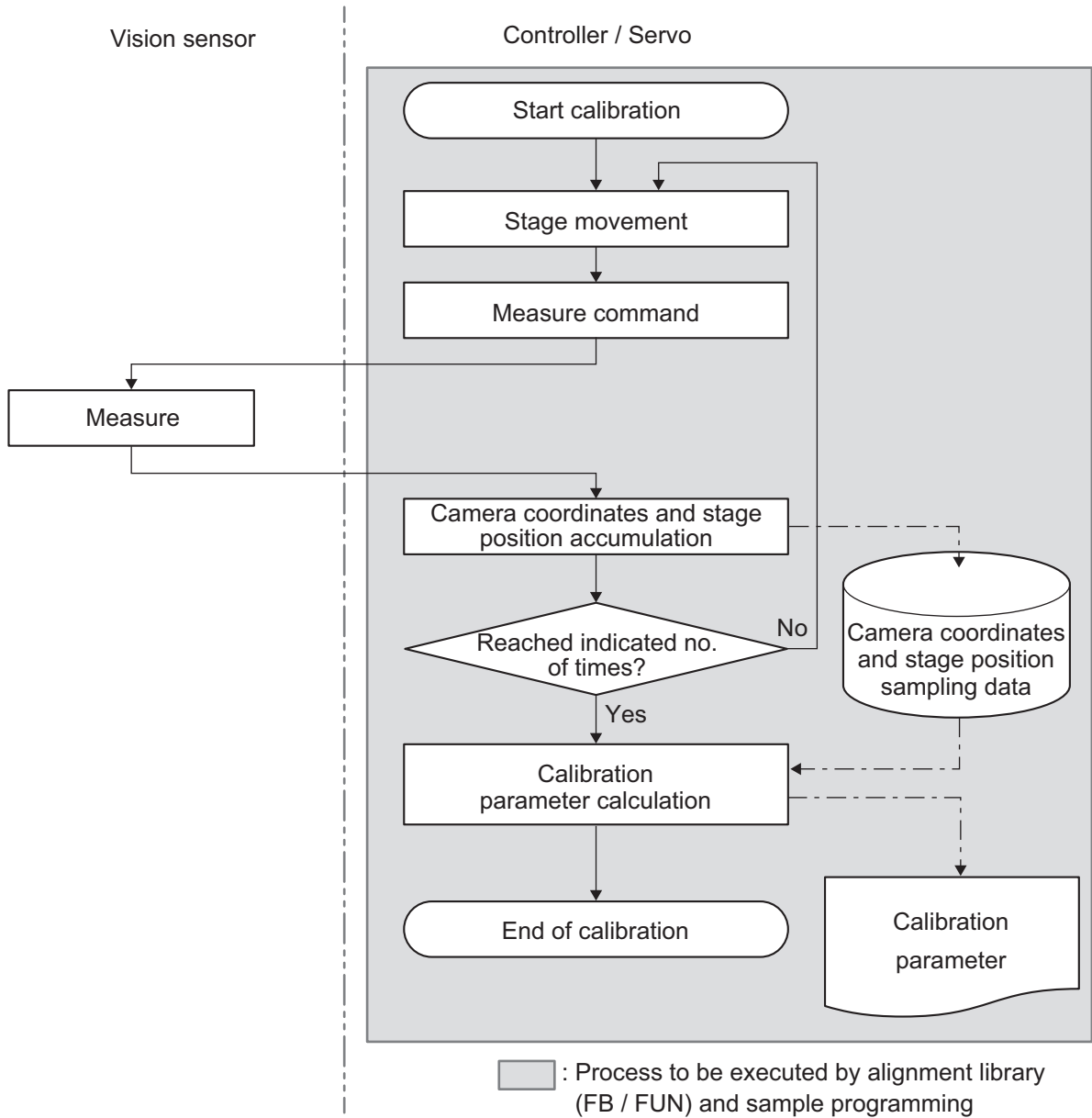


- Alignment whereby measurement positions and reference positions for multiple points are aligned. The difference from 1 point is that the part to perform coordinate conversion and multipoint position angle calculation differs based on the measurement result of multiple points.



2-5-2 Generate Calibration Parameter Flow Chart

The following is a flow chart showing how calibration parameters are generated using this library.



2-6 Visual Feedback Alignment Library Function Specification

2-6-1 Function Overview

The functions needed for alignment control can be classified into the following three major items.

- (1) Position measurement of an *alignment mark on a workpiece* or *specific features of the workpiece such as a corner* by a vision sensor.
- (2) Travel distance calculation of the stage for aligning the workpiece to a predetermined position (reference position) based on the measurement result.
- (3) Position control of the stage according to the travel distance.

Also in (1), the following function is required to conform to the alignment control in order to convert the unit of dimension in the image captured by the camera into the unit of dimension in stage operation.

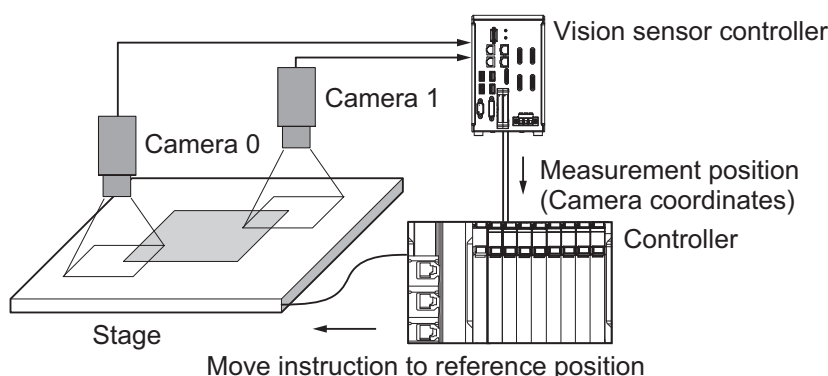
- (4) The *Calibration* function aligns the stage coordinates and the camera coordinates.

In general, the vision sensor controller is responsible for (1), (2) and (4) and the controller gives a command to the motor driver that drives the stage (3) according to the output from the vision sensor controller.

In this library, the vision sensor controller is responsible for only the position measurement of (1) and (4), and the controller performs the various calculations of (2), (3) and (4) as well as Stage drive.

- Alignment (workpiece alignment)

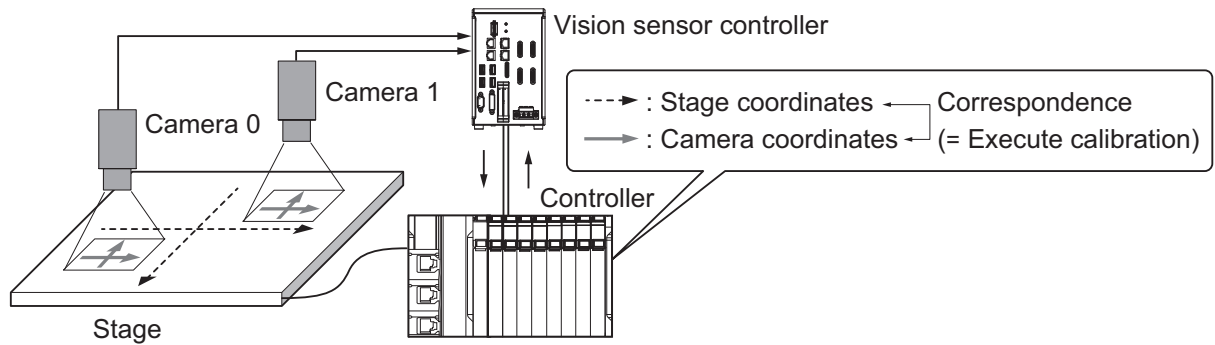
Measure features of a workpiece object to get a measurement position in camera coordinates. The features to use include *alignment marks on workpiece* and *specific features of the workpiece such as corners*. Next, using the calibration parameter calculated by calibration, convert the measurement position to the stage coordinates. Finally, calculate the axis travel distance required to align the measurement position with the reference position, and apply the control to the stage position.



- Calibration

Because coordinate systems differ between camera and stage, prior to measurements, calculations must be done to make the coordinate systems compatible with each other. That process is referred to as *Calibration*.

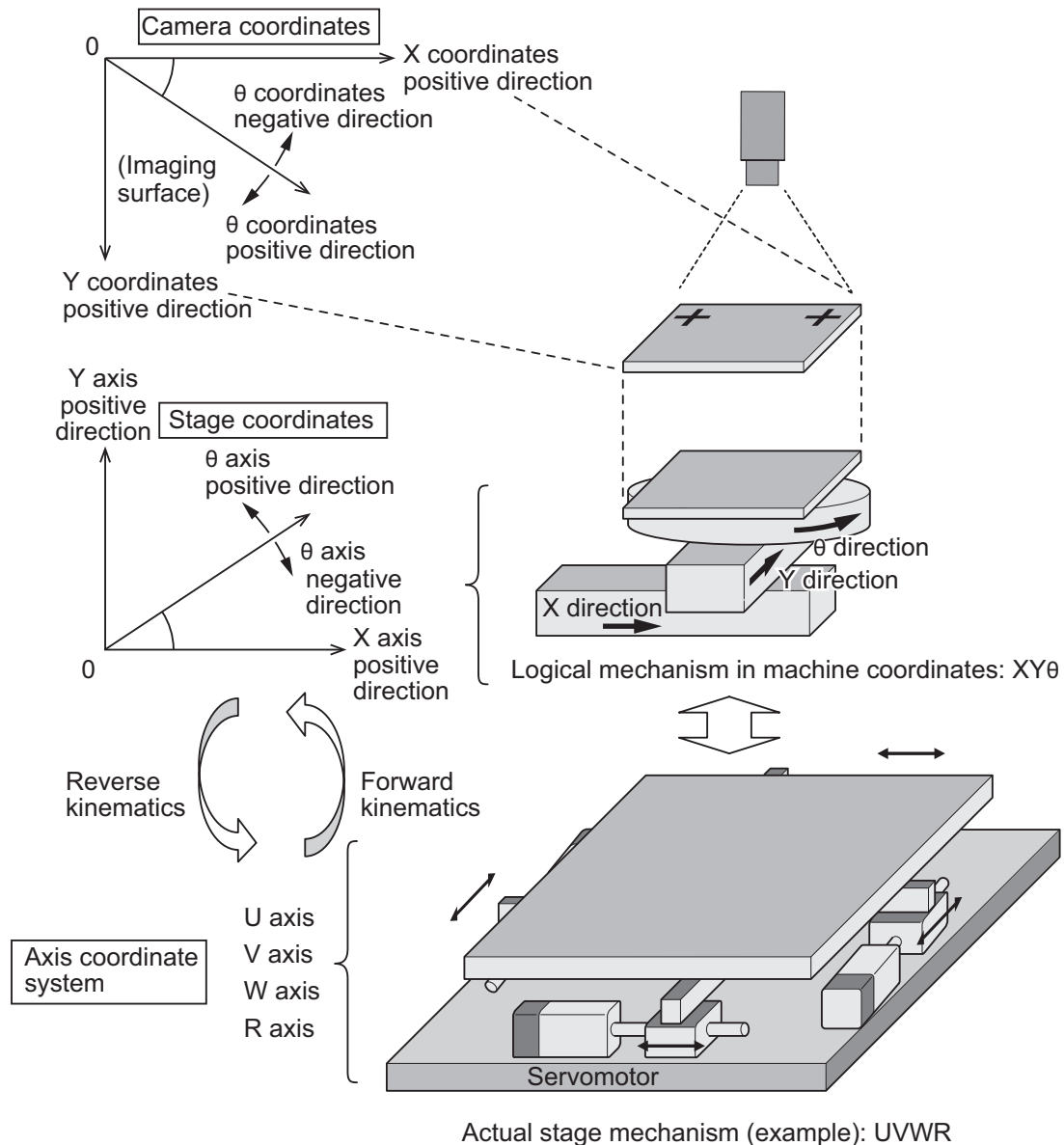
The controller moves the stage and calculates the calibration parameter by continually repeating a set operation pattern of *move workpiece* → *position measurement*.



2-6-2 Coordinate Systems

There are three types of coordinate systems handled in the alignment control targeted by this library. In this library, calculations of the alignment application are all converted into XYθ stage. For the control output of this computational XYθ stage (hereinafter referred to as the virtual XYθ stage), kinematics transformation according to the actual stage mechanism is performed and the stage mechanism is controlled.

Type	Overview	Definition	Unit
Camera coordinates	The Coordinate System for camera imaging	The upper left corner of the captured image is set to be zero, the X coordinate is the positive direction in the horizontal direction, the Y coordinate is the positive direction in the vertical direction, the θ coordinate (rotation angle) is the positive direction in the clockwise direction on the XY plane.	X,Y: Pixels θ : ° (degree)
Stage coordinates	The Coordinate System on a stage	Zero is the home of the stage and from it is formed the XY plane in the right-handed system (Main system) of Cartesian coordinates with respect to the direction of movement of the stage. In the stage coordinates, location management is performed according to the virtual XYθ stage, regardless of the mechanism of the actual stage.	X,Y: mm θ : ° (degree)
Axis Coordinate System (ACS)	The Coordinate System of the stage drive axis	The position coordinates of the individual drive axes determined by the stage mechanism are determined with respect to the position of the stage coordinates. The position coordinates of particular stage coordinates are converted into the position (rotation distance) of each axis by solving the reverse kinematics with respect to the mechanism. Conversely, when each axis position (rotation distance) is obtained, the stage position determined by that position becomes clear by solving the forward kinematics.	Each axis: Depends on stage mechanism




2-6-3 Control Blocks

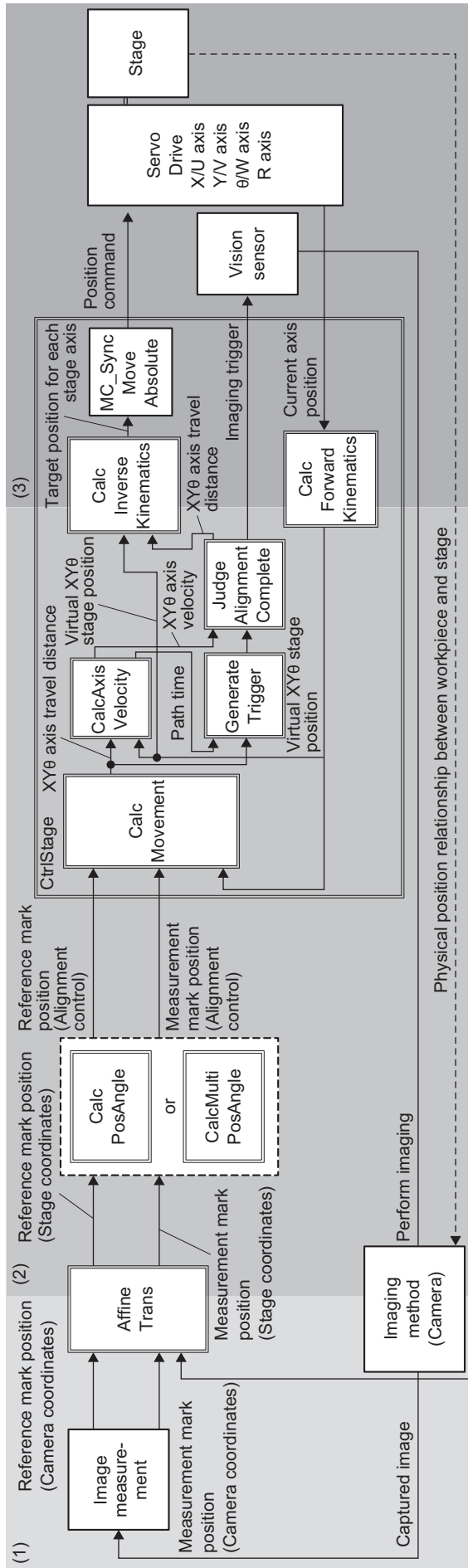
In order to perform the functions (1) to (4) that are provided in *2-6-1 Function Overview* on page 2 - 13, the alignment control targeted by this library consists of control blocks as shown below. The parts indicated by the numbers (1) to (4) are the control elements for performing the corresponding function.

The control block (CtrlStage) in (3) is the area of alignment control specifically targeted for the visual feedback provided by this library. The control configuration for everything outside of this area is otherwise equivalent to existing alignment control.

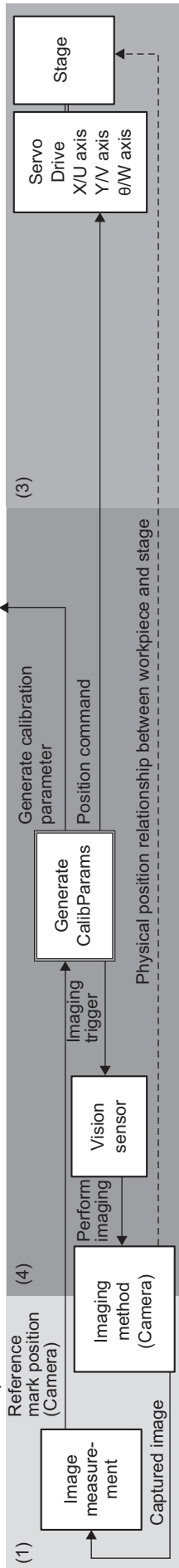
Also, with respect to each control block, depending on the coordinate system (unit of data) to which the handled information belongs and on which target attribute the information is classified, it is possible to distinguish it by the lower row of the figure, the coordinate system classification and the information classification.

 : POU provided in this library (Function Block / Function)

Block diagram of alignment control function



Generate calibration parameter function block



Coordinate system classification

Camera coordinates

Stage coordinates¹⁾

Position coordinates of alignment mark

Stage position coordinates

Axis position information

Axis coordinate system

¹⁾ With this library, the axis travel distance and axis velocity for alignment control are calculated based on the virtual XYθ stage. Therefore, each axis position of the stage is converted into position coordinates on the virtual XYθ stage coordinates by CalcForwardKinematics (forward kinematics calculation), and the axis travel distance / axis velocity on the calculated virtual XYθ stage coordinates is converted by CalcInverseKinematics (reverse kinematics calculation) into the target position of the stage axis. With this method, it is possible to perform alignment control with the same module configuration for all the stages described in 2-4-2 Supported Stages.

2-6-4 Execute Measurement (Imaging of Workpiece)

The vision sensor takes an image of the measurement workpiece or the reference workpiece and measures it with a "Search" type processing item to get the measurement position/angle and reference position/angle in camera coordinates.

These measurements in camera coordinates are the input to the controller (library).

Even if a "reference workpiece" is not imaged, but the reference position is set to a fixed position, the reference position is entered into the controller (library) with camera coordinates.

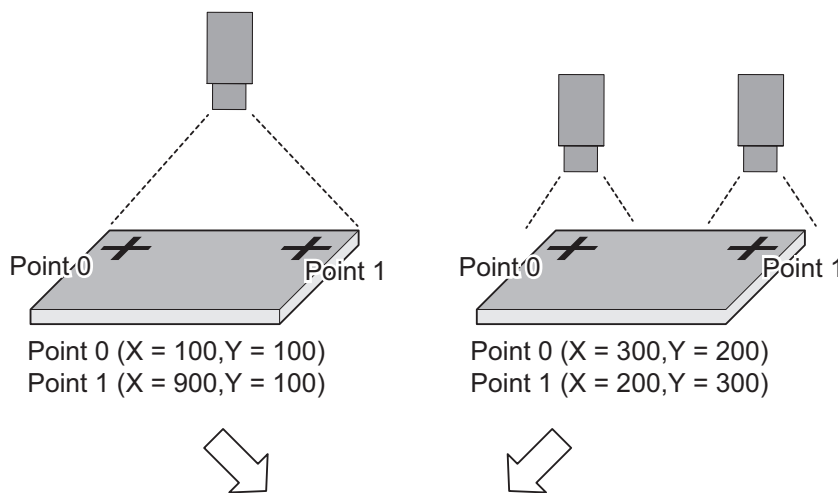
This library supports the use of 1 point to 4 points to specify the reference position and the measurement position.

Since the vision sensor does not perform coordinate conversion (outputs the camera coordinates as they are), cases where multiple reference positions or measurement positions are detected by one camera and cases where multiple cameras detect reference positions or measurement positions are handled in the same way.

Example: When the reference position or measurement position is determined by 2 points

When detecting 2 points with 1 camera

When detecting 2 points with 2 cameras



Handled similarly on the controller (library) side

2-6-5 When Vision Sensor NG Measurement Occurs

When the measurement result from the vision sensor is an NG measurement result (an output of TRUE for Overall Judgment (TotalJudgment)), the alignment control block ignores the measurement NG and continues control based on the previous measurement position/angle. Even after a measurement NG occurs, a measurement trigger is generated according to the Generate trigger condition. Therefore, when the next measurement is successfully done, alignment control based on the updated measurement position/angle is performed.

The generate calibration parameter (GenerateCalibParams) is aborted (CommandAborted) when the measurement result of the vision sensor is an NG measurement.

3

Common Specifications of Function Blocks

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

3-1	Common Variables	3 - 2
3-1-1	Definition of Input Variables and Output Variables	3 - 2
3-1-2	Execute-type Function Blocks	3 - 3
3-1-3	Enable-type Function Blocks	3 - 5
3-2	Precautions	3 - 7
3-2-1	Nesting	3 - 7
3-2-2	Instruction Options	3 - 7
3-2-3	Re-execution of Function Blocks	3 - 7

3-1 Common Variables

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

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

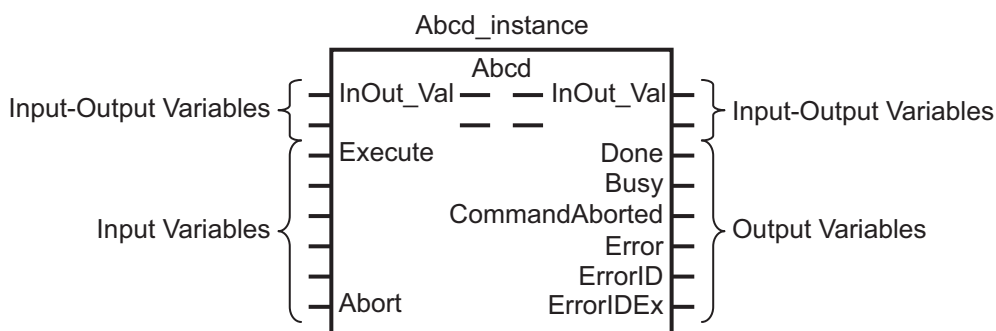
Common input variables and output variables used in functions and function blocks are as follows.

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

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

3-1-2 Execute-type Function Blocks

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

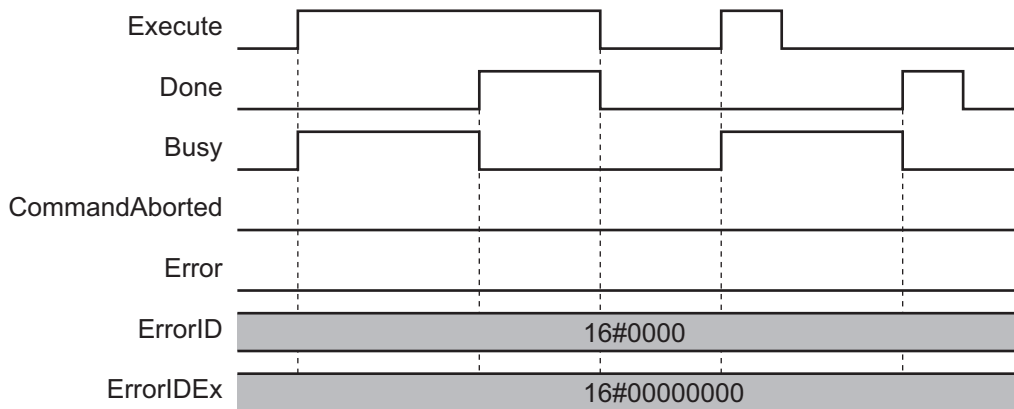


- If Execute is TRUE and Done, CommandAborted, or Error changes to TRUE, Done, CommandAborted, or Error changes to FALSE when Execute is changed to FALSE.
- If Execute is FALSE and Done, CommandAborted, or Error changes to TRUE, Done, CommandAborted, or Error changes to TRUE for only one task period.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

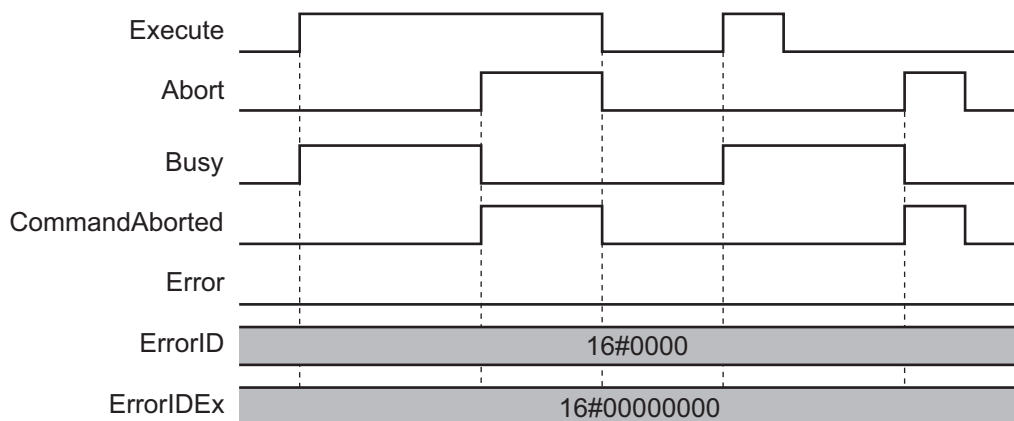
Timing Chart

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

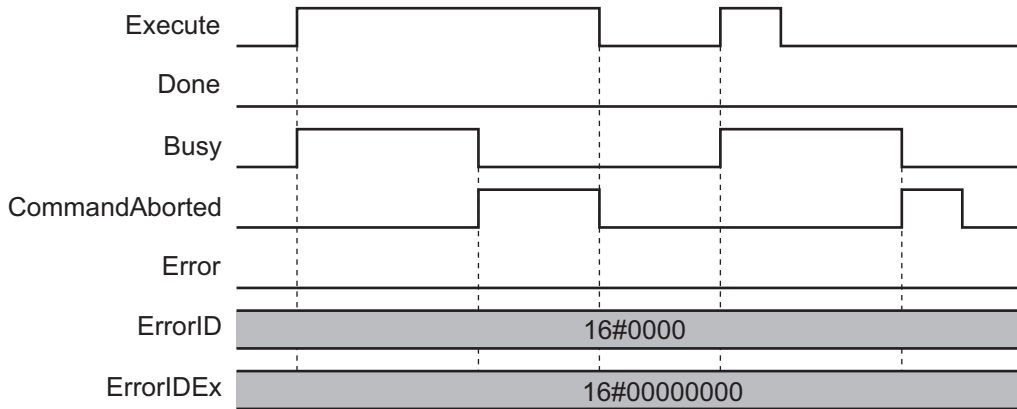
● Normal End



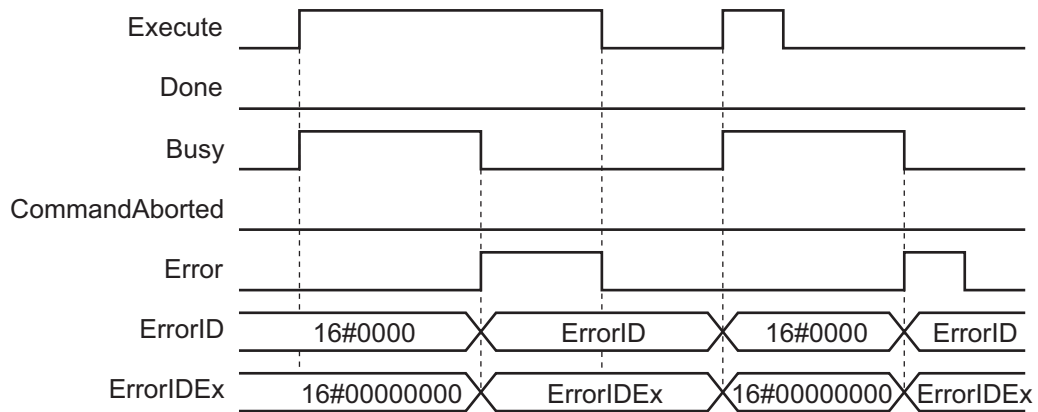
● Canceled Execution



● Aborted Execution

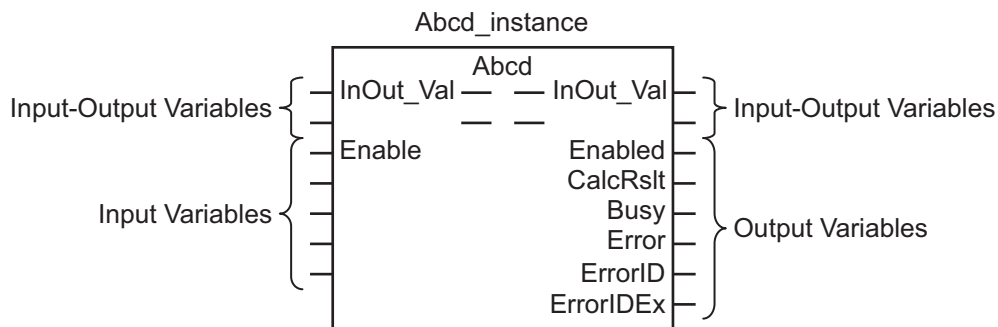


● Errors



3-1-3 Enable-type Function Blocks

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



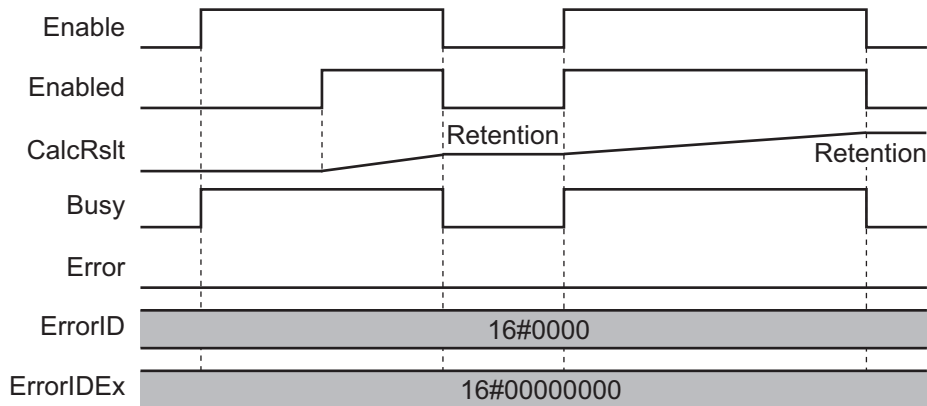
- If an error occurs in the function block, the relevant error code and expansion error code are set in ErrorID (Error Code) and ErrorIDEx (Expansion Error Code). The error codes are retained even after Error changes to FALSE, but ErrorID is set to 16#0000 and ErrorIDEx is set to 16#0000 0000 when Execute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

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

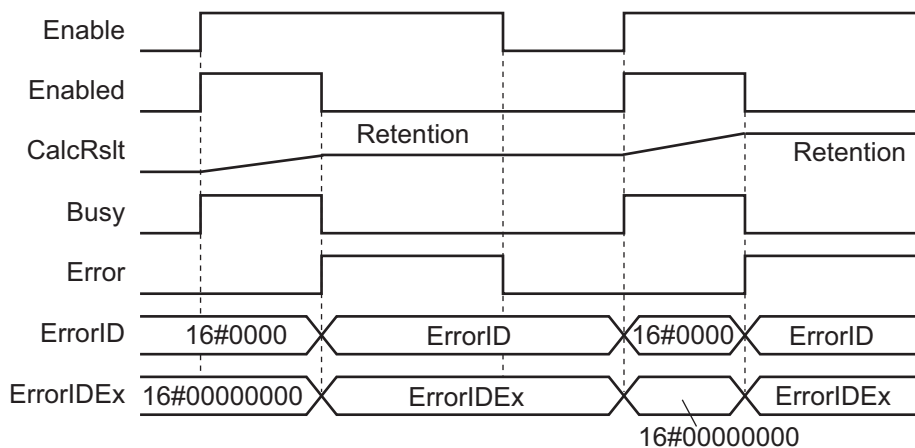
Timing Charts

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

● Normal End



● Errors



3-2 Precautions

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

3-2-1 Nesting

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

Refer to *NJ/NX-series CPU Unit Software User's Manual (Cat. No. W501)* or *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Software User's Manual (Cat. No. W558)* for details on the nesting function block.

3-2-2 Instruction Options

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

3-2-3 Re-execution of Function Blocks

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

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

Refer to *NJ/NX-series CPU Unit Motion Control User's Manual (Cat. No. W507)* or *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Motion Control User's Manual (Cat. No. W509)* for details on re-execution.

4

Individual Specifications of FB/FUN

This section describes the individual specifications for each Function (FUN) and Function Block (FB) in the Visual Feedback Alignment Library.

4

Structure	4 - 2
AffineTrans	4 - 22
CalcPosAngle	4 - 25
CalcMultiPosAngle	4 - 30
CtrlStage	4 - 35
CalcMovement	4 - 45
CalcInverseKinematics	4 - 48
CalcForwardKinematics	4 - 51
CalcAxisVelocity	4 - 54
JudgeAlignmentComplete	4 - 64
GenerateTrigger	4 - 73
GenerateCalibParams	4 - 78
Sample Programming	4 - 90

Structure

The structure of the alignment library is as follows.

XYθ Position (sPOSITION)

Stores the XYθ coordinates.

Variable name	Data type	Description	Valid range	Unit
OmronLib\VF_Alignment\sPOSITION	STRUCT	—	—	—
X	LREAL	X coordinate	Depends on data type.	*1
Y	LREAL	Y coordinate	Depends on data type.	*1
TH	LREAL	Angle	Depends on data type.	° (degree)

- *1. The unit differs depending on which Coordinate System is stored.
 Camera coordinates: Pixels
 Stage coordinates: Axis Instruction Units

Calibration Parameter (sCALIB_PARAMS)

Stores the parameters used for calibration.

For each coefficient, please refer to *Function* on page 4 - 23.

Variable name	Data type	Description	Valid range	Unit
OmronLib\VF_Alignment\sCALIB_PARAMS	STRUCT	—	—	—
A	LREAL	Coefficient A	Depends on data type.	—
B	LREAL	Coefficient B	Depends on data type.	—
C	LREAL	Coefficient C	Depends on data type.	—
D	LREAL	Coefficient D	Depends on data type.	—
E	LREAL	Coefficient E	Depends on data type.	—
F	LREAL	Coefficient F	Depends on data type.	—

Alignment Control Parameter (sALIGNMENT_PARAMS)

Set parameters for alignment control.

Variable name	Data type	Description	Valid range	Unit
OmronLib\VF_Alignment\sALIGNMENT_PARAMS	STRUCT	—	—	—

Variable name	Data type	Description	Valid range	Unit
StageParams	OmronLib\VF_Alignment \sSTAGE_PARAMS	Sets stage parameter.	—	—
KinematicsParams	OmronLib\VF_Alignment \sKINEMATICS_PARAMS	Sets forward/reverse kinematics calculation parameter.	—	—
CalcVirtualMoveParams	OmronLib\VF_Alignment \sCALCVIRTUALMOVE_PARAMS	Sets the parameters for virtual XYθ stage axis position/velocity calculation parameter.	—	—
CalcCurveParams	OmronLib\VF_Alignment \sCALCCURVE_PARAMS	Sets the fifth order trajectory calculation parameter.	—	—
ImagingParams	OmronLib\VF_Alignment \sIMAGING_PARAMS	Set the imaging parameters.	—	—
JudgeParams	OmronLib\VF_Alignment \sJUDGE_PARAMS	Set the Done judgment parameters.	—	—

Stage Parameter (sSTAGE_PARAMS)

Stores stage settings.

Variable name	Data type	Description	Valid range	Unit
OmronLib\VF_Alignment \sSTAGE_PARAMS	STRUCT	—	—	—
StageType	UINT	Stage type 0 : XY 1 : XYθ 2 : θXY 3 : X 4 : Y 5 : Xθ 6 : Yθ 7 : θX 8 : θY 9 : UVW 10 : UVWR	0 to 10	—
AxNo	ARRAY [0..3] OF UINT	Sets the axis number (Ax- is variable.Cfg.AxNo). It is not necessary to set an unused axis. [0]: X/U axis [1]: Y/V axis [2]: θ/W axis [3]: R axis	*1	—
StopDec	ARRAY [0..3] OF LREAL	Deceleration speed at De- celeration stop It is not necessary to set an unused axis. [0]: X/U axis [1]: Y/V axis [2]: θ/W axis [3]: R axis	Positive number or 0	Axis Instruction Unit/s ²

Variable name	Data type	Description	Valid range	Unit
XYTHStageParams	OmronLib \\VF_Alignment \\sXYTH_STAGE_PARAMS	Sets XYθ stage parameter *2.	—	—
UVWRStageParams	OmronLib \\VF_Alignment \\sUVWR_STAGE_PARAMS	Sets UVWR stage parameter *3.	—	—

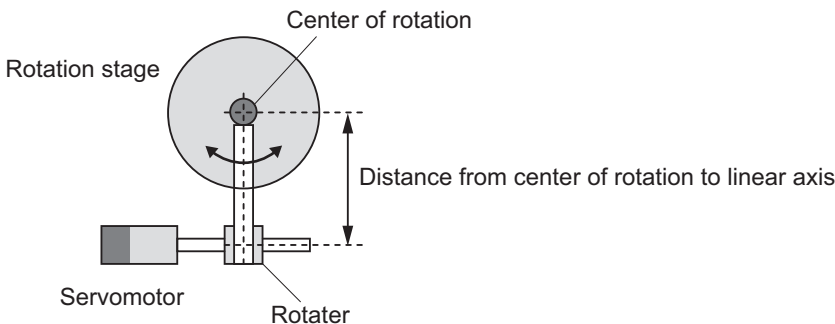
- *1. The number that you can set for axis number is the maximum number of controlled axes. The maximum number of controlled axes varies by controller used. Refer to the manual for the controller you are using.
- *2. For stage types 0 to 8, XYθ stage parameters are used.
- *3. For stage types 9 to 10, UVWR stage parameters are used.

XYθ Stage Parameter (sXYTH_STAGE_PARAMS)

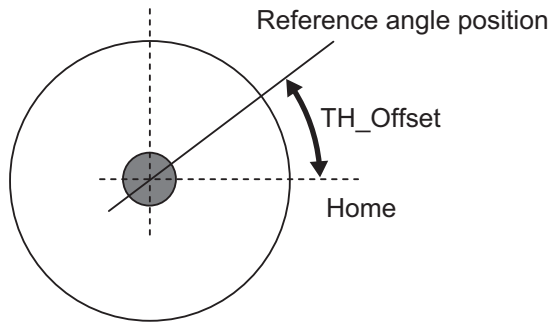
Stores XYθ Stage settings.

Variable name	Data type	Description	Valid range	Unit
OmronLib\\VF_Alignment \\sXYTH_STAGE_PARAMS	STRUCT	—	—	—
Rotation	UINT	Rotation polarity 0 : Positive polarity (In X-axis to Y-axis direction on the stage coordinates system) 1: Negative polarity (In Y-axis to X-axis direction on the stage coordinates system)	0, 1	—
TH_Type	UINT	θ axis type 0: Direct drive 1: Linear drive	0, 1	—
TH_Length	LREAL	Distance *1 from center of rotation to linear axis	Positive number or 0	Axis Instruction Unit
TH_Offset	LREAL	Distance *2 from reference angle (0 °) of θ axis to home	Depends on data type.	° (degree)

- *1. Please set so that the current position of the linear axis is zero when the rotater to which the rotation stage and linear axis are connected is perpendicular to the linear axis.



- *2. When setting the reference angle, if the θ axis position of the reference angle does not match the home, please set that distance.



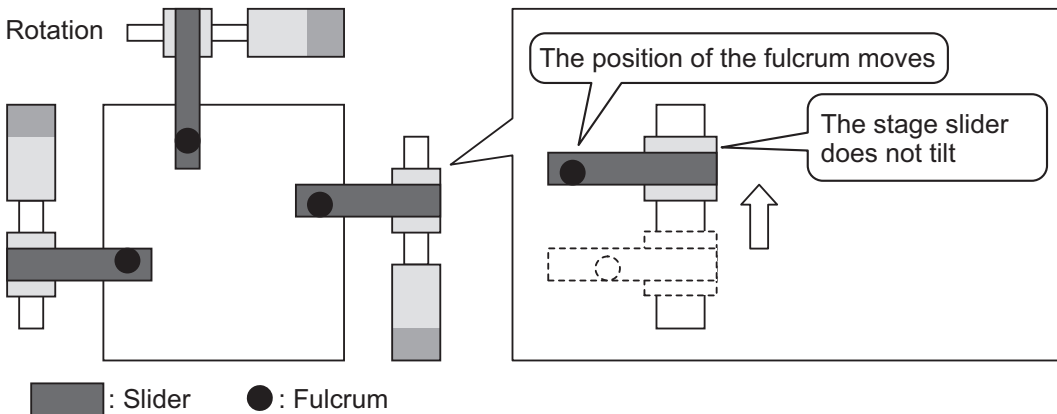
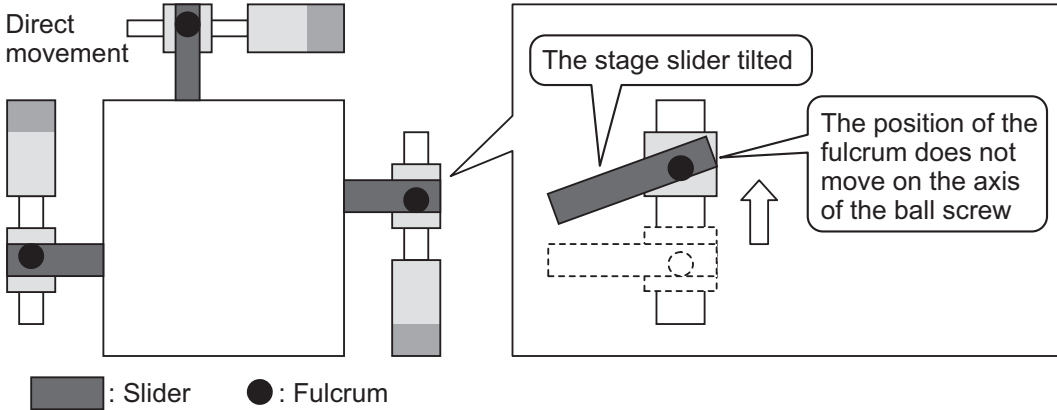
UVWR Stage Parameter (sUVWR_STAGE_PARAMS)

Stores UVWR Stage settings.

Variable name	Data type	Description	Valid range	Unit
OmronLib\VF_Alignment \sUVWR_STAGE_PARAMS	STRUCT	—	—	—
FulcType	UINT	Fulcrum type*1 0: Direct movement 1: Rotation	0, 1	—
SettingMode	UINT	Set the setting method for the position of the UVWR axis fulcrum. 0: Coordinate setting 1: Direct setting (length, angle)	0 to 1	—
RotationCenter_X	LREAL	The X coordinate of the center of rotation in the homing status.*2	Depends on data type.	Axis Instruction Unit
RotationCenter_Y	LREAL	The Y coordinate of the center of rotation in the homing status.*2	Depends on data type.	Axis Instruction Unit
U_Params	OmronLib \VF_Alignment \sUVWR_PARAMS	U axis parameter	—	—
V_Params	OmronLib \VF_Alignment \sUVWR_PARAMS	V axis parameter	—	—
W_Params	OmronLib \VF_Alignment \sUVWR_PARAMS	W axis parameter	—	—

Variable name	Data type	Description	Valid range	Unit
R_Params	OmronLib WF_Align- ment sUVWR_PA RAMS	R axis parameter	—	—

*1. The differences between direct movement and rotation of the fulcrum type are as shown in the figure below.



*2. Coordinates are specified in stage coordinates. Refer to 2-6-2 *Coordinate Systems* on page 2 - 14 for details on stage coordinates.

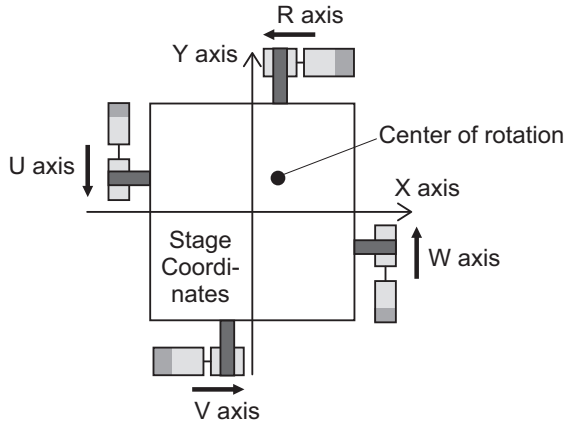
UVWR Axis Parameter (sUVWR_PARAMS)

Stores UVWR axis settings.

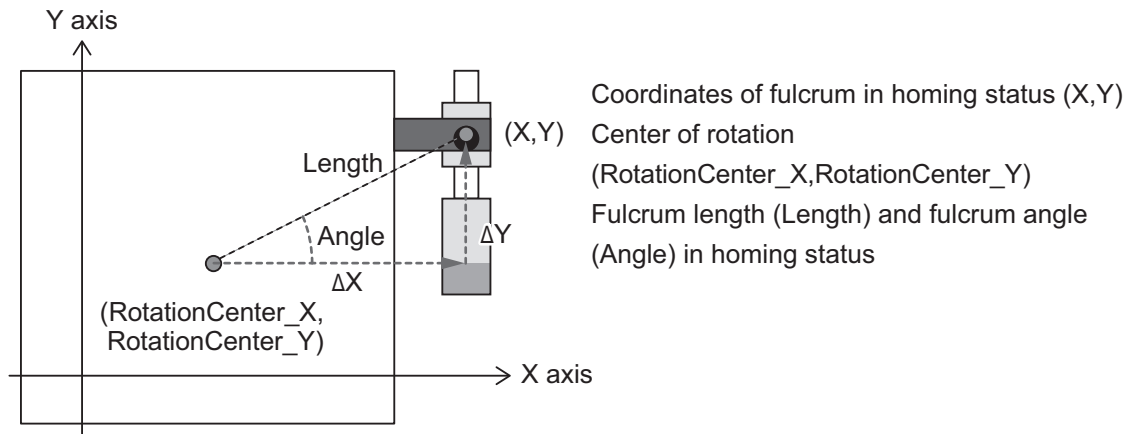
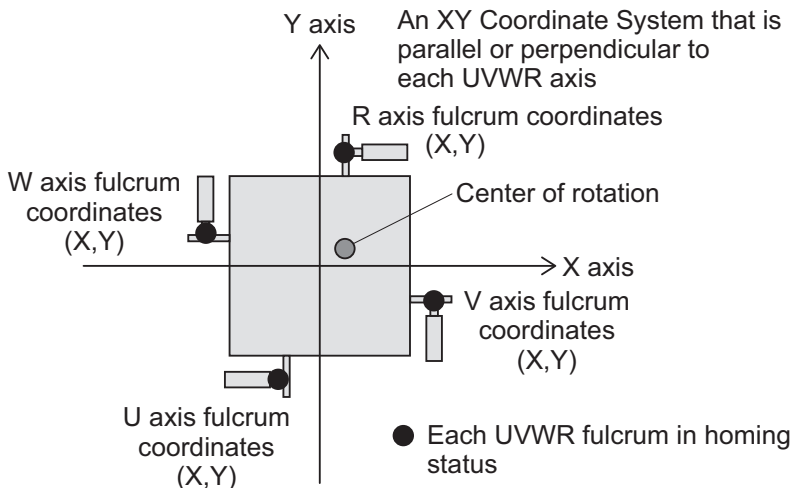
Variable name	Data type	Description	Valid range	Unit
OmronLib\WF_Alignment sUVWR_PARAMS	STRUCT	—	—	—
Dir	UINT	Sets the axis direction. *1 0: X axis positive direction 1: X axis negative direction 2: Y axis positive direction 3: Y axis negative direction	0 to 3	—

Variable name	Data type	Description	Valid range	Unit
X	LREAL	When sUVWR_STAGE_PARAMS.SettingMode is 0: Coordinate setting, please set the X coordinate of the fulcrum. When the Stage Control FB (CtrlStage) is executed, this setting value is converted into the length (Length) of the line segment connecting the center of rotation of the stage and the pivot point and the angle of the fulcrum (Angle). *2	Depends on data type.	Axis Instruction Unit
Y	LREAL	When sUVWR_STAGE_PARAMS.SettingMode is 0: Coordinate setting, please set the Y coordinate of the fulcrum. When the stage control FB (CtrlStage) is executed, this setting value is converted into the length (Length) of the line segment connecting the center of rotation of the stage and the pivot point and the angle of the fulcrum (Angle). *2	Depends on data type.	Axis Instruction Unit
Length	LREAL	When sUVWR_STAGE_PARAMS.SettingMode is 1: Direct setting, please set the line segment connecting the center of rotation of the stage and the fulcrum. *3	Positive number	Axis Instruction Unit
Angle	LREAL	When sUVWR_STAGE_PARAMS.SettingMode is 1: Direct setting, also set the angle of the fulcrum. *3	$0^\circ \leq \text{Angle} < 360^\circ$	° (degree)

- *1. Set the direction of each axis of the UVWR axis to be either positive, or negative direction with respect to the X axis or Y axis direction at the XY coordinates (stage coordinates) where the center of rotation is the home. This will be either parallel or perpendicular to each UVWR axis.
The center of rotation of the stage is specified with the UVWR stage parameters *X coordinate of the center of rotation in the homing status* and *Y coordinate of the center of rotation in the homing status*.
In the case of the figure below, the setting is as follows.
U axis: Negative direction on the Y axis
V axis: Positive direction on the X axis
W axis: Positive direction on the Y axis
R axis: Negative direction on the X axis



- *2. Sets the coordinates of each axis at the homing status.
 When the Stage Control FB (CtrlStage) is executed, this setting value is converted to the length (Length) of the line segment connecting the center of rotation of the stage and the fulcrum. It is also converted to the angle (Angle) of the fulcrum by the following arithmetic expression done on that same set value. This then updates the Length and Angle of the structure member. Reverse kinematics calculation FUN (CalcInverseKinematics) and forward kinematics calculation FUN (CalcForwardKinematics) refer to Length and Angle. The center of rotation of the stage is specified with the UVWR stage parameters *X coordinate of the center of rotation in the homing status* and *Y coordinate of the center of rotation in the homing status*.



$$\Delta X = X - \text{RotationCenter_X}$$

$$\Delta Y = Y - \text{RotationCenter_Y}$$

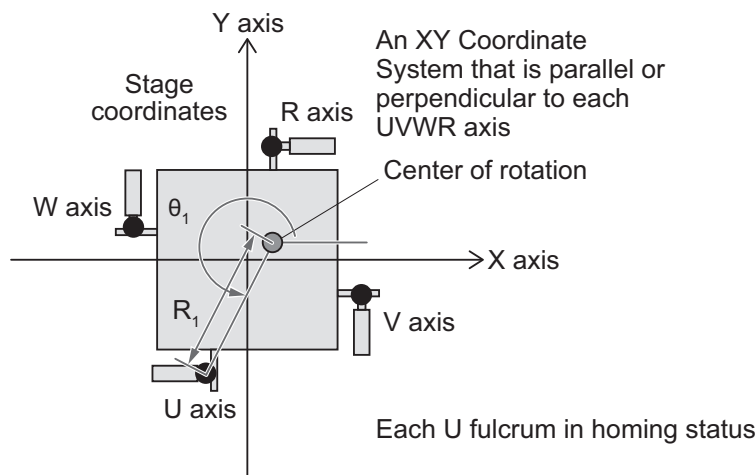
$$\text{Angle} = \tan^{-1}\left(\frac{\Delta Y}{\Delta X}\right)$$

$$\text{Length} = \sqrt{\Delta X^2 + \Delta Y^2}$$

- *3. Set the fulcrum of each axis of the UVWR axis in the homing status, the length of the line segment connecting the center of rotation, and the angle from the X axis of the line segment connecting the center of rotation and each axis fulcrum.

The center of rotation of the stage is specified with the UVWR stage parameters *X coordinate of the center of rotation in the homing status* and *Y coordinate of the center of rotation in the homing status*.

Definitions for the U axis:



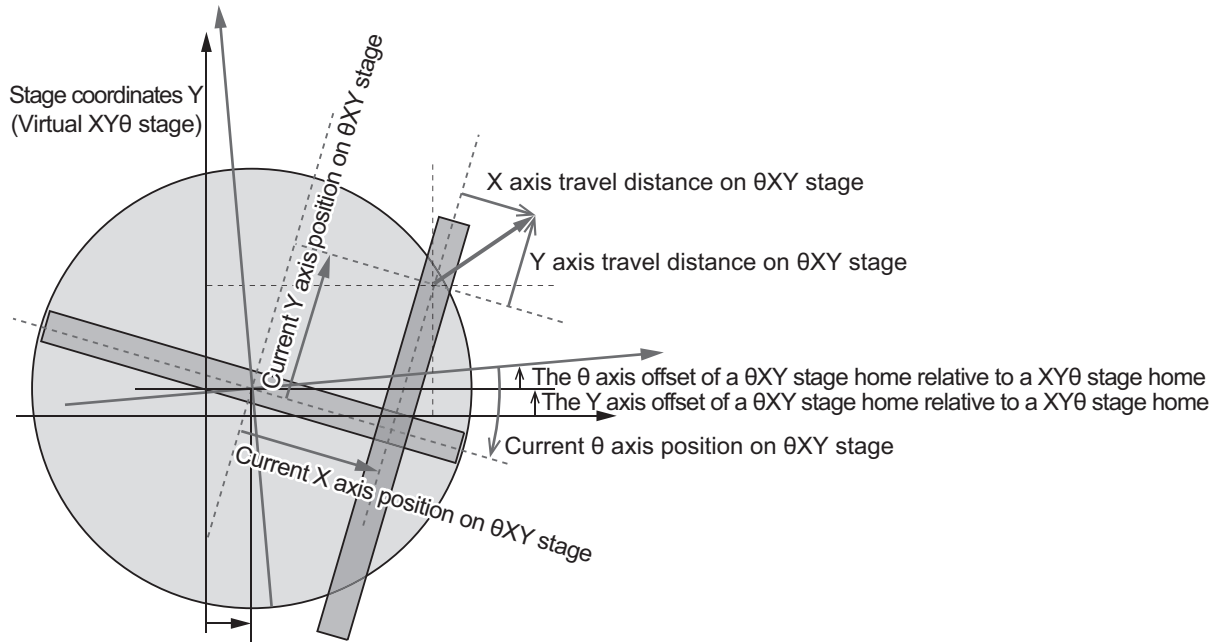
Forward/Reverse Kinematics Calculation Parameter (SKINEMATICS_PARAMS)

A forward kinematics calculation for converting the θXY ($\theta X, \theta Y$) stage coordinates or UVW (R) coordinates into virtual $XY\theta$ stage coordinates, and reverse kinematics calculation for converting virtual $XY\theta$ stage coordinates to θXY stage coordinates or UVW (R) coordinates. This parameter stores the settings used for both calculations.

Variable name	Data type	Description	Valid range	Unit
OmronLib\VF_Alignment\SKINEMATICS_PARAMS	STRUCT	—	—	—
Offset_X	LREAL	Sets the X axis offset. *1	Depends on data type.	Axis Instruction Unit
Offset_Y	LREAL	Sets the Y axis offset. *1	Depends on data type.	Axis Instruction Unit
Offset_TH	LREAL	Sets the θ axis offset. *1	Depends on data type.	° (degree)

*1. Set these for stage types 2, 7, 8.

In this library, the home position of the actual stage mechanism ($(x,y,\theta) = (0,0,0)$) is the home position of the virtual $XY\theta$. When using this library, offset values of the above axes are not used. Therefore, use the default value of 0.



The X axis offset of a θXY stage home relative to a XYθ stage home

Virtual XYθ Stage Axis Position/Velocity Calculation Parameter (sCALC-VIRTUALMOVE_PARAMS)

Set the parameters to be used to calculate the axis position and velocity on the Virtual XYθ Stage coordinates for alignment control.

Variable name	Data type	Description	Valid range	Unit
OmronLib\VF_Alignment \sCALCVIRTUALMOVE_PARAMS	STRUCT	—	—	—
MoveParams_X	OmronLib \VF_Alignment\sAXIS_MOVE_PARAMS	X axis motion parameter	—	—
MoveParams_Y	OmronLib \VF_Alignment\sAXIS_MOVE_PARAMS	Y axis motion parameter	—	—
MoveParams_TH	OmronLib \VF_Alignment\sAXIS_MOVE_PARAMS	θ axis motion parameter	—	—
InPosMode	UINT	In-Position Judgment Mode 0: Diameter / Angle 1: XYθ Position	0 to 1	—

Variable name	Data type	Description	Valid range	Unit
InPosRange	ARRAY[0..2] OF LREAL	Sets the In-Position range. Set this matching the values in InPosMode below. <ul style="list-style-type: none"> InPosMode = 0 [0]: Radius [1]: Half width of angle [2]: (Not used) InPosMode = 1 [0]: X-axis half width [1]: Y-axis half width [2]: θ-axis half width 	0 or greater	Axis Instruction Unit <ul style="list-style-type: none"> Angle, θ: ° (degree)
InPosTarget	ARRAY[0..2] OF LREAL	Sets the In-Position judgment target. TRUE: Judgment target FALSE: Not judgment target Set this matching the values in InPosMode below. <ul style="list-style-type: none"> InPosMode = 0 [0]: Diameter [1]: Angle [2]: (Not used) InPosMode = 1 [0]: X-axis [1]: Y-axis [2]: θ-axis 	TRUE, FALSE	—

Axis Motion Parameter (sAXIS_MOVE_PARAMS)

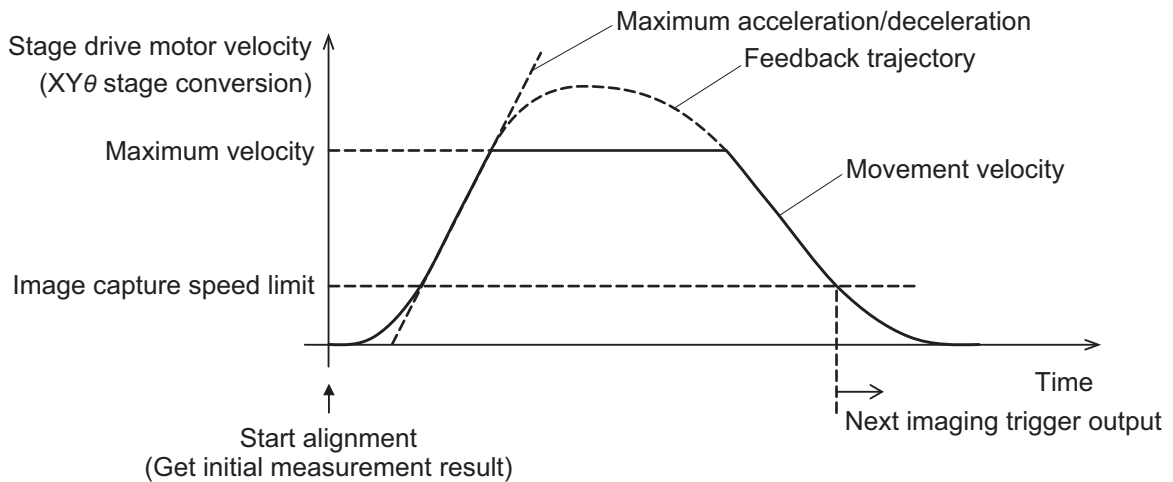
Sets the axis movement for a virtual stage.

Variable name	Data type	Description	Valid range	Unit
OmronLib\VF_Alignment \sAXIS_MOVE_PARAMS	STRUCT	—	—	—
Kp	LREAL	Proportional gain	0.0 to 3000.0	1/s
UseMaxVel	BOOL	TRUE: Use maximum ve- locity limit FALSE: Do not use maxi- mum velocity limit	TRUE, FALSE	—
MaxVel	LREAL	Maximum velocity* ¹	Positive number	Axis Instruction Unit/s <ul style="list-style-type: none"> θ only: degree/s
UseAccDec	BOOL	TRUE: Use Maximum Ac- celeration/Deceleration speed limit FALSE: Do not use Maxi- mum Acceleration/Decel- eration speed limit	TRUE, FALSE	—
AccDec	LREAL	Maximum Acceleration/ Deceleration* ¹	Positive number or 0	Axis Instruction Unit/s ² <ul style="list-style-type: none"> θ only: degree/s²

Variable name	Data type	Description	Valid range	Unit
ImagingLimitVel	LREAL	Image capture speed limit ^{*1}	Positive number	Axis Instruction Unit/s • θ only: degree/s

*1. These setting values are in relation to the commands generated by this library. If the motion control parameters - Operation Setting - maximum velocity, maximum acceleration, and maximum deceleration are set, the actual stage operation will follow those settings.

The axis motion parameters are for the motion of X, Y, and θ axes on the Virtual XYθ Stage coordinates that perform the alignment control calculation. Please set this parameter converted to XYθ stage operation. For the velocity of the feedback control calculated from the measurement result of the vision sensor, the set values of the maximum velocity and the maximum acceleration/deceleration are reflected as shown below.



As a guide for each stage mechanism, specify as follows.

XY, XYθ, θXY, X, Y, Xθ, Yθ, θX, θY: Set the maximum velocity and maximum acceleration/deceleration of X, Y, θ axes for each stage type

UVW, UVWR: Set the maximum velocity and maximum acceleration/deceleration that is common to each axis for UVW (R) as the axis motion parameters of X, Y, θ axes.

The unit of the θ axis motion parameter is angle (degree) (velocity: degree/s, acceleration/deceleration: degree/s²).

Set the following calculated value as a guide by using the length to the fulcrum relative to the center of rotation of the U/V/W/R stage.

$$\theta \text{ velocity} = \tan^{-1} (\text{U/V/W/R axis velocity} / \text{length to the fulcrum relative to the center of rotation})$$

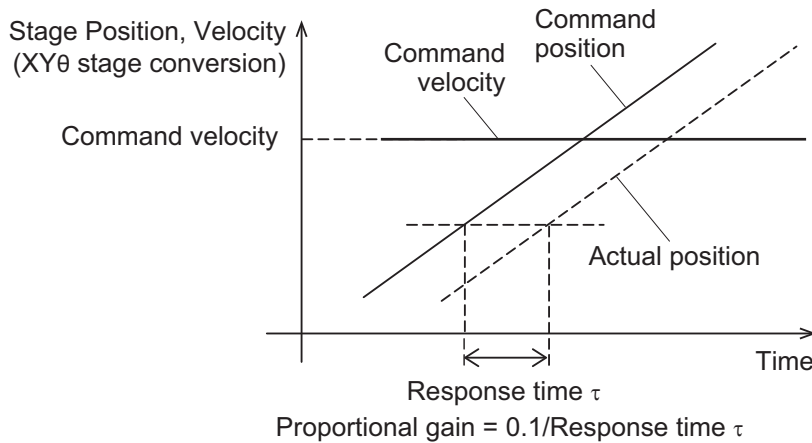
$$\theta \text{ Acceleration/Deceleration} = \tan^{-1} (\text{U/V/W/R Axis Acceleration/Deceleration} / \text{length to the fulcrum relative to the center of rotation})$$

When you want to reduce the occurrence of measurement errors due to shaking of the workpiece object, specify the upper velocity limit of the stage drive axis that outputs the second and subsequent measurement triggers (virtual XYθ stage conversion) using the Image capture speed limit. The following measurement trigger is output when any condition is satisfied with respect to the velocity limit specified for the XYθ axes.

For the Image capture speed limit, please set the following calculated value as a guide against the target accuracy of alignment and camera exposure time (1/shutter speed).

$$\text{Image capture speed limit} = \text{target accuracy} \times 20 / \text{camera exposure time}$$

Proportional gain specifies the degree of how much stage operation should be performed with respect to the alignment travel distance calculated based on the reference position and measurement position measured by the vision sensor in visual feedback control. Increasing the value will try to resolve the calculated alignment travel distance in a short time, but if it is too large, the motion will become abrupt or oscillatory. If the value is small, the operation will be smooth, but it will require alignment time. The proportional gain must be determined according to the time lag of the lag element of the alignment system being used. For the time lag between command position and feedback position when the stage is operated at a constant velocity, set the following calculated values as a guide and adjust according to the operation result.



Fifth Order Trajectory Calculation Parameter (sCALCURVE_PARAMS)

Sets the fifth order trajectory calculation parameters used for axis velocity calculation Function Block (FB).

Variable name	Data type	Description	Valid range	Unit
OmronLib\VF_Alignment \sCALCURVE_PARAMS	STRUCT	—	—	—
UseCurve	BOOL	TRUE: Use fifth order trajectory FALSE: Do not use fifth order trajectory	TRUE, FALSE	—
DistNumber*1	UINT	Sets the number of distributions. If "0" is set, it will change to "1".	0 to 9999	—
UseCurveSequence	BOOL	TRUE: Use fifth order trajectory sequence FALSE: Do not use fifth order trajectory sequence	TRUE, FALSE	—
UseConnectingVel	BOOL	TRUE: Use connecting velocity FALSE: Do not use connecting velocity	TRUE, FALSE	—

Variable name	Data type	Description	Valid range	Unit
MinCurve-Time	LREAL	Reserved by the system. Since it is updated by internal processing of CtrlStage, do not set a value during execution. If this variable is rewritten during execution, a normal fifth order trajectory calculation will not be performed. If you use the POU of the CtrlStage alone, set the minimum trajectory time.	0 or greater	ms
MaxCurve-Time	LREAL	Sets the maximum trajectory time. If "0" is set, it will be 1000 ms (= 1 s).	0 or greater	ms

*1. Disabled when using fifth order trajectory calculation (UseCurve = TRUE).

Imaging Parameter (sIMAGING_PARAMS)

To achieve correct alignment control, set parameters related to image capture on the vision sensor.

Variable name	Data type	Description	Valid range	Unit
OmronLib\VF_Alignment\sIMAGING_PARAMS	STRUCT	—	—	—
ShutterSpeed	LREAL	Sets the shutter speed. When multiple cameras with different shutter speeds are used, enter an average of those values.	Positive number	ms

Done Judgment Parameter (JUDGE_PARAMS)

Set the parameters for performing Completion Judgment for alignment control.

Variable name	Data type	Description	Valid range	Unit
OmronLib\VF_Alignment\sJUDGE_PARAMS	STRUCT	—	—	—
InPosMode	UINT	In-Position Judgment Mode	0: Diameter / Angle 1: XYθ Position	—

Variable name	Data type	Description	Valid range	Unit
InPosRange	ARRAY[0..2] OF LREAL	Sets the In-Position range of the measurement mark position in relation to the reference mark position. Set this matching the values in InPosMode below. <ul style="list-style-type: none"> InPosMode = 0 [0]: Radius [1]: Half width of angle [2]: (Not used) InPosMode = 1 [0]: X-axis half width [1]: Y-axis half width [2]: θ-axis half width 	0 or greater	Axis Instruction Unit <ul style="list-style-type: none"> Angle, θ: ° (degree)
InPosTarget	ARRAY [0..2] OF BOOL	Sets the In-Position judgment target. TRUE: Judgment target FALSE: Not judgment target Set this matching the values in InPosMode below. <ul style="list-style-type: none"> InPosMode = 0 [0]: Radius [1]: Angle [2]: (Not used) InPosMode = 1 [0]: X-axis [1]: Y-axis [2]: θ-axis 	TRUE, FALSE	—
TargetMark	ARRAY [0..3] OF BOOL	Set whether measurement point 0, point 1, point 2, or point 3 is used for Completion Judgment target. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array. TRUE: Calculate FALSE: Do not calculate	TRUE, FALSE	—

Calibration Axis Motion Parameter (sCALIB_MOVE_PARAMS)

Stores axis travel settings used when generating calibration parameters.

Variable name	Data type	Description	Valid range	Unit
OmronLib\VF_Alignment\sCALIB_MOVE_PARAMS	STRUCT	—	—	—
Movement_X	LREAL	Total travel distance in X axis direction	Positive number	Axis Instruction Unit
Movement_Y	LREAL	Total travel distance in Y axis direction	Positive number	Axis Instruction Unit
Movement_TH	LREAL	Total travel distance in θ axis direction	Positive number	° (degree)

Variable name	Data type	Description	Valid range	Unit
Velocity_XU	LREAL	X, U axes travel speed* ¹	Positive number	Axis Instruction Unit/s
AccDec_XU	LREAL	X, U axes Acceleration/Deceleration* ¹	Positive number or 0	Axis Instruction Unit/s ²
Velocity_YV	LREAL	Y, V axes travel speed* ¹	Positive number	Axis Instruction Unit/s
AccDec_YV	LREAL	Y, V axes Acceleration/Deceleration* ¹	Positive number or 0	Axis Instruction Unit/s ²
Velocity_THW	LREAL	θ , W axes travel speed* ¹	Positive number	Axis Instruction Unit/s
AccDec_THW	LREAL	θ , W axes Acceleration/Deceleration* ¹	Positive number or 0	Axis Instruction Unit/s ²
Velocity_R	LREAL	R axis travel speed* ¹	Positive number	Axis Instruction Unit/s
AccDec_R	LREAL	R axis Acceleration/Deceleration* ¹	Positive number or 0	Axis Instruction Unit/s ²
StopDec	ARRAY [0..3] OF LREAL	Deceleration speed at Deceleration stop* ¹ It is not necessary to set an unused axis. [0]: X/U axis [1]: Y/V axis [2]: θ /W axis [3]: R axis	Positive number or 0	Axis Instruction Unit/s ²
WaitTime	UINT	The wait time from Axis movement complete to Measurement trigger output. By setting the wait time, it is possible to take an image after any vibration to the camera due to the axis travel that is attenuated. When set to 0, there will be no wait time.	Positive number or 0	ms
HandSystem	BOOL	This is set when the type of the stage to be used is of the type in which only one axis out of X or Y axes exists. Specifies the direction of Angle from the existing axis. * ² TRUE: Left-handed system FALSE: Right-handed system	TRUE, FALSE	—
Angle	LREAL	This is set when the type of the stage to be used is of the type in which only one axis out of X or Y axes exists. Specifies the angle between the axes. * ² When set to 0, it will be 90°.	0° ≤ Angle < 360°	° (degree)

Variable name	Data type	Description	Valid range	Unit
Alpha	LREAL	This is set when the type of the stage to be used is of the type in which only one axis out of X or Y axes exists. Sets the relative magnification for the existing axes. *2 If set to 0, it will be 1.0.	Depends on data type.	—

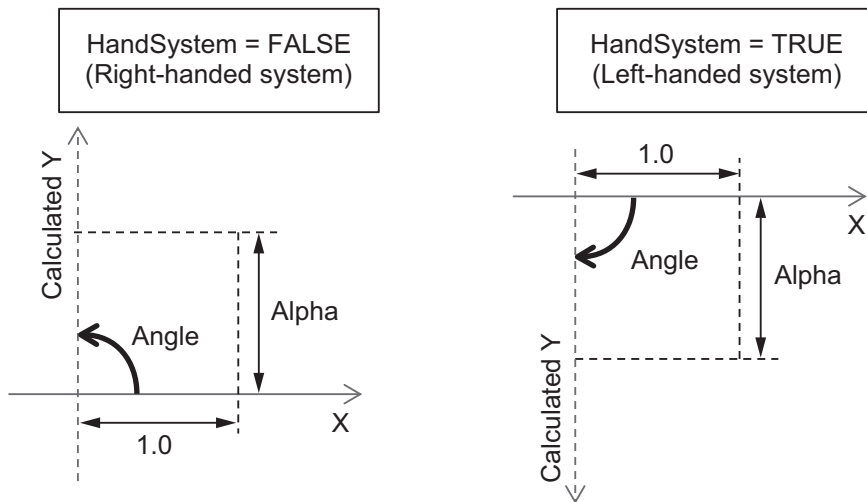
- *1. These setting values are in relation to the commands generated by this library. If the motion control parameters - Operation Setting - maximum velocity, maximum acceleration, and maximum deceleration are set, the actual stage operation will follow those settings.
- *2. This is an explanation of the right-handed system and the left-handed system of Xθ and θX stages where the Y axis does not exist. If the stage does not have the X axis, just exchange the X axis and Y axis information of the explanation below.

For the Xθ, θX stage, the translational motion is only for the X axis, but you can specify the coordinates on a two-dimensional plane together with the movement on the θ axis. In order to measure a two-dimensional plane, the coordinates in the Y axis direction are also taken into account for the calibration. The calibration axis motion parameters HandSystem, Angle, and Alpha are parameters for specifying the Y axis on this calculation.

HandSystem is the orientation of the coordinate system assumed by the machine and selects either right-handed or left-handed according to the direction of the Y axis on the machine.

Angle is the angle between the X axis and the calculated Y axis and specifies 90 ° in a typical Cartesian coordinate system. When set to 0, it will be 90°.

Alpha is the ratio of the length of the calculated axis (Y axis) to the unit length of the existing axis (X axis) and specifies the ratio of the length in the X axis direction and the length in the Y axis direction of the image acquired by the camera. When set to 0, it will be 1.0.



FB/FUN Structure Usage

The following table shows which members of the structure (for reference or update) are used in the FB/FUN for this library. Members that are not used even if there are structures in FB/FUN Input/Output variables are excluded from error checking.

Members not used by setting are also excluded from error checking.

For details of usage conditions, please check the individual structures or FB/FUN function descriptions.

Meaning of Symbols:

O: Reference only

●: Reference and Update

Structure name	FB/FUN name										
	AffineTrans	CalcPosAngle	CalcMultiPosAngle	CtrlStage	CalcMovement	CalcInverseKinematics	CalcForwardKinematics	CalcAxisVelocity	JudgeAlignmentComplete	GenerateTrigger	GenerateCalibParams
sPOSITION	—	—	—	—	—	—	—	—	—	—	—
X	●	●	●	●	○	○	●	○	○		○
Y	●	●	●	●	○	○	●	○	○		○
TH	●	●	●	●	○	○	●	○	○		○
sCALIB_PARAMS	—	—	—	—	—	—	—	—	—	—	—
A	○										●
B	○										●
C	○										●
D	○										●
E	○										●
F	○										●
sALIGNMENT_PARAMS	—	—	—	—	—	—	—	—	—	—	—
StageParams (sSTAGE_PARAMS)	—	—	—	—	—	—	—	—	—	—	—
StageType				○		○	○				○
AxNo				○		○	○				○
StopDec				○							
XYTHStageParams (sXYTH_STAGE_PARAMS)	—	—	—	—	—	—	—	—	—	—	—
Rotation						○	○				
TH_Type						○	○				
TH_Length						○	○				
TH_Offset						○	○				
UVWRStageParams (sUVWR_STAGE_PARAMS)	—	—	—	—	—	—	—	—	—	—	—
FulcType						○	○				
SettingMode				○							○
RotationCenter_X				○							○
RotationCenter_Y				○							○
U_Params (sUVWR_PARAMS)	—	—	—	—	—	—	—	—	—	—	—
Dir						○	○				
X				○							○
Y				○							○
Length				●		○	○				●
Angle				●		○	○				●
V_Params (sUVWR_PARAMS)	—	—	—	—	—	—	—	—	—	—	—

Structure name				FB/FUN name															
				AffineTrans	CalcPosAngle	CalcMultiPosAngle	CtrlStage	CalcMovement	CalcInverseKinematics	CalcForwardKinematics	CalcAxisVelocity	JudgeAlignmentComplete	GenerateTrigger	GenerateCallbParams					
			Dir						○	○									
			X				○										○		
			Y				○											○	
			Length				●		○	○								●	
			Angle				●		○	○								●	
			W_Params (sUVWR_PARAMS)				—	—	—	—	—	—	—	—	—	—	—	—	
			Dir								○	○							
			X							○								○	
			Y							○								○	
			Length							●		○	○					●	
			Angle							●		○	○					●	
			R_Params (sUVWR_PARAMS)				—	—	—	—	—	—	—	—	—	—	—	—	
			Dir									○	○						
			X							○								○	
			Y							○								○	
			Length							●		○	○					●	
			Angle							●		○	○					●	
			KinematicsParams (sKINEMATICS_PARAMS)				—	—	—	—	—	—	—	—	—	—	—	—	—
			Offset_X										○	○					
			Offset_Y										○	○					
Offset_TH										○	○								
CalcVirtualMoveParams (sCALCVIRTUALMOVE_PARAMS)				—	—	—	—	—	—	—	—	—	—	—	—	—			
MoveParams_X (sAXIS_MOVE_PARAMS)				—	—	—	—	—	—	—	—	—	—	—	—	—			
Kp												○							
UseMaxVel												○							
MaxVel												○							
UseAccDec												○							
AccDec												○							
ImagingLimitVel												○		○					
MoveParams_Y (sAXISMOVE_PARAMS)				—	—	—	—	—	—	—	—	—	—	—	—	—			
Kp												○							
UseMaxVel												○							
MaxVel												○							
UseAccDec												○							

Structure name			FB/FUN name									
			AffineTrans	CalcPosAngle	CalcMultiPosAngle	CtrlStage	CalcMovement	CalcInverseKinematics	CalcForwardKinematics	CalcAxisVelocity	JudgeAlignmentComplete	GenerateTrigger
		AccDec								<input type="radio"/>		
		ImagingLimitVel								<input type="radio"/>	<input type="radio"/>	
		MoveParams_TH (sAXIS-MOVE_PARAMS)	—	—	—	—	—	—	—	—	—	—
		Kp								<input type="radio"/>		
		UseMaxVel								<input type="radio"/>		
		MaxVel								<input type="radio"/>		
		UseAccDec								<input type="radio"/>		
		AccDec								<input type="radio"/>		
		ImagingLimitVel								<input type="radio"/>	<input type="radio"/>	
		InPosMode								<input type="radio"/>		
		InPosRange								<input type="radio"/>		
		InPosTarget								<input type="radio"/>		
		CalcCurveParams (sCALCURVE_PARAMS)	—	—	—	—	—	—	—	—	—	—
		UseCurve								<input type="radio"/>	<input type="radio"/>	
		DistNumber								<input type="radio"/>		
		UseCurveSequence								<input type="radio"/>		
		UseConnectingVel								<input type="radio"/>		
		MinCurveTime				●				<input type="radio"/>	<input type="radio"/>	
		MaxCurveTime								<input type="radio"/>		
		ImagingParams (sIMAGING_PARAMS)	—	—	—	—	—	—	—	—	—	—
		ShutterSpeed								<input type="radio"/>		
		JudgeParams (sJUDGE_PARAMS)	—	—	—	—	—	—	—	—	—	—
		InPosMode								<input type="radio"/>		
		InPosRange								<input type="radio"/>		
		InPosTarget								<input type="radio"/>		
		TargetMark								<input type="radio"/>		
		sCALIB_MOVE_PARAMS	—	—	—	—	—	—	—	—	—	—
		Movement_X									<input type="radio"/>	
		Movement_Y									<input type="radio"/>	
		Movement_TH									<input type="radio"/>	
		Velocity_XU									<input type="radio"/>	
		AccDec_XU									<input type="radio"/>	
		Velocity_YV									<input type="radio"/>	
		AccDec_YV									<input type="radio"/>	
		Velocity_THW									<input type="radio"/>	
		AccDec_THW									<input type="radio"/>	
		Velocity_R									<input type="radio"/>	
		AccDec_R									<input type="radio"/>	

Structure name	FB/FUN name										
	GenerateCalibParams	Generate Trigger	JudgeAlignmentComplete	CalcAxisVelocity	CalcForwardKinematics	CalcInverseKinematics	CalcMovement	CtrlStage	CalcMultiPosAngle	CalcPosAngle	AffineTrans
StopDec	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
WaitTime	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
HandSystem	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Angle	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Alpha	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

AffineTrans

Converts the reference or measurement mark position output from the vision sensor from the XYθ coordinates of the camera coordinates to the XYθ coordinates of the stage coordinates based on the calibration parameters.

It is possible to convert up to 4 coordinates at the same time.

Function name	Name	FB/FUN	Graphic expression	ST expression
Affine-Trans	Coordinate conversion	FUN		<pre>Out:=\\OmronLib VF_Alignment\Affi- neTrans(MarkPosCamera:=, CalibParams:=, CalcPosTarget:=, MarkPosStage:=);</pre>

Function Block and Function Information

Item	Description
Library file name	OmronLib_VF_Alignment_Vx_x.slr (x indicates the version)
Namespace	OmronLib\VF_Alignment
Source code published/not published	Not Published
Function Block and Function number	00201

Variables

Input Variables

Input variable	Name	Data type	Default	Valid range	Description
EN	Execute	BOOL	FALSE	TRUE, FALSE	TRUE: The function is executed FALSE: The function is not executed

Output Variables

Output variable	Name	Data type	Valid range	Description
ENO	ENO	BOOL	TRUE, FALSE	Only ladder diagram program is always output as TRUE.
Out	Return Value	UINT	*1	Output detailed conversion results.

*1. Refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 24 for details.

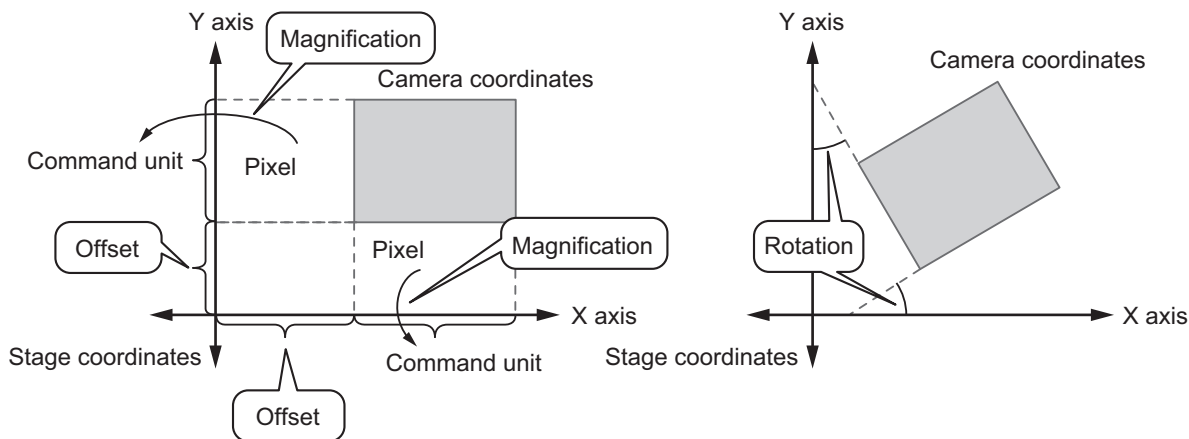
Input-Output Variables

Input-output variable	Name	Data type	Valid range	Description
MarkPosCamera	Mark position (Camera coordinates)	ARRAY[0..3] OF OmronLib \VF_Alignment\sPOSITION	—	Enter the mark position output from the vision sensor. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array.
CalibParams	Calibration parameter	ARRAY[0..3] OF OmronLib \VF_Alignment\sCALIB_PARAMS	—	Enter the calibration parameters for each point. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array.
CalcPosTarget	Number to process	ARRAY[0..3] OF BOOL	TRUE, FALSE	Sets whether each element of the MarkPosCamera array is to be calculated. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array. TRUE: Calculate FALSE: Do not calculate
MarkPosStage	Mark position (Stage coordinates)	ARRAY[0..3] OF OmronLib \VF_Alignment\sPOSITION	—	Outputs mark position converted to the stage coordinates.

Function

Affine conversion is used for converting (coordinate conversion) from the camera coordinates to the stage coordinates.

As the following diagram illustrates, in the affine conversion, it is necessary to calculate the calibration parameters including the individual magnifications of the X axis and the Y axis (the magnification to be converted from the pixel to the Axis Instruction Unit), the offset, and the rotation in the θ direction with respect to the coordinates of the camera coordinates. It converts all of these from the camera coordinates to the stage coordinates.



Trapezoidal distortion correction and lens distortion correction can not be performed.
 The point set as TRUE in the number to process (CalcPosTarget) is used in the calculation.
 No calculation is done for points set to FALSE and MarkPosStage does not change.

Precautions for Correct Use

Please set 0 for any coordinate value which can not be acquired from the vision sensor.
 Example: If only XY coordinates can be acquired, put 0 in for θ .

Calculations cannot be done in the following cases.

A non-zero value is output to the return value (Out), and MarkPosStage does not change.

For more information on the output value from Out, refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 24.

- When all elements of CalcPosTarget array are FALSE
- If the calculation result for any of X, Y, or θ for MarkPosStage is non-numeric ($\pm\infty$).

Sample Programming

Please refer to *Sample Programming* on page 4 - 90.

Troubleshooting (Error Codes and Corrective Actions)

Out	Status	Description	Corrective action
0	Conversion successful	—	—
1	Nothing to calculate	When all elements of CalcPosTarget array are FALSE	Please set so that at least one point can be calculated.
2	Invalid calculation result	The result of XY θ calculation is non-numeric ($\pm\infty$).	Please check whether the values entered for CalibParams and MarkPosCamera are ones that can be used in calculations.

CalcPosAngle

Calculate the position/angle (XYθ coordinates) to be used for alignment control from the 1 to 4 reference mark position, or measurement mark position points converted to the stage coordinates.

To calculate the position, use the *midpoint of target points*.

Function name	Name	FB/FUN	Graphic expression	ST expression
CalcPosAngle	Position and Angle Calculation	FUN		<pre>Out:=\\OmronLib VF_Alignment \CalcPosAngle(CalcAngleType:=, MarkPosCamera:=, CalcPosTarget:=, MarkPosAlign:=);</pre>

Function Block and Function Information

Item	Description
Library file name	OmronLib_VF_Alignment_Vx_x.slr (x indicates the version)
Namespace	OmronLib\VF_Alignment
Source code published/not published	Not Published
Function Block and Function number	00202

Variables

Input Variables

Input variable	Name	Data type	Default	Valid range	Description
EN	Execute	BOOL	FALSE	TRUE, FALSE	TRUE: The function is executed FALSE: The function is not executed
CalcAngle-Type	Angle calculation method	UINT	0	0 to 15	Sets the angle calculation method. For the value to set, refer to <i>Function</i> on page 4 - 26.

Output Variables

Output variable	Name	Data type	Valid range	Description
ENO	ENO	BOOL	TRUE, FALSE	Only ladder diagram program is always output as TRUE.
Out	Return Value	UINT	*1	Output detailed calculation results.

*1. Refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 28 for details.

Input-Output Variables

Input-output variable	Name	Data type	Valid range	Description
Mark-Pos-Stage	Mark position (Stage coordinates)	ARRAY[0..3] OF OmronLib\VF_Alignment\sPOSITION	—	Set the mark position converted to the stage coordinates. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array.
CalcPos-Target	Number to process	ARRAY[0..3] OF BOOL	TRUE, FALSE	Sets whether each element of the Mark-PosStage array is to be calculated. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array. TRUE: Calculate FALSE: Do not calculate
MarkPosAlign	Mark position (Alignment control)	OmronLib\VF_Alignment\sPOSITION	—	Outputs the calculation result for the Position of alignment control mark.

Function

Calculate the XYθ position to be used for alignment control from the mark positions 1 to 4.

- XY Calculation

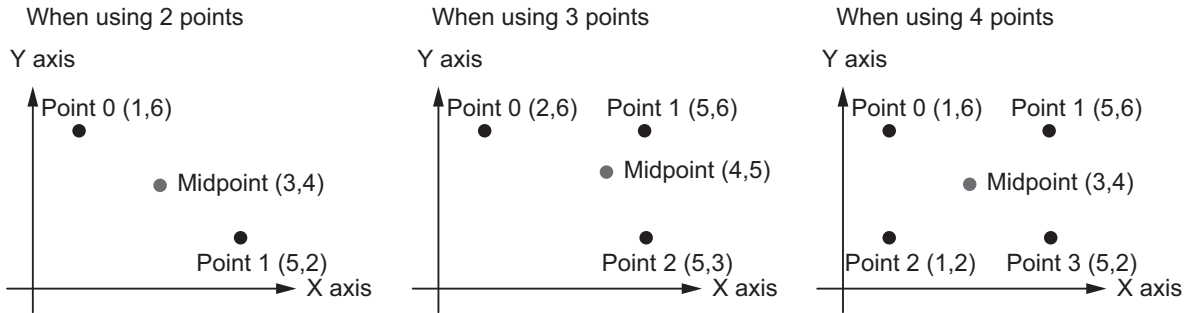
The midpoint of specified point is the XY position.

The point set as TRUE in the number to process (CalcPosTarget) is used in the calculation.

Points set to FALSE are not included in the calculation.

$$\text{Midpoint X} = \frac{\Sigma \text{ X coordinate of the point}}{\text{Number of points to be calculated}}$$

$$\text{Midpoint Y} = \frac{\Sigma \text{ Y coordinate of the point}}{\text{Number of points to be calculated}}$$



• Angle calculation

The angle is calculated based on the value set for the angle calculation method.

Values for "CalcAngleType"	Calculation method
0	Angle of point 0
1	Angle of point 1
2	Angle of point 2
3	Angle of point 3
4	Angle formed by points 0 to 1
5	Angle formed by points 1 to 0
6	Angle formed by points 2 to 0
7	Angle formed by points 3 to 0
8	Angle formed by points 0 to 2
9	Angle formed by points 1 to 2
10	Angle formed by points 2 to 1
11	Angle formed by points 3 to 1
12	Angle formed by points 0 to 3
13	Angle formed by points 1 to 3
14	Angle formed by points 2 to 3
15	Angle formed by points 3 to 2

CalcAngleType = 0 to 3

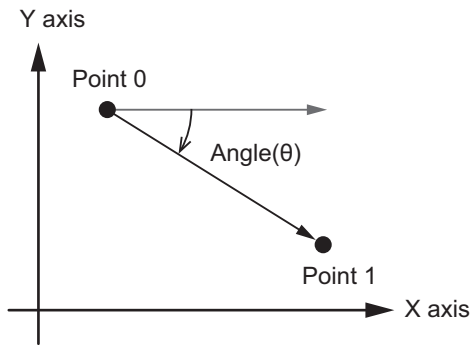
Set the input θ of the specified point to the output θ .

CalcAngleType = 4 to 15

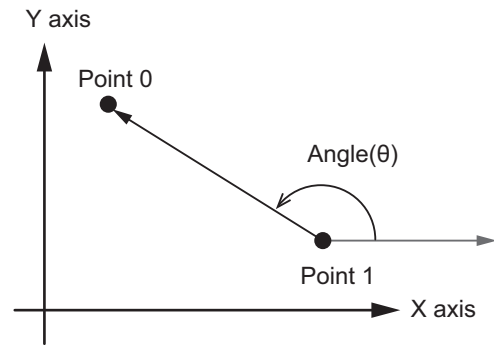
For Output θ , set the angle formed between the points M to N where there is a straight line drawn from point M in the positive direction on the X axis and the straight line drawn from the point M to the point N.

• Angle example

Angle formed by points 0 to 1



Angle formed by points 1 to 0



Precautions for Correct Use

Enter the mark position to the MarkPosStage array in stage coordinates. If you enter the mark position in camera coordinates, the correct result can not be obtained.

Perform the conversion calculation of the reference mark position and measurement mark position to the position for alignment control with the same function and calculation method.

If alignment control is performed according to positions calculated by different functions and calculation methods, correct control can not be performed.

Calculations cannot be done in the following cases.

A non-zero value is output to the return value (Out), and MarkPosAlign does not change.

Please refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 28 for output values for Out.

- When all elements of CalcPosTarget array are FALSE
- When the value set for "CalcAngleType" is outside of the valid range
- When the point specified by "CalcAngleType" does not exist in the "MarkPosStage" array
Example: CalcPosTarget [3] = FALSE if CalcAngleType=15 (angle formed by point 3-2)
- When the calculation result of MarkPosAlign is non-numeric ($\pm\infty$)

Sample Programming

Please refer to *Sample Programming* on page 4 - 90.

Troubleshooting (Error Codes and Corrective Actions)

Out	Status	Description	Corrective action
0	Calculation Successful	—	—
1	Nothing to calculate	When all elements of CalcPosTarget array are FALSE.	Please set so that at least one point can be calculated.
2	CalcAngleType Outside valid range	The value set for CalcAngleType is outside of the valid range.	Correct the value for CalcAngleType.

Out	Status	Description	Corrective action
3	CalcAngleType Incorrect	When the point specified by "CalcAngleType" does not exist in the "MarkPosStage" array.	Correct the value for CalcAngleType.
4	Invalid calculation result	The calculation result for MarkPosAlign is non-numeric ($\pm\infty$).	Please check whether the values entered for MarkPosAlign are ones that can be used in calculations.

CalcMultiPosAngle

Calculate the position/angle (XYθ coordinates) to be used for alignment control from the 2 to 4 reference mark position, or measurement mark position points converted to the stage coordinates.

To calculate the position, you can select from two methods, *least squares method* and *Maximum Error Minimization* method.

Function name	Name	FB/FUN	Graphic expression	ST expression
CalcMultiPosAngle	Multi-point Position and Angle Calculation	FUN		<pre>Out:=\\OmronLib \VF_Alignment \CalcMultiPosAngle(CalcType:=, ExistRotationAxis:=, TargetMarkPos- Stage:=, MeasMarkPos- Stage:=, CalcPosTarget:=, TargetMarkPosA- lign:=, MeasMarkPosA- lign:=);</pre>

Function Block and Function Information

Item	Description
Library file name	OmronLib_VF_Alignment_Vx_x.slr (x indicates the version)
Namespace	OmronLib\VF_Alignment
Source code published/not published	Not Published
Function Block and Function number	00203

Variables

Input Variables

Input variable	Name	Data type	Default	Valid range	Description
EN	Execute	BOOL	FALSE	TRUE, FALSE	TRUE: The function is executed FALSE: The function is not executed
CalcType	Calculation method	UINT	0	0 to 1	Sets the calculation method. For the value to set, refer to <i>Function</i> on page 4 - 32.

Input variable	Name	Data type	Default	Valid range	Description
ExistRotationAxis	With/without rotation	BOOL	FALSE	TRUE, FALSE	TRUE: With rotation FALSE: Without rotation

Output Variables

Output variable	Name	Data type	Valid range	Description
ENO	ENO	BOOL	TRUE, FALSE	Only ladder diagram program is always output as TRUE.
Out	Return Value	UINT	*1	Output detailed calculation results.

*1. Refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 34 for details.

Input-Output Variables

Input-output variable	Name	Data type	Valid range	Description
Target-Mark Pos-Stage	Reference mark position (Stage coordinates)	ARRAY[0..3] OF OmronLib\VF_Alignment\POSITION	—	Input the reference mark position using the stage coordinates. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array.
Meas-Mark Pos-Stage	Measurement mark position (Stage coordinates)	ARRAY[0..3] OF OmronLib\VF_Alignment\POSITION	—	Input the measurement mark position using the stage coordinates. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array.
CalcPos-Target	Number to process	ARRAY[0..3] OF BOOL	TRUE, FALSE	Sets whether each element of the Target-MarkPosStage array is to be calculated. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array. TRUE: Calculate FALSE: Do not calculate
Target-Mark PosAlign	Reference mark position (Alignment control)	OmronLib\VF_Alignment\POSITION	—	Outputs the calculation result for the alignment control reference mark position.

Input-output variable	Name	Data type	Valid range	Description
Meas-Mark PosAlign	Measurement mark position (Alignment control)	OmronLib\VF_Alignment\sPOSITION	—	Outputs the calculation result for the alignment control measurement mark position.

Function

Calculates the reference mark position (for alignment control) and measurement mark position (for alignment control) to be used for alignment. This is obtained from 2 to 4 reference mark positions (stage coordinates) and measurement mark positions (stage coordinates).

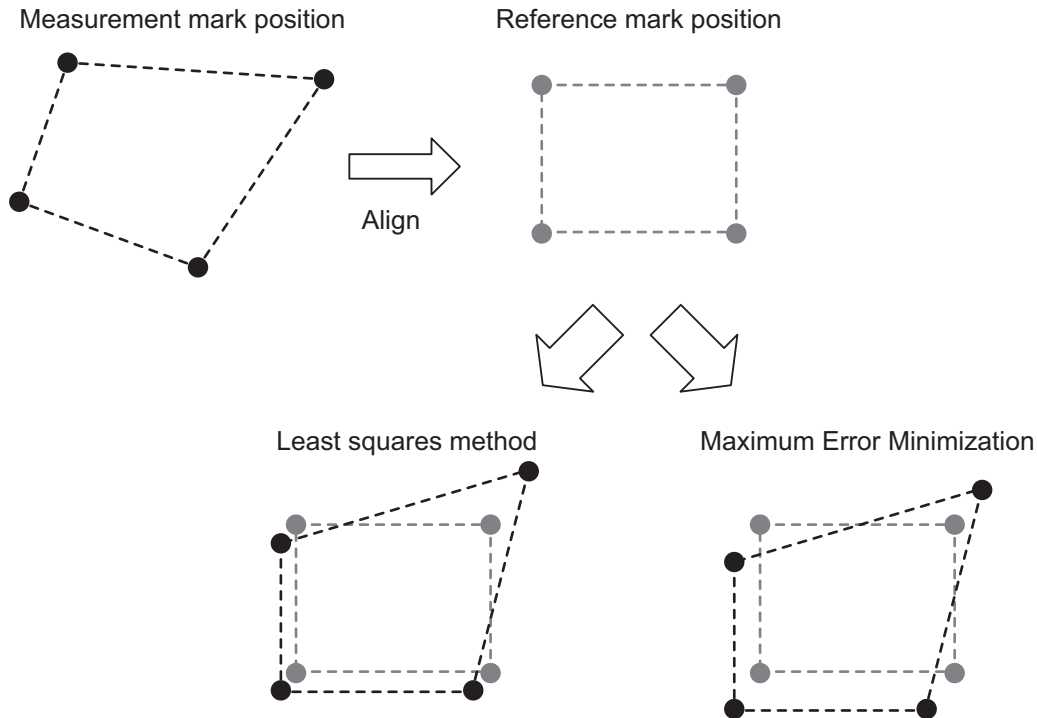
The point set as TRUE is used in the calculation.

Set with/without rotation to TRUE when the stage to be controlled is XY θ , θ XY, X θ , Y θ , θ X, θ Y, UVW, UVWR.

Set all other stage types besides these to FALSE.

Select one of the following 2 calculation methods.

Calculation method	CalcType value	Description
Least squares method	0	Calculate the reference mark position and measurement mark position so that the sum of squares of errors of all points (distance between the reference position and the measurement position) is minimized.
Maximum error minimization	1	Calculate the reference and measurement position so that the maximum value of the error of all points (the distance between the reference mark position and the measurement mark position) is minimized.



The values output to the X and Y coordinates of the reference mark position (for alignment control) and the measurement mark position (for alignment control) become the center of gravity of the input reference mark position (stage coordinates) and measurement mark position (stage coordinates).

The value output to the θ coordinate of the reference mark position (for alignment control) is 0 (degree).

The value output to the θ coordinate of the measurement mark position (for alignment control) is the θ amount of deviation (degree) of the reference and measurement.

Precautions for Correct Use

Always use the same function and calculation method to calculate the reference mark position in the stage coordinate system, the measurement mark position and the conversion of the measurement mark position to the reference mark position for alignment control.

If alignment control is performed according to positions calculated by different functions and calculation methods, correct control can not be performed.

Calculations cannot be done in the following cases.

A non-zero value is output to the return value (Out), and TargetMarkPosAlign and MeastMarkPosAlign do not change.

Please refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 34 for output values for Out.

- CalcType is outside valid range.
- There are less than 2 points set as TRUE in the CalcPosTarget array.
- When the calculation result of TargetMarkPosAlign is non-numeric ($\pm\infty$).
- When the calculation result of MeastMarkPosAlign is non-numeric ($\pm\infty$).

Sample Programming

Please refer to *Sample Programming* on page 4 - 90.

Troubleshooting (Error Codes and Corrective Actions)

Out	Status	Description	Corrective action
0	Calculation Successful	—	—
1	Insufficient for calculation	There are less than 2 valid points in the CalcPosTarget array.	Please set so that at least two points can be calculated.
2	CalcType Outside valid range	The value set for CalcType is outside of the valid range.	Correct the value for CalcType.
3	Reference XYθ Invalid calculation result	The calculation result for TargetMarkPosAlign is non-numeric ($\pm\infty$).	Please check whether the values entered for TargetMarkPosStage and MeasMarkPosStage are ones that can be used in calculations.
4	Measured XYθ Invalid calculation result	The calculation result for MeastMarkPosAlign is non-numeric ($\pm\infty$).	Please check whether the values entered for TargetMarkPosStage and MeasMarkPosStage are ones that can be used in calculations.
5	Calculation Time-out	Calculation processing was aborted because the calculation time for Maximum Error Minimization exceeded 50% of the task period.	Increase the task period setting of the controller.

CtrlStage

Alignment control is performed to make the reference XYθ position align with the measurement XYθ position.

Visual feedback control is used for axis travel distance calculation and axis movement control.

FB name	Name	FB/FUN	Graphic expression	ST expression
CtrlStage	Stage Control	FB	<p>The graphic expression shows a rectangular block representing the CtrlStage instance. The inputs on the left are: Execute, TargetMarkPosAlign, MeasMarkPosAlign, TriggerAck, TriggerBusy, ResultNotification, TotalJudgment, Abort, AlignmentParams, TargetMarkPosStage, and MeasMarkPosStage. The outputs on the right are: Done, Busy, Active, Trigger, VirtualStagePos, Difference_X, Difference_Y, Difference_TH, AlignmentParams, TargetMarkPosStage, MeasMarkPosStage, CommandAborted, Error, ErrorID, and ErrorIDEx.</p>	<pre>CtrlStage_instance(Execute:=, TargetMarkPosAlign:=, MeasMarkPosAlign:=, TriggerAck:=, TriggerBusy:=, ResultNotification:=, TotalJudgment:=, Abort:=, Done=>, Busy=>, Active=>, Trigger=>, VirtualStagePos=>, Difference_X=>, Difference_Y=>, Difference_TH=>, CommandAborted=>, Error=>, ErrorID=>, ErrorIDEx=>, AlignmentParams:=, TargetMarkPosStage:=, MeasMarkPosStage:=);</pre>

Function Block and Function Information

Item	Description
Library file name	OmronLib_VF_Alignment_Vx_x.slr (x indicates the version)
Namespace	OmronLib_VF_Alignment
Source code published/not published	Not Published
Function Block and Function number	00204

Variables

Input Variables

Input variable	Name	Data type	Default	Valid range	Description
Execute	Execute	BOOL	FALSE	TRUE, FALSE	TRUE: The function is executed
TargetMark PosAlign	Reference mark position (Alignment control)	OmronLib \WF_Alignment'sPOSITION	—	—	Enter the Alignment control reference mark position calculated by CalcPosAngle or CalcMultiPosAngle.
MeasMark PosAlign	Measurement mark position (Alignment control)	OmronLib \WF_Alignment'sPOSITION	—	—	Enter the Alignment control measurement mark position calculated by CalcPosAngle or CalcMultiPosAngle.
TriggerAck	Trigger acknowledged state	BOOL	FALSE	TRUE, FALSE	Input the Trigger acknowledged state signal (Trigger Ack) that is output from the vision sensor.
TriggerBusy	Busy	BOOL	FALSE	TRUE, FALSE	Input the Busy signal (Busy) that is output from the vision sensor.
ResultNotification	Data output complete	BOOL	FALSE	TRUE, FALSE	Enter the Data output complete signal (Result Notification) from the vision sensor.
TotalJudgment	Output Overall judgment	BOOL	FALSE	TRUE, FALSE	Enter the Overall judgment output signal (Total Judgment) from the vision sensor.
Abort	Abort	BOOL	FALSE	TRUE, FALSE	If TRUE, alignment control is aborted.

Output Variables

Output variable	Name	Data type	Valid range	Description
Done	Done	BOOL	TRUE, FALSE	Outputs TRUE when alignment control is complete.
Busy	Executing	BOOL	TRUE, FALSE	It is TRUE while executing.
Active	Alignment control	BOOL	TRUE, FALSE	Outputs TRUE while alignment control is in process.
Trigger	Measurement trigger	BOOL	TRUE, FALSE	The output signal which becomes the input (Trigger) to the Execute measurement bit of the vision sensor.
VirtualStagePos	Virtual stage position	OmronLib \WF_Alignment'sPOSITION	—	Enter the stage position in the Virtual XYθ Stage coordinates.

Output variable	Name	Data type	Valid range	Description
Difference_X	X axis deviation	LREAL	Depends on data type.	Outputs the X axis deviation between the measurement mark and the reference mark in the Virtual XYθ Stage coordinates.
Difference_Y	Y axis deviation	LREAL	Depends on data type.	Outputs the Y axis deviation between the measurement mark and the reference mark in the Virtual XYθ Stage coordinates.
Difference_TH	θ axis deviation	LREAL	Depends on data type.	Outputs the θ axis deviation between the measurement mark and the reference mark in the Virtual XYθ Stage coordinates.
CommandAborted	Command Aborted	BOOL	TRUE, FALSE	TRUE when alignment control is aborted.
Error	Error	BOOL	TRUE, FALSE	TRUE: Error end FALSE: Normal end, executing, or execution condition not met
ErrorID	Error Code	WORD	*1	This is the error ID for an error end. The value is 16#0000 for normal end.
ErrorIDEx	Expansion Error Code	DWORD	*1	This is the Extension Error ID at error end. The value is 16#00000000 for Normal end.

*1. Refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 43 for details.

- The timing to refresh outputs

Variable	Condition for becoming TRUE	Condition for becoming FALSE
Done	<ul style="list-style-type: none"> • When Alignment Control is completed 	<ul style="list-style-type: none"> • When Execute is FALSE When Execute is already FALSE on completion, after one task period.
Busy	<ul style="list-style-type: none"> • When Execute is TRUE 	<ul style="list-style-type: none"> • When Done is TRUE • When CommandAborted is TRUE • When Error is TRUE However, if the axis is in motion, TRUE continues to be output until Axis stop is completed.
Active	<ul style="list-style-type: none"> • After Execute Start via Execute = TRUE, when the first measurement is completed (ResultNotification = TRUE and TotalJudgment = FALSE). 	<ul style="list-style-type: none"> • When Judgment for Alignment control complete is given (JudgeAlignmentComplete.CtrlComplete = TRUE) • When CommandAborted is TRUE • When Error is TRUE
Trigger	<ul style="list-style-type: none"> • When JudgeAlignmentComplete.Trigger is TRUE 	<ul style="list-style-type: none"> • When JudgeAlignmentComplete.Trigger is FALSE • When CommandAborted is TRUE • When Error is TRUE
CommandAborted	<ul style="list-style-type: none"> • When Abort is TRUE • When CommandAborted occurs in the internal axis control FB 	<ul style="list-style-type: none"> • When Execute is FALSE When Execute is already FALSE on completion, after one task period.
Error	<ul style="list-style-type: none"> • At Error end 	<ul style="list-style-type: none"> • When Execute is FALSE When Execute is already FALSE on completion, after one task period.

Input-Output Variables

Input-output variable	Name	Data type	Valid range	Description
AlignmentParams	Alignment control parameter	OmronLib\VF_Alignment\sALIGNMENT_PARAMS	—	Input alignment control parameters. Refer to <i>Alignment Control Parameter (sALIGNMENT_PARAMS)</i> on page 4 - 2 for details.
Target-Mark-Pos-Stage	Reference mark position (Stage coordinates)	ARRAY[0..3] OF OmronLib\VF_Alignment\sPOSITION	—	Set the reference mark position converted to stage coordinates. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array.
Meas-Mark-Pos-Stage	Measurement mark position (Stage coordinates)	ARRAY[0..3] OF OmronLib\VF_Alignment\sPOSITION	—	Set the measurement mark position converted to the stage coordinates. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array.

Function

The alignment control is performed so that the reference mark position (for alignment control) calculated by AffineTrans FUN, CalcPosAngle FUN or CalcMultiPosAngle FUN matches the measurement mark position (for alignment control).

This FB includes the following FB/FUN to calculate the axis travel distance and control the axis travel by visual feedback control using the stage parameter, output information from the vision sensor, and encoder information.

- CalcMovement FUN
- CalcInverseKinematics FUN
- CalcForwardKinematics FUN
- CalcAxisVelocity FB
- JudgeAlignmentComplete FB
- GenerateTrigger FB
- MC_SyncMoveAbsolute
- MC_MoveVelocity
- MC_Stop

If Execute (Execute) is TRUE, alignment control starts.

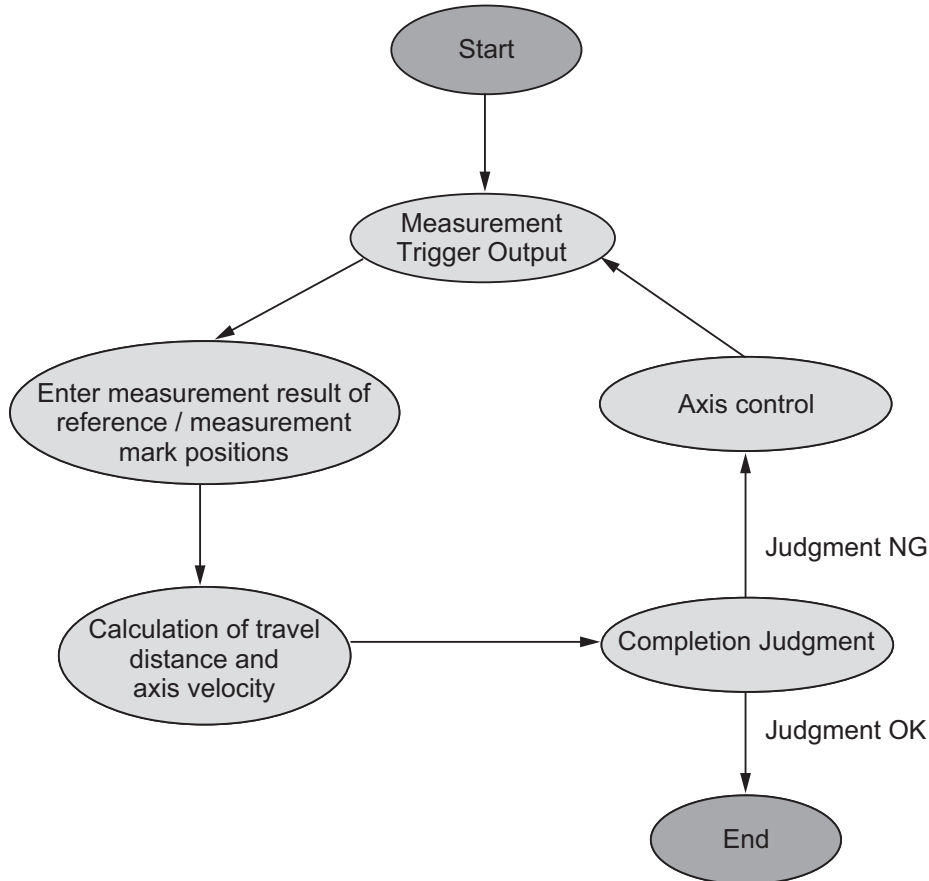
While executing alignment control, Executing (Busy) is TRUE.

Since the measurement trigger (Trigger) is output at the timing required for the vision sensor, please input to the vision sensor.

When the first successful measurement is detected (ResultNotification = TRUE and CtrlComplete = FALSE) by input from the vision sensor, the axis travel distance and axis velocity are calculated from

the reference mark position and the measurement mark position at that time, and the position control is started and alignment control in progress (Active) is set to TRUE.

Alignment control is executed repeatedly as shown in the figure below.



When it can be determined that the difference between the reference mark position and measurement mark position specified as the Completion Judgment target in sJUDGE_PARAMS.TargetMark is all within the In-Position range (sJUDGE_PARAMS.InPosRange), the alignment control is completed and alignment control in progress (Active) returns to FALSE.

For details on Completion Judgment, please refer to *Function* on page 4 - 67 under JudgeAlignment-Complete FB.

After completion of alignment control, MC_Stop is executed for all axes and Done = TRUE after confirming completion of stop of all axes.

- Command Aborted initiated by other axis control commands
During execution of this FB, do not execute other axis control commands on the axis used in the stage.
When axis control is performed by another command, stop all controlled axes, stop alignment control, and output TRUE to Command Aborted (CommandAborted).
Command Aborted (CommandAborted) is output while Execute (Execute) is TRUE.
To re-execute alignment control, set Execute (Execute) once to FALSE and then set it to TRUE.
- Command Aborted due to error condition
If an error occurs, TRUE is output for error (Error).
Alignment control will not be performed while there is an error state.

Please refer to *Precautions for Correct Use* on page 4 - 42 for the conditions that cause errors. To re-execute alignment control, set Execute (Execute) to FALSE, correct the cause of the error and set Execute (Execute) to TRUE again.

- The relevant axis decelerates to a stop.
This FB stops the controlled axis by using the MC_Stop instruction after decelerating with MC_MoveVelocity if the following conditions occur.
 - a) Alignment control complete
 - b) If *Abort* is entered
 - c) When an internal error in this FB occurs (except for MC_SyncMoveAbsolute command and MC_MoveVelocity command errors).
 - d) When there is a *CommandAborted* in MC_SyncMoveAbsolute
SCALIB_MOVE_PARAMS.StopDec is applied to the deceleration rate (MC_Stop.Deceleration) at this time.
If an error (Error) of the MC_SyncMoveAbsolute instruction or the MC_MoveVelocity instruction or an abort of the execution of the MC_MoveVelocity instruction (CommandAborted) occurs, stop immediately with the MC_Stop instruction.
When an error (Error) or Command Aborted (CommandAborted) for MC_Stop instruction occurs, it is treated the same as Done of the MC_Stop command and it is assumed that the immediate stop has been completed.

- Parameter setting
Please refer to *FB/FUN Structure Usage* on page 4 - 17 for the use of the members belonging to each structure.

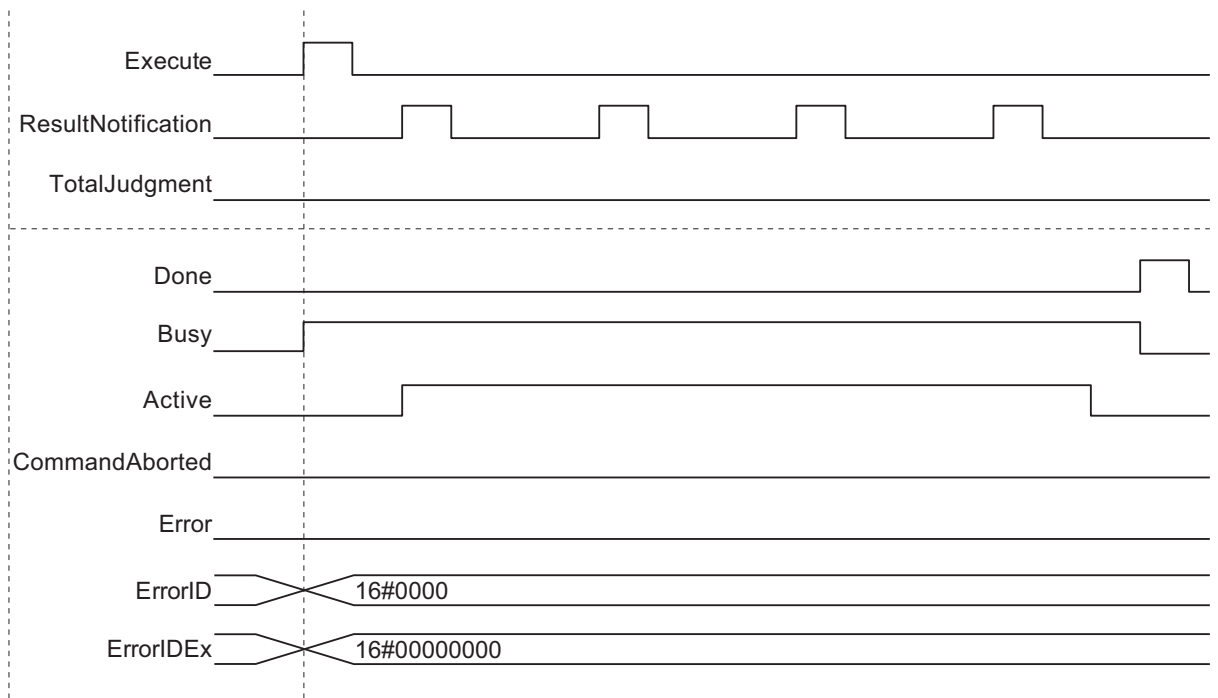
- Changing input parameters
The alignment control parameter retains the value at the start of execution.
Changes during execution are not reflected in the control.
If you want to make changes effective, restart Execute (Execute) after completing (Done = TRUE), aborting execution (CommandAborted = TRUE), and terminating the error (Error = TRUE). It is not possible to reboot while executing this.

Timing Charts

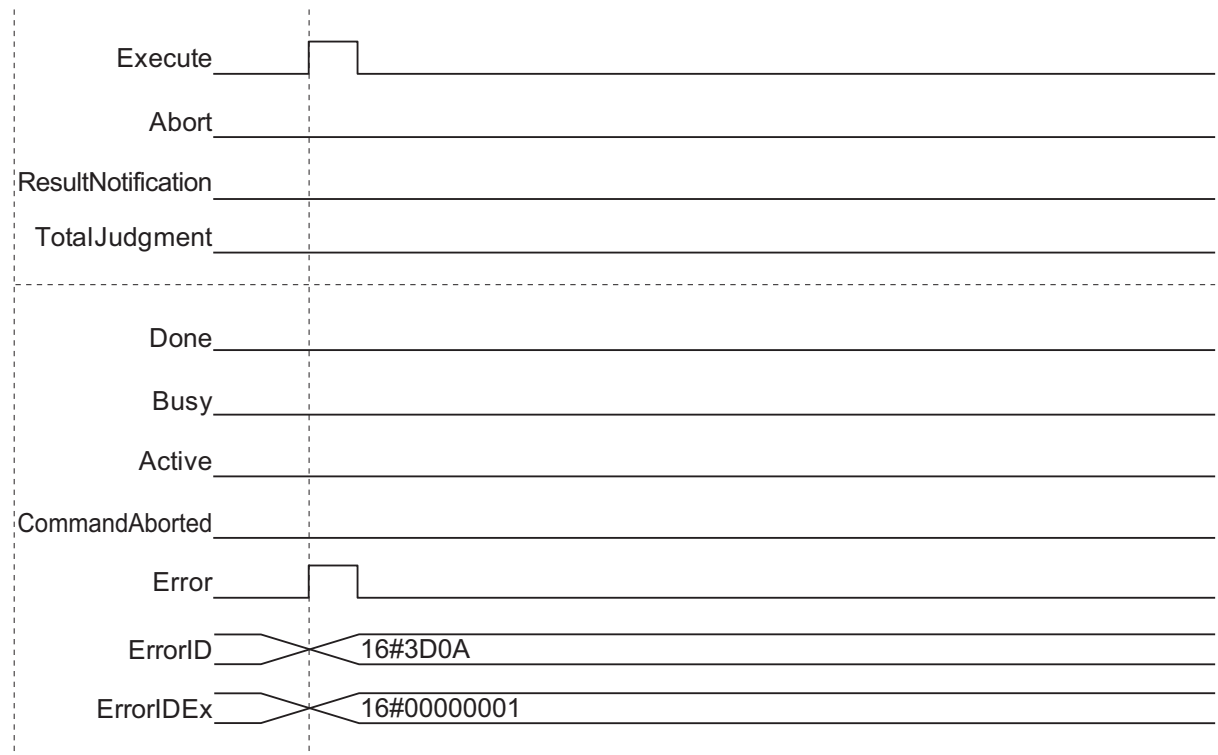
The timing charts are shown below.

- If Execute (Execute) is TRUE, alignment control starts.
- If an error occurs during execution of this function block, end calculation and Error (Error) changes to TRUE.
Alignment control will not be performed while there is an error state.
You can find out the cause of the error by accessing the values output to Error Code (ErrorID) and Expansion Error Code (ErrorIDEx).
- While Execute (Execute) is TRUE, it will stay in the Error (Error) state.
The Error Code (ErrorID) or Expansion Error Code (ErrorIDEx) is maintained until Execute is re-executed.
Alignment control will not be performed while there is an error state.
- If another axis control command is executed while this FB is being executed, Command Aborted (CommandAborted) becomes TRUE.
- While Execute (Execute) is TRUE, Command Aborted (CommandAborted) will be maintained.

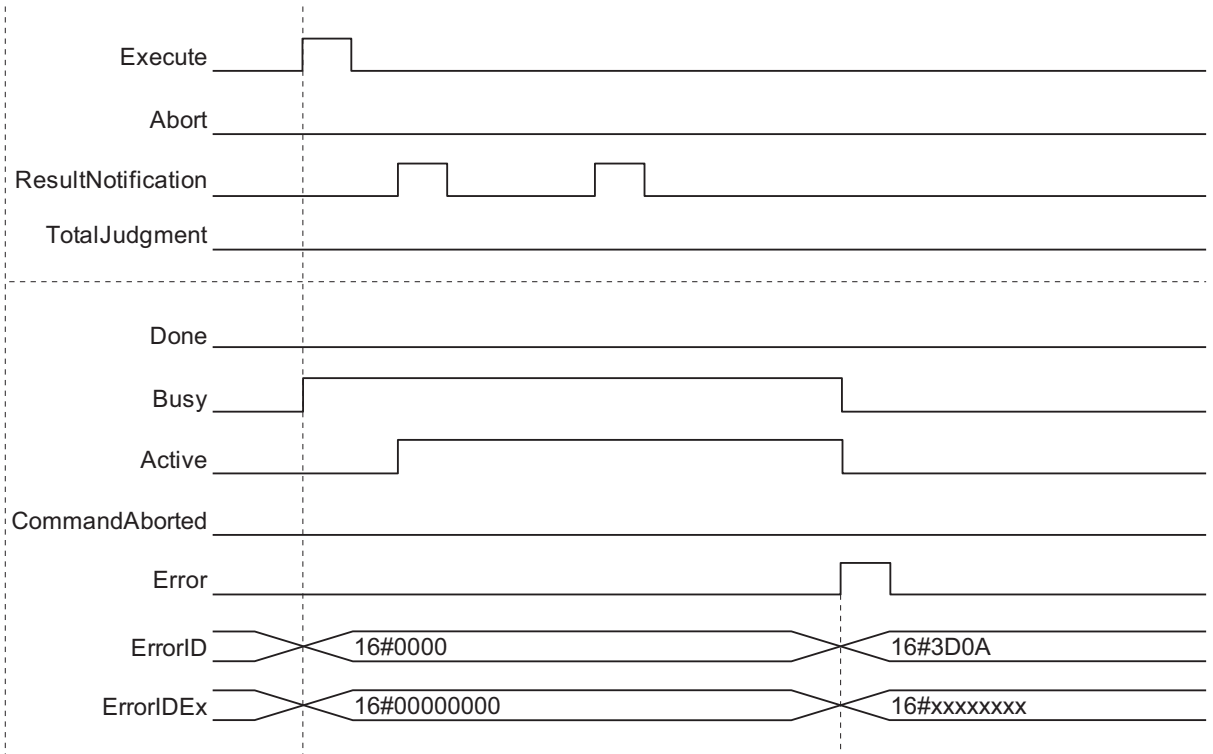
- Alignment control will not be performed while in a Command Aborted state.
- Timing Chart for Normal End



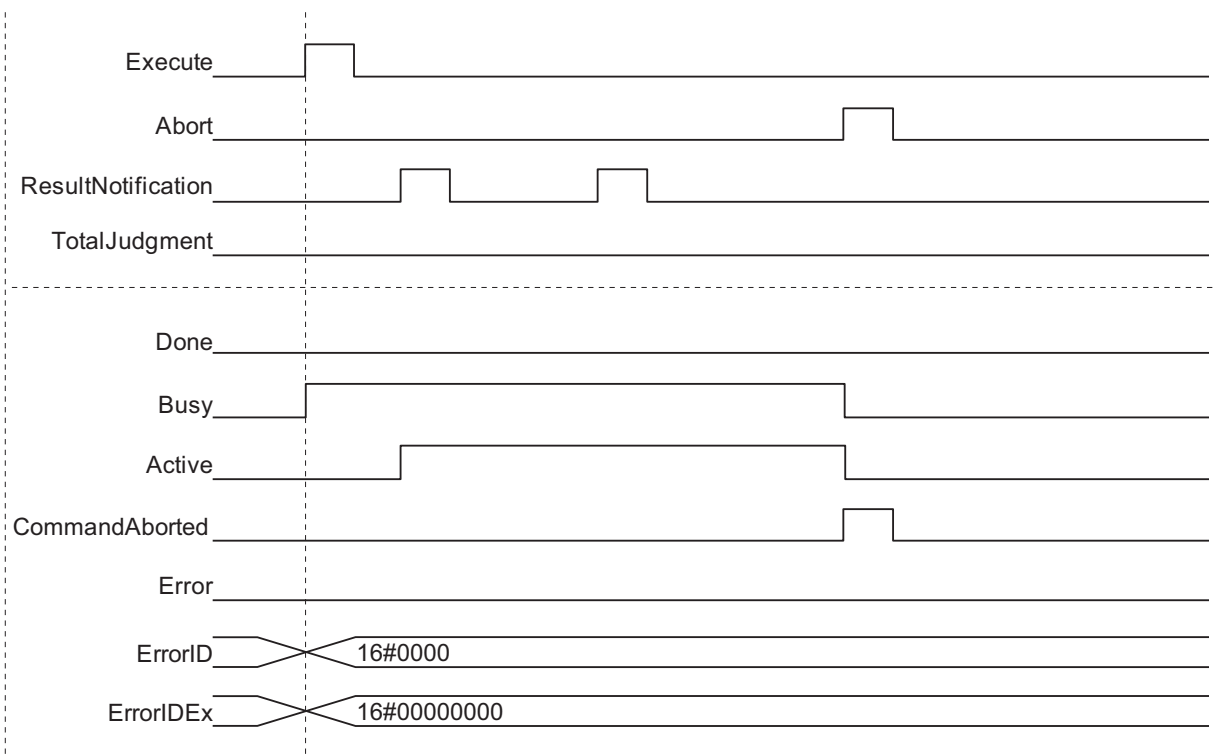
- Timing Chart for Error End (Error at start of execution)



- Timing Chart for Error End (Error while execution is in progress)



- Timing Chart for Command Aborted (Command Aborted while execution is in progress, or when entering Abort input variable)



Precautions for Correct Use

The following will result in an error.

TRUE is output for (Error) and all control axes decelerate to a stop.

If an error occurs in the user program, stop the axis with the user program.

- If an alignment control parameter outside of the valid range is set
- When the FB/FUN terminates with an error end

Within this FB, there is no monitoring of errors common to MC Function Module (MC common errors) and errors occurring on each axis (axis errors). Therefore, monitor during execution of this FB in the program and, in case of any error, abort execution of this FB and stop the axis movement.

For details on monitoring and stopping methods, refer to the following manuals.

- *NJ/NX-series CPU Unit Motion Control User's Manual (Cat. No. W507)*
- *NJ/NX-series Motion Control Instructions Reference Manual (Cat. No. W508)*
- *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Motion Control User's Manual (Cat. No. W559)*
- *NY Series Motion Control Instructions Reference Manual (Cat. No. W561)*

Within this FB, there is no verification process on its own prerequisite conditions for operation (such as verification that there is no error caused by EtherCAT communication with the vision sensor and Servo Drive, or that there is no error beyond a minor fault on the controlled axis, that it is in the Servo ON state, or that it is in the home defined state). For this reason, please check each state before executing this FB in any program.

For each state, refer to the following manual.

- *NJ/NX-series CPU Unit Motion Control User's Manual (Cat. No. W507)*
- *NJ/NX-series Motion Control Instructions Reference Manual (Cat. No. W508)*
- *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Motion Control User's Manual (Cat. No. W559)*
- *NY Series Motion Control Instructions Reference Manual (Cat. No. W561)*

Sample Programming

Please refer to *Sample Programming* on page 4 - 90.

Troubleshooting (Error Codes and Corrective Actions)

Error code	Ex-pan-sion error code	Status	Description	Corrective action
16#0000	16#000000	Normal end	—	—
16#3D0A	16#0000001	Invalid alignment control parameter	If an alignment control parameter outside of the valid range is set.	Correct the value entered for the alignment control parameter.
	16#100000x	CalcMovement Error	There is an error in CalcMovement.	Check the value for x against the Out value from CalcForwardKinematics and correct as needed.
	16#200000x	CalcInverseKinematics Error	There is an error in CalcInverseKinematics.	Check the value for x against the Out value from CalcInverseKinematics and correct as needed.

Error code	Expansion error code	Status	Description	Corrective action
	16#300000x	CalcForwardKinematics Error	There is an error in CalcForwardKinematics.	Check the value for x against the Out value from CalcForwardKinematics and correct as needed.
	16#4xxx	CalcAxisVelocity Error	There is an error in CalcAxisVelocity.	Check the xxxxxx value against the CalcAxisVelocity Extension Error Code (ErrorIDEx) and correct as needed.
	16#5xxx	JudgeAlignmentComplete Error	There is an error in JudgeAlignmentComplete.	Check the xxxxxx value against the JudgeAlignmentComplete Extension Error Code (ErrorIDEx) and correct as needed.
	16#6xxx	GenerateTrigger Error	There is an error in GenerateTrigger.	Check the xxxxxx value against the GenerateTrigger Extension Error Code (ErrorIDEx) and correct as needed.
	16#7000xxx	MC_SyncMoveAbsolute Error	There is an error in MC_SyncMoveAbsolute.	Check the xxxx value against the MC_SyncMoveAbsolute Error Code (ErrorID) and correct as needed. *1
	16#8000xxx	MC_MoveVelocity Error	There is an error in MC_MoveVelocity.	Check the xxxx value against the MC_MoveVelocity Error Code (ErrorID) and correct as needed. *1
	16#9000xxx	MC_Stop Error	There is an error in MC_Stop.	Check the xxxx value against the MC_Stop Error Code (ErrorID) and correct as needed. *1

*1. For error codes of MC_SyncMoveAbsolute, MC_MoveVelocity, MC_Stop, refer to the error code list in *NJ/NX-series Motion Control Instructions Reference Manual (Cat. No. W508)* or *NY Series Motion Control Instructions Reference Manual (Cat. No. W561)*.

CalcMovement

From the input reference XYθ position, measurement XYθ position, stage position XYθ position, calculate the travel distance on the X, Y, and θ axes on the virtual XYθ stage coordinates.

Function name	Name	FB/FUN	Graphic expression	ST expression
Calc-Movement	Travel distance calculation	FUN		<pre>Out:=\OmronLib \VF_Alignment \CalcMovement(TargetMarkPosA- lign:=, MeasMarkPosA- lign:=, VirtualStagePos:=, Movement_X=>, Movement_Y=>, Movement_TH=>, Difference_X=>, Difference_Y=>, Difference_TH=>);</pre>

Function Block and Function Information

Item	Description
Library file name	OmronLib_VF_Alignment_Vx_x.slr (x indicates the version)
Namespace	OmronLib\VF_Alignment
Source code published/not published	Not Published
Function Block and Function number	00205

Variables

Input Variables

Input variable	Name	Data type	Default	Valid range	Description
EN	Execute	BOOL	FALSE	TRUE, FALSE	TRUE: The function is executed FALSE: The function is not executed
TargetMark PosAlign	Reference mark position (Alignment control)	OmronLib \VF_Alignment\sPOSITION	—	—	Input the Alignment control reference mark position.

Input variable	Name	Data type	Default	Valid range	Description
MeasMark PosAlign	Measurement mark position (Alignment control)	OmronLib WF_Align- ment'sPOSI- TION	—	—	Input the alignment control measurement mark position.
VirtualStage- Pos	Virtual stage position	OmronLib WF_Align- ment'sPOSI- TION	—	—	Enter the stage position in the Virtual XYθ Stage coordi- nates.

Output Variables

Output variable	Name	Data type	Valid range	Description
ENO	ENO	BOOL	TRUE, FALSE	Only ladder diagram program is always output as TRUE.
Out	Return Val- ue	UINT	*1	Output detailed calculation results.
Move- ment_X	X axis trav- el distance	LREAL	Depends on data type.	Outputs the calculation result for the X axis travel distance.
Move- ment_Y	Y axis trav- el distance	LREAL	Depends on data type.	Outputs the calculation result for the Y axis travel distance.
Move- ment_TH	θ axis travel distance	LREAL	Depends on data type.	Outputs the calculation result for the θ axis travel distance.
Differ- ence_X	X axis devi- ation	LREAL	Depends on data type.	Outputs the X axis deviation between the measurement mark and the refer- ence mark in the Virtual XYθ Stage co- ordinates.
Differ- ence_Y	Y axis devi- ation	LREAL	Depends on data type.	Outputs the Y axis deviation between the measurement mark and the refer- ence mark in the Virtual XYθ Stage co- ordinate system.
Differ- ence_TH	θ axis devi- ation	LREAL	Depends on data type.	Outputs the θ axis deviation between the measurement mark and the refer- ence mark in the Virtual XYθ Stage co- ordinates.

*1. Refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 47 for details.

Function

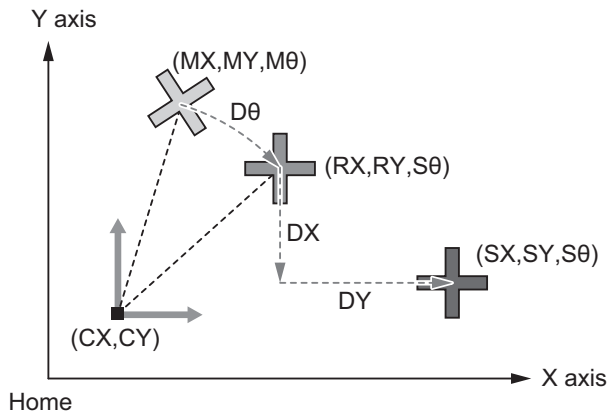
Performs the travel distance calculation of the X, Y and θ axes from the entered reference mark position, measurement mark position and current stage position.

For the value to be set to the virtual stage position, please convert the real position of the axis to the coordinates on the virtual XYθ stage coordinate system using the forward kinematics calculation (CalcForwardKinematics FUN) in advance.

If you set each current position of the stage control axis directly, the correct result may not be obtained.

It is necessary to convert the calculated travel distance to the travel distance of each axis controlling the stage by using reverse kinematics calculation (CalcInverseKinematics FUN).

- Diagram of converted image



Reference position/angle	(S_X, S_Y, S_θ)
Measurement position/angle	(M_X, M_Y, M_θ)
Current axis position	(C_X, C_Y)
≈ Position of center of rotation	
Axis travel distance	(D_X, D_Y, D_θ)
Midpoint position	(R_X, R_Y)
≈ Measurement position when measurement angle matches reference angle	

- Deviation

During execution, the differences between the X coordinate, Y coordinate, and θ coordinate of the reference mark position and measurement mark position are stored in Difference_X, Difference_Y, Difference_TH.

- Parameter setting

Please refer to *FB/FUN Structure Usage* on page 4 - 17 for the use of the members belonging to each structure.

Precautions for Correct Use

Calculations cannot be done in the following cases.

A non-zero value is output to the return value (Out) and 0 (zero) is output for Movement_X, Movement_Y, Movement_TH.

Please refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 47 for output values for Out.

- When the calculation result of Movement_X, Movement_Y, Movement_TH is non-numeric ($\pm\infty$).
- When the calculation result of Difference_X, Difference_Y, Difference_TH is non-numeric ($\pm\infty$).

Sample Programming

Please refer to *Sample Programming* on page 4 - 90.

Troubleshooting (Error Codes and Corrective Actions)

Out	Status	Description	Corrective action
0	Calculation Successful	—	—
1	Invalid X, Y, θ axis movement calculation result	The calculation result of X, Y, θ axes travel distance is non-numeric ($\pm\infty$).	Please check whether the values entered for TargetMarkPosAlign, MeasMarkPosAlign and VirtualStagePos are ones that can be used in calculations.
2	Invalid X,Y, θ deviation calculation result	The calculation result of X,Y, θ deviation is non-numeric ($\pm\infty$).	Please check whether the values entered for TargetMarkPosAlign, MeasMarkPosAlign and VirtualStagePos are ones that can be used in calculations.

CalcInverseKinematics

Performs reverse kinematics calculation to convert the travel distance of the XYθ axis on virtual XYθ stage coordinates to the target positions on the X/U axis, Y/V axis, θ/W axis, and R axis.

Function name	Name	FB/FUN	Graphic expression	ST expression
CalcInverseKinematics	Reverse kinematics calculation	FUN		<pre>Out:=\OmronLib\VF_Alignment\CalcInverseKinematics(Movement_X:=, Movement_Y:=, Movement_TH:=, VirtualStagePos:=, TargetPos_XU=>, TargetPos_YV=>, TargetPos_THW=>, TargetPos_R=>, AlignmentParams:=);</pre>

Function Block and Function Information

Item	Description
Library file name	OmronLib_VF_Alignment_Vx_x.slr (x indicates the version)
Namespace	OmronLib\VF_Alignment
Source code published/not published	Not Published
Function Block and Function number	00206

Variables

Input Variables

Input variable	Name	Data type	Default	Valid range	Description
EN	Execute	BOOL	FALSE	TRUE, FALSE	TRUE: The function is executed FALSE: The function is not executed
Movement_X	X axis travel distance	LREAL	0.0	Depends on data type.	Input the X axis travel distance.
Movement_Y	Y axis travel distance	LREAL	0.0	Depends on data type.	Input the Y axis travel distance.
Movement_TH	θ axis travel distance	LREAL	0.0	Depends on data type.	Input the θ axis travel distance.

Input variable	Name	Data type	Default	Valid range	Description
VirtualStage-Pos	Virtual stage position	OmronLib\VF_Alignment\sPOSITION	—	—	Enter the stage position in the Virtual XYθ Stage coordinates.

Output Variables

Output variable	Name	Data type	Valid range	Description
ENO	ENO	BOOL	TRUE, FALSE	Only ladder diagram program is always output as TRUE.
Out	Return Value	UINT	*1	Output detailed calculation results.
Target-Pos_XU	Target Position X/U axis	LREAL	Depends on data type.	Outputs the target position of the X/U axes.
TargetPos_YV	Target Position Y/V axis	LREAL	Depends on data type.	Outputs the target position of the Y/V axes.
Target-Pos_THW	Target Position θ/W axis	LREAL	Depends on data type.	Outputs the target position of the θ/W axes.
Target-Pos_R	Target Position R axis	LREAL	Depends on data type.	Outputs the target position of the R axis.

*1. Refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 50 for details.

Input-Output Variables

Input-output variable	Name	Data type	Valid range	Description
AlignmentParams	Alignment control parameter	OmronLib\VF_Alignment\sALIGNMENT_PARAMS	—	Input Alignment control parameters. Refer to <i>Alignment Control Parameter (sALIGNMENT_PARAMS)</i> on page 4 - 2 for details.

Function

Performs reverse kinematics calculation to convert the amount of movement in virtual XYθ stage coordinates to the target positions on the X/U axis, Y/V axis, θ/W axis, and R axis.

0 is always output as the target position for any axis that does not exist in the selected stage type.

When using CalcAxisVelocity FB, calculate the travel distance by the following formula.

Travel distance = CalcAxisVelocity travel velocity output × Task period

The following items are required to be set for alignment control parameters.

Meaning of Symbols

○ : Required setting

× : Settings not required (Not referenced)

Variable name		XY stage	XYθ stage	θXY stage	UVW stage
StageParams	StageType	○	○	○	○
	StageType setting values	(0,3,4)	(1,5,6)	(2,7,8)	(9,10)
	AxNo	○	○	○	○
	XYTHStageParams	×	○	○	×
	UVWRStageParams	×	×	×	○
Kinematics_Params		×	×	○	×

- Parameter setting

Please refer to *FB/FUN Structure Usage* on page 4 - 17 for the use of the members belonging to each structure.

Precautions for Correct Use

Calculations cannot be done in the following cases.

A non-zero value is output to the return value (Out) and *Current Axis Position* is output for TargetPos_XU, TargetPos_YV, TargetPos_THW, and TargetPos_R.

However, a value of 0 is output as the target position of the axis that does not exist on the stage.

Please refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 50 for output values for Out.

- If an alignment control parameter outside of the valid range is set
- When the calculation result for TargetPos_XU, TargetPos_YV, TargetPos_THW, or TargetPos_R is non-numeric ($\pm\infty$)

Sample Programming

Please refer to *Sample Programming* on page 4 - 90.

Troubleshooting (Error Codes and Corrective Actions)

Out	Status	Description	Corrective action
0	Calculation Successful	—	—
1	Invalid alignment control parameter	If an alignment control parameter outside of the valid range is set.	Correct the value entered for the alignment control parameter.
2	Invalid target position calculation	Calculation results for X/U, Y/V, θ/W, R axes target positions are non-numeric ($\pm\infty$).	Please check whether the values entered for Movement_X, Movement_Y, Movement_TH, VirtualStagePos are ones that can be used in calculations. Correct the value so it can be calculated as an alignment control parameter.

CalcForwardKinematics

Perform a forward kinematics calculation to convert the current position of each axis of the stage to the coordinate position on the Virtual XYθ Stage coordinates.

Function name	Name	FB/FUN	Graphic expression	ST expression
CalcForwardKinematics	Forward kinematics calculation	FUN		<pre>Out:=\OmronLib VF_Alignment \CalcForwardKine- matics(AlignmentParams:=, VirtualStagePos:=);</pre>

Function Block and Function Information

Item	Description
Library file name	OmronLib_VF_Alignment_Vx_x.slr (x indicates the version)
Namespace	OmronLib\FVF_Alignment
Source code published/not published	Not Published
Function Block and Function number	00207

Variables

Input Variables

Input variable	Name	Data type	Default	Valid range	Description
EN	Execute	BOOL	FALSE	TRUE, FALSE	TRUE: The function is executed FALSE: The function is not executed

Output Variables

Output variable	Name	Data type	Valid range	Description
ENO	ENO	BOOL	TRUE, FALSE	Only ladder diagram program is always output as TRUE.
Out	Return Value	UINT	*1	Output detailed calculation results.

*1. Refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 53 for details.

Input-Output Variables

Input-output variable	Name	Data type	Valid range	Description
AlignmentParams	Alignment control parameter	OmronLib\VF_Alignment\sALIGNMENT_PARAMS	—	Input Alignment control parameters. Refer to <i>Alignment Control Parameter (sALIGNMENT_PARAMS)</i> on page 4 - 2 for details.
VirtualStagePos	Virtual stage position	OmronLib\VF_Alignment\sPOSITION	—	Enter the stage position calculation result in the Virtual XYθ Stage coordinates.

Function

Perform a forward kinematics calculation to convert from the current position of each axis on the stage to the coordinate position on the virtual XYθ stage coordinates.

Enter the value that this FUN outputs as the input to virtual stage position (VirtualStagePos) in CalcMovement and CalcInverseKinematics.

The following items are required settings for alignment control parameters.

Meaning of Symbols

○ : Required setting

× : Settings not required (Not referenced)

Variable name		XY stage	XYθ stage	θXY stage	UVW stage
StageParams	StageType	○	○	○	○
	StageType setting values	(0,3,4)	(1,5,6)	(2,7,8)	(9,10)
	AxNo	○	○	○	○
	XYTHStageParams	×	○	○	×
	UVWRStageParams	×	×	×	○
KinematicsParams		×	×	○	×

- Parameter setting

Please refer to *FB/FUN Structure Usage* on page 4 - 17 for the use of the members belonging to each structure.

Precautions for Correct Use

Calculations cannot be done in the following cases.

A non-zero value is output to the return value (Out), and VirtualStagePos does not change.

Please refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 53 for output values for Out.

- If an alignment control parameter outside of the valid range is set
- If the calculation result for virtual stage position is non-numeric ($\pm\infty$)

Sample Programming

Please refer to *Sample Programming* on page 4 - 90.

Troubleshooting (Error Codes and Corrective Actions)

Out	Status	Description	Corrective action
0	Calculation Successful	—	—
1	Invalid alignment control parameter	If an alignment control parameter outside of the valid range is set.	Correct the value entered for the alignment control parameter.
2	Invalid calculation result	The virtual stage position calculation result is non-numeric ($\pm\infty$).	Correct the value so it can be calculated as an alignment control parameter.

CalcAxisVelocity

The axis movement velocity for each task period is output from the travel distance of each axis on the Virtual XYθ Stage coordinates and from the current stage position.

Since the task period of the controller is shorter than the calculation time of the vision sensor, in the task period where the measurement result is not output from the vision sensor, the axis movement velocity is output by interpolation calculation.

FB name	Name	FB/FUN	Graphic expression	ST expression
CalcAxisVelocity	Axis velocity calculation	FB	<p>The graphic expression shows the following inputs and outputs for the CalcAxisVelocity_instance block:</p> <ul style="list-style-type: none"> Inputs: Enable, Movement_X, Movement_Y, Movement_TH, VirtualStagePos, TriggerInput, TriggerAck, TriggerBusy, ResultNotification, TotalJudgment, AlignmentParams Outputs: Velocity_X, Velocity_Y, Velocity_TH, CurveTime_X, CurveTime_Y, CurveTime_TH, MeasDataRefresh, InpositionPV, Error, ErrorID, ErrorIDEx 	<pre>CalcAxisVelocity_instance(Enable:=, Movement_X:=, Movement_Y:=, Movement_TH:=, VirtualStagePos:=, TriggerInput:=, TriggerAck:=, TriggerBusy:=, ResultNotification:=, TotalJudgment:=, Enabled=>, Velocity_X=>, Velocity_Y=>, Velocity_TH=>, CurveTime_X=>, CurveTime_Y=>, CurveTime_TH=>, MeasDataRefresh=>, InpositionPV=>, Error=>, ErrorID=>, ErrorIDEx=>, AlignmentParams:=);</pre>

Function Block and Function Information

Item	Description
Library file name	OmronLib_VF_Alignment_Vx_x.slr (x indicates the version)
Namespace	OmronLib_VF_Alignment
Source code published/not published	Not Published
Function Block and Function number	00208

Variables

Input Variables

Input variable	Name	Data type	Default	Valid range	Description
Enable	Execute	BOOL	FALSE	TRUE, FALSE	TRUE: Execute FALSE: Do not execute
Movement_X	X axis travel distance	LREAL	0	Depends on data type.	Input the X axis travel distance.
Movement_Y	Y axis travel distance	LREAL	0	Depends on data type.	Input the Y axis travel distance.
Movement_TH	θ axis travel distance	LREAL	0	Depends on data type.	Input the θ axis travel distance.
VirtualStage-Pos	Virtual stage position	OmronLib \VF_Alignment\POSITION	—	—	Enter the stage position in the Virtual XY θ Stage coordinates.
TriggerInput	Measurement trigger input	BOOL	FALSE	TRUE, FALSE	Inputs the measurement trigger input to the vision sensor.
TriggerAck	Trigger acknowledged state	BOOL	FALSE	TRUE, FALSE	Input the Trigger acknowledged state signal (Trigger Ack) that is output from the vision sensor.
TriggerBusy	Busy	BOOL	FALSE	TRUE, FALSE	Input the Busy signal (Busy) that is output from the vision sensor.
ResultNotification	Data output complete	BOOL	FALSE	TRUE, FALSE	Enter the Data output complete signal (Result Notification) from the vision sensor.
TotalJudgment	Output Overall judgment	BOOL	FALSE	TRUE, FALSE	Enter the Overall judgment output signal (Total Judgment) from the vision sensor.

Output Variables

Output variable	Name	Data type	Valid range	Description
Enabled	Executing	BOOL	TRUE, FALSE	It is TRUE while executing.
Velocity_X	X axis movement velocity	LREAL	Depends on data type.	Unit: Axis Instruction Unit/s Outputs the calculated X axis movement velocity.
Velocity_Y	Y axis movement velocity	LREAL	Depends on data type.	Unit: Axis Instruction Unit/s Outputs the calculated Y axis movement velocity.
Velocity_TH	θ axis movement velocity	LREAL	Depends on data type.	Unit: degree/s Outputs the calculated θ axis movement velocity.
Curve-Time_X	X axis path time	LREAL	Depends on data type.	Unit: ms Outputs the calculated X axis path time.

Output variable	Name	Data type	Valid range	Description
Curve-Time_Y	Y axis path time	LREAL	Depends on data type.	Unit: ms Outputs the calculated Y axis path time.
Curve-Time_TH	θ axis path time	LREAL	Depends on data type.	Unit: ms Outputs the calculated θ axis path time.
MeasDataRefresh	Refresh measurement data	BOOL	TRUE, FALSE	TRUE when the measurement is completed.
InpositionPV	Inposition according to the command position	BOOL	TRUE, FALSE	TRUE when the measured position is within the In-Position range of the reference position.
Error	Error	BOOL	TRUE, FALSE	TRUE: Error end FALSE: Normal end, executing, or execution condition not met
ErrorID	Error Code	WORD	*1	This is the error ID for an error end. The value is 16#0000 for normal end.
ErrorIDEx	Expansion Error Code	DWORD	*1	This is the Extension Error ID at error end. The value is 16#00000000 for normal end.

*1. Refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 63 for details.

- The timing to refresh outputs

Variable	Condition for becoming TRUE	Condition for becoming FALSE
Enabled	<ul style="list-style-type: none"> • When Enable is TRUE 	<ul style="list-style-type: none"> • When Enable is FALSE • When Error is TRUE
MeasDataRefresh	<ul style="list-style-type: none"> • When the fifth order trajectory sequence is valid (sCALCURVE_PARAMS.UseCurveSequence = TRUE) and the measurement is completed (Data output complete or Start imaging or shutter speed time exceeded) 	<ul style="list-style-type: none"> • When Enable is FALSE • Next task cycle when this output becomes TRUE • When Error is TRUE
InpositionPV	<ul style="list-style-type: none"> • When measurement position is within the reference In-Position range 	<ul style="list-style-type: none"> • When measurement position is outside of the reference In-Position range
Error	<ul style="list-style-type: none"> • At Error end 	<ul style="list-style-type: none"> • When Enable is FALSE

Input-Output Variables

Input-output variable	Name	Data type	Valid range	Description
AlignmentParams	Alignment control parameter	OmronLib\VF_Alignment\sALIGNMENT_PARAMS	—	Input Alignment control parameters. Refer to <i>Alignment Control Parameter (sALIGNMENT_PARAMS)</i> on page 4 - 2 for details.

Function

The axis movement velocity for each task period is output from the travel distance of the XYθ axes in Virtual XYθ Stage coordinates and from the stage position.

Use the axis movement velocity output by this FB to control the stage axis.

Also, output of path time (only when fifth order trajectory is enabled), In-Position Judgment of measurement position and reference position is performed.

Velocity calculation and In-Position Judgment are performed while execution (Enable) is TRUE.

While executing, TRUE is output for Enabled.

When execution (Enable) is FALSE, output velocity and path time are always 0, and InPositionPV is FALSE.

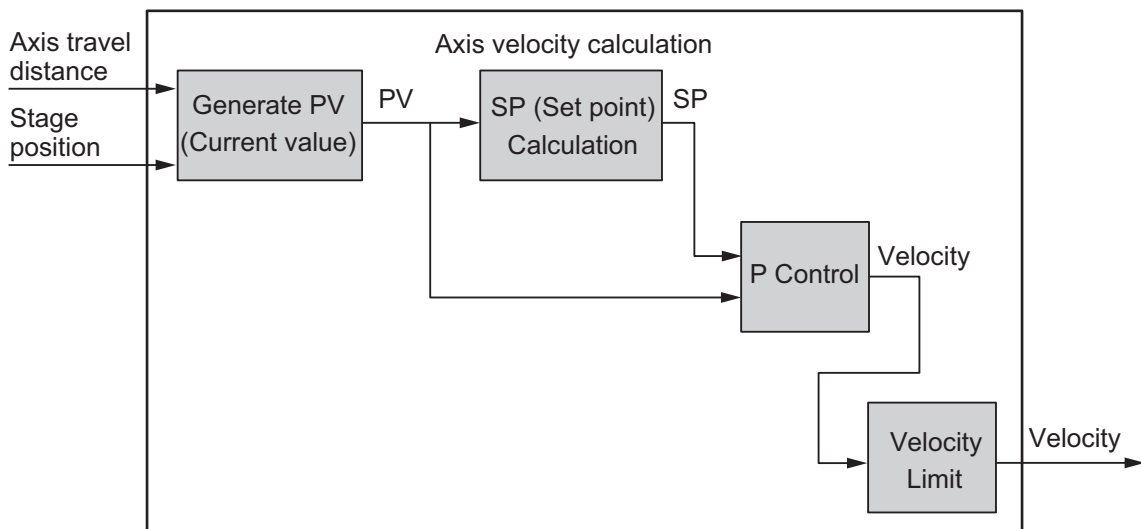
If an error occurs, TRUE is output for error (Error).

During an error, no calculation is done, 0 is output for output velocity and path time, and FALSE is output for InPositionPV.

Please refer to *Precautions for Correct Use* on page 4 - 63 for the conditions that cause errors.

- Axis velocity calculation

Each axis velocity calculation of XYθ is composed of four processes: Generate PV (Current value), Generate SP (Set point), Generate velocity by P control, and Velocity limit as shown below.



PV and SP are values generated to calculate the velocity inside the FB and are never output outside the FB.

- Generate PV (Current value)

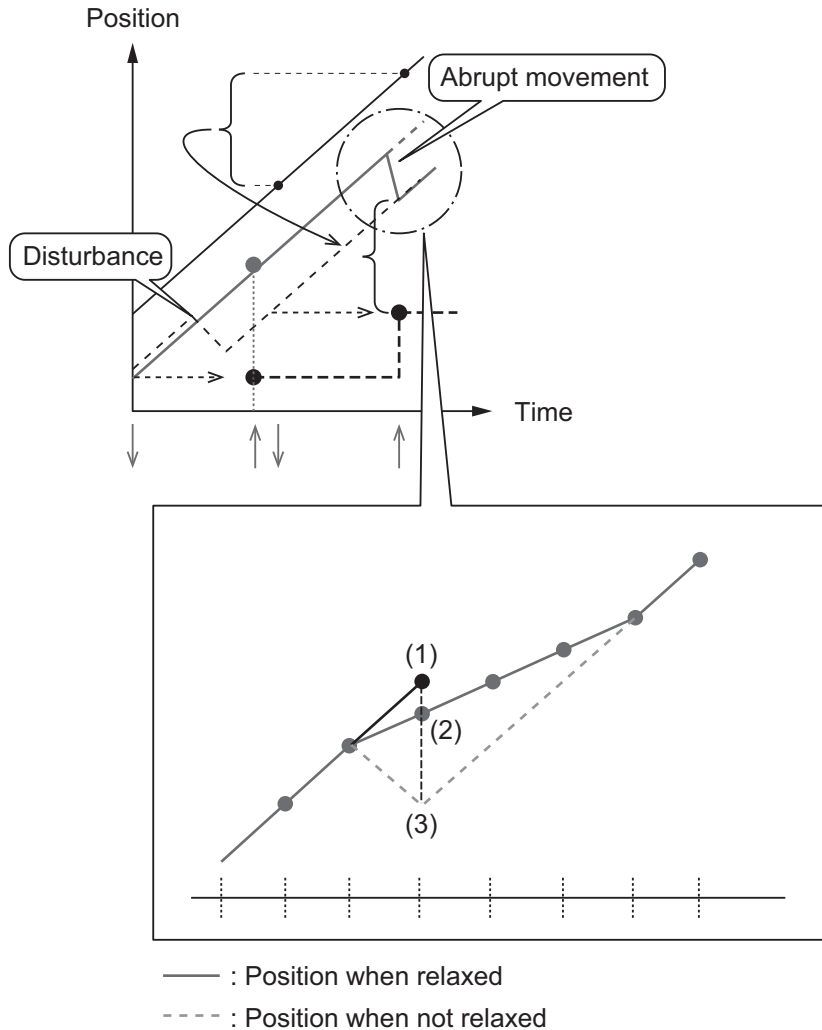
For PV generation, with respect to the travel distance calculated from the measurement result of the vision sensor, using the current value of the encoder at the time of imaging and the current value of the encoder when receiving the travel distance, perform acquisition of the imaging position, encoder interpolation, and mitigation measures to calculate PV.

- Mitigation measures

Consider that the movement may be abrupt due to disturbance factors such as vibration of measurement workpieces and the influence of the variation of the time from the output of the measurement trigger to the imaging (the difference per one task period of the measurement position after interpolation becomes larger than the most recent).

In the latter stage of the encoder interpolation, since axis control is performed by P control, the sharp movement of the measurement position after interpolation becomes abrupt movement of the axis.

In order to alleviate this behavior, a mitigation measure reflects the difference per one task period in the measurement position after interpolation immediately after the measurement end normally over multiple task periods rather than reflecting it in one task period.



It is assumed that the position (2) where the difference between the temporary position (1) and the position (3) in the case where the slope is not the same as the previous task period is divided by the number of distributions is relaxed.

This processing is done to reduce the occurrence of sudden velocity when trajectory calculation is not performed for P control.

In case of using the fifth order trajectory for trajectory computation for P control, this process is not executed because there is no sudden change in velocity.

b) Generate SP (Set point)

In order to perform high speed movement and smooth Start / Stop, generate SP using fifth order polynomial.

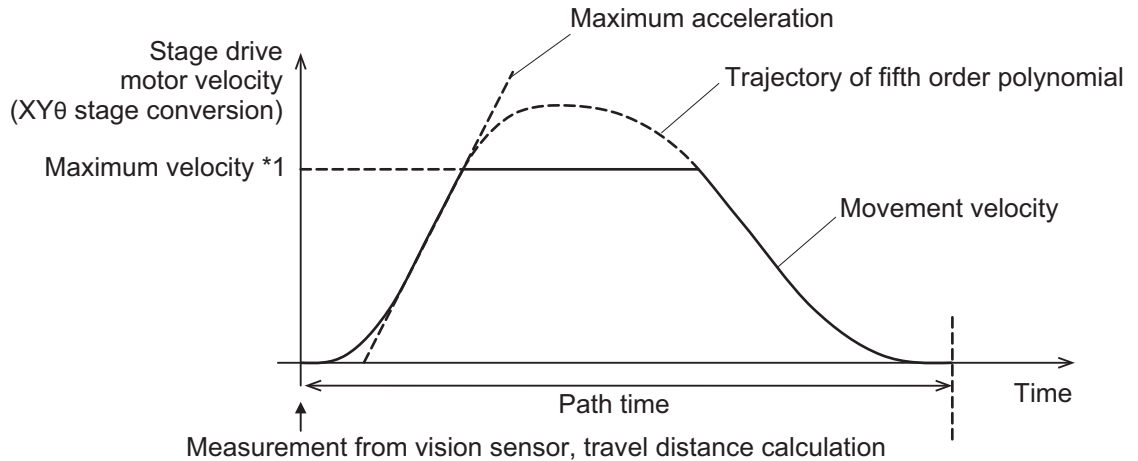
Use PV in Generate SP when the SP is used for a fifth order polynomial.

When generating an SP which uses a fifth order polynomial, set sCALCURVE_PARAMS.UseCurve to TRUE.

If fifth order polynomial is not used, calculate P control with SP set to 0.

The fifth order polynomial is generated as shown below using the set value of the axis motion parameter for the travel distance calculated from the measurement result of the vision sensor. The maximum acceleration is the upper limit of the acceleration specified by the axis motion parameter (sAXIS_MOVE_PARAMS.AccDec).

The path time is corrected to be sCALCURVE_PARAMS.MinCurveTime or more and sCALCURVE_PARAMS.MaxCurveTime or less.



*1. After generating fifth order polynomial, the maximum velocity is limited by the maximum velocity (MaxVel) specified by the axis motion parameter (sAXIS_MOVE_PARAMS). If the velocity is lower than the maximum velocity of the generated fifth order polynomial, the actual operation time will be longer than the specified path time.

SP is generated for measurement results from all cameras, and velocity command value is generated from PV and SP.

After updating the fifth order polynomial, the command value will be close to 0.

For this reason, we have a connecting velocity function to enable us to take over the previous velocity.

There is also a fifth order trajectory sequence function that prevents updating the trajectory during vision sensor imaging.

- Connecting velocity

When using this function, set sCALCURVE_PARAMS.UseConnectingVel to be TRUE.

- Fifth order trajectory sequence

This function is to prevent updating the trajectory while imaging with the vision sensor.

Specifically, there is a case where the next imaging starts immediately after the imaging is completed, and updating the fifth order trajectory after the imaging will increase the deviation only by the velocity limit described later.

For this reason, update fifth order trajectory after the shutter speed time elapses from trigger input of the next imaging. In other words, it does not update the trajectory until the shutter speed time elapses from the trigger input for the next imaging.

When using this function, set sCALCURVE_PARAMS.UseCurveSequence to be TRUE.

c) P Control

For generating axis velocity, calculate by applying proportional gain (P control) to the difference between PV and SP.

$$\text{Axis velocity} = (-\text{SP} + \text{PV}) \times \text{Proportional gain}$$

The proportional gain is the axis motion parameter (sAXIS_MOVE_PARAMS.Kp).

d) Velocity limit

According to the settings of the axis motion parameter (sAXIS_MOVE_PARAMS), the axis velocity calculated above can be limited.

When the maximum velocity limit is valid, the axis velocity for each task period never exceeds the set value.

When the maximum acceleration/deceleration speed limit is valid, the axis velocity increase or decrease for each task period never exceeds the set value.

If fifth order trajectory is enabled, the axis velocity for each task period will not exceed the image capture speed limit until the shutter speed time elapses from the trigger.

When multiple restrictions are valid, the limitation on the smallest velocity value is applied.

Limit name	Enable condition	Setting value
Maximum velocity limit	sAXIS_MOVE_PARAMS.UseMaxVel=TRUE	sAXIS_MOVE_PARAMS.MaxVel
Maximum Acceleration/Deceleration speed limit	sAXIS_MOVE_PARAMS.UseAccDec=TRUE	sAXIS_MOVE_PARAMS.AccDec
Velocity Limit at image capture	sCALCAXVEL_PARAMS.UseCurve=TRUE	sAXIS_MOVE_PARAMS.ImagingLimitVel

- Path time output

When fifth order trajectory is valid, it outputs the path time on the X axis, Y axis, θ axis calculated above.

When fifth order trajectory is invalid, 0 is output.

- In-Position Judgment by command position

In-Position Judgment is done by two methods; Diameter / Angle judgment and XY θ position judgment.

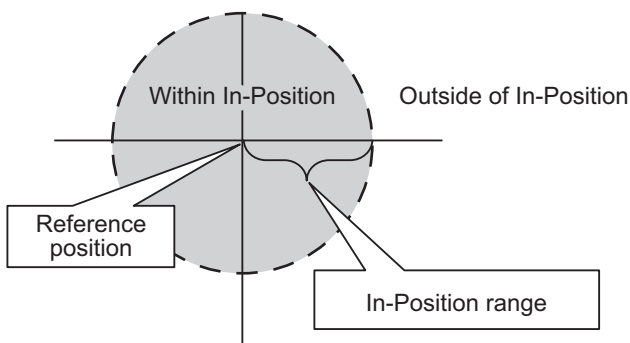
Set this with sCALCVIRTUALMOVE_PARAMS.InPosMode.

InpositionPV = TRUE when all targets set to TRUE in the judgment target (sCALCVIRTUALMOVE_PARAMS.InPosTarget) are within the In-Position range. If any one of them is outside of the In-Position range, InpositionPV = FALSE.

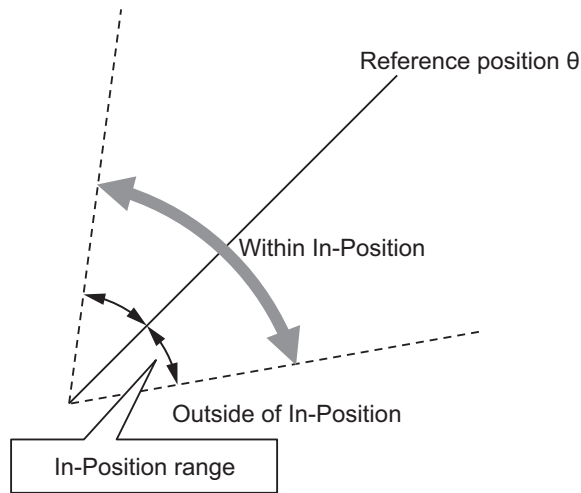
a) Diameter / Angle (InPosMode = 0)

Judgment done by diameter and angle.

Diameter In-Position Judgment is based on the reference mark position as a center, within the circle with the In-Position range (sCALCVIRTUALMOVE_PARAMS.InPosRange [0]) as the radius, if the command position is included, it is within the In-Position, when it is outside the circle it will be outside the In-Position.

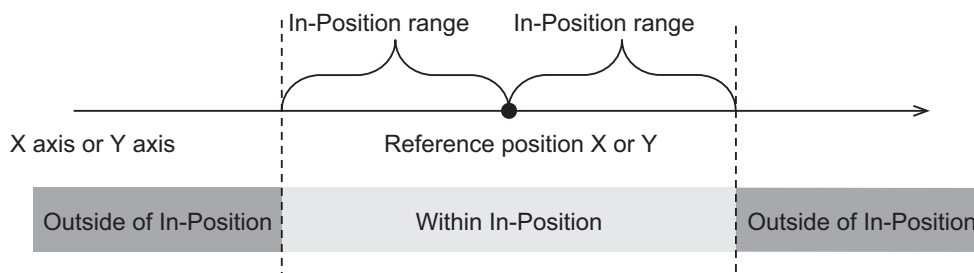


For the angle In-Position Judgment, if the θ Command value is less than or equal to the \pm In-Position range (sCALCVIRTUALMOVE_PARAMS.InPosRange[1]) with respect to the reference mark position θ value, it is within the In-Position. If it is larger than that range, it will be outside of the In-Position.



b) XY θ Position (InPosMode = 1)

As a reference mark position midpoint on each axis on virtual XY θ stage coordinates If the command position is within \pm In-Position Range, it is within the In-Position, when it is outside the range it will be outside the In-Position.



The example illustrating the θ axis is the same as the angle of InPos Mode = 0.

- Parameter setting

Please refer to *FB/FUN Structure Usage* on page 4 - 17 for the use of the members belonging to each structure.

- Changing input parameters

The alignment control parameter retains the value at the start of execution.

Changes during execution are not reflected in the calculation.

If you want to make the change effective, set Execute (Enable) to FALSE once and then set it to TRUE again.

Timing Charts

The timing charts are shown below.

- When Execute (Enable) is TRUE, begin axis velocity calculation.
- When Execute (Enable) is FALSE, end calculation.
- If an error occurs during execution of this function block, end calculation and Error changes to TRUE.

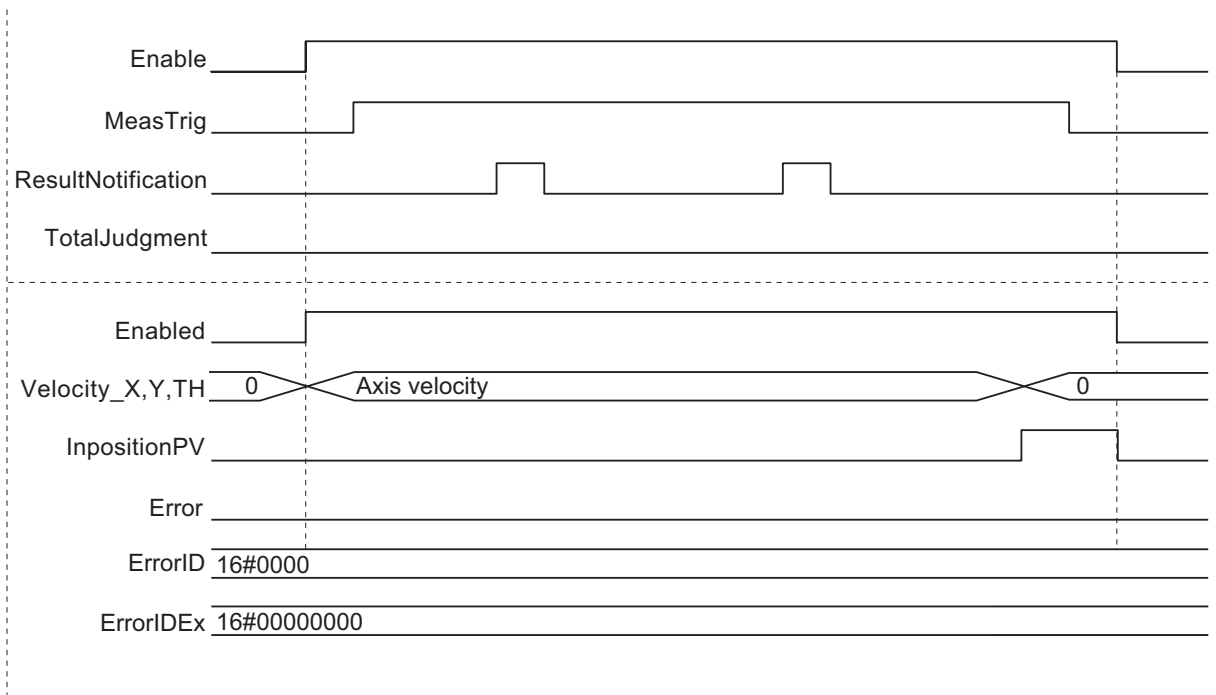
While in Error state, 0 is output for axis velocity.

You can find out the cause of the error by accessing the values output to Error Code (ErrorID) and Expansion Error Code (ErrorIDEx).

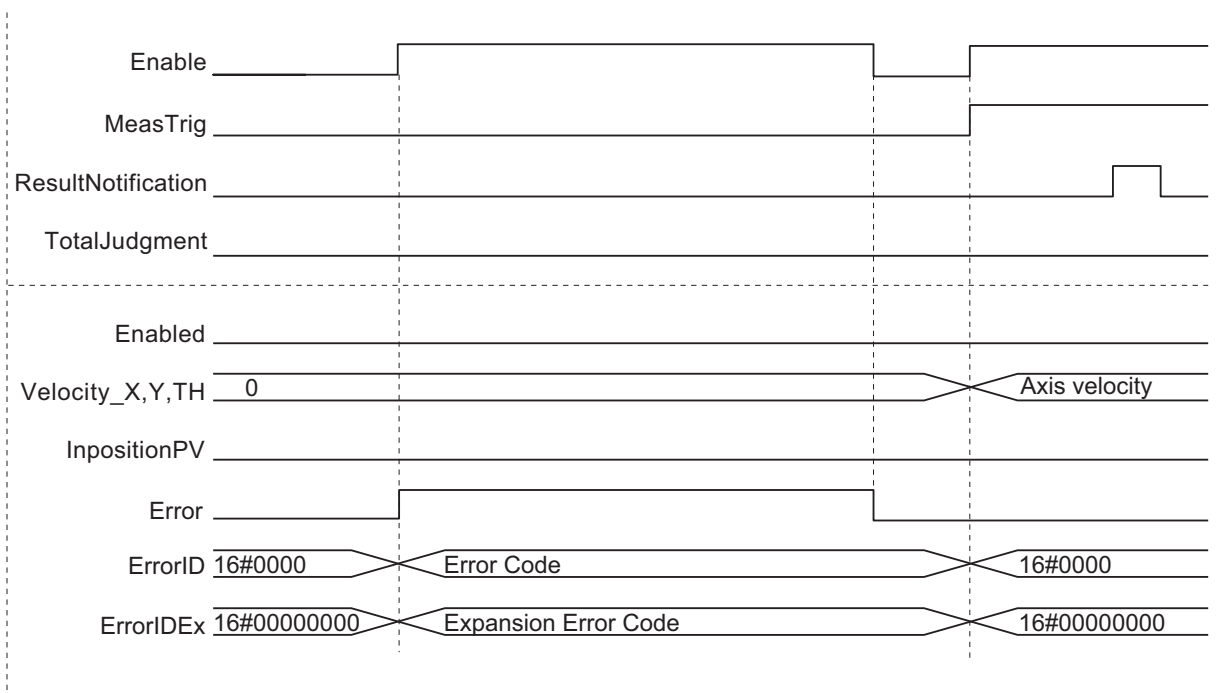
- Error (Error) is maintained while Execute (Enable) remains TRUE.

The Error Code (ErrorID) or Expansion Error Code (ErrorIDEx) is maintained until Execute is performed again.

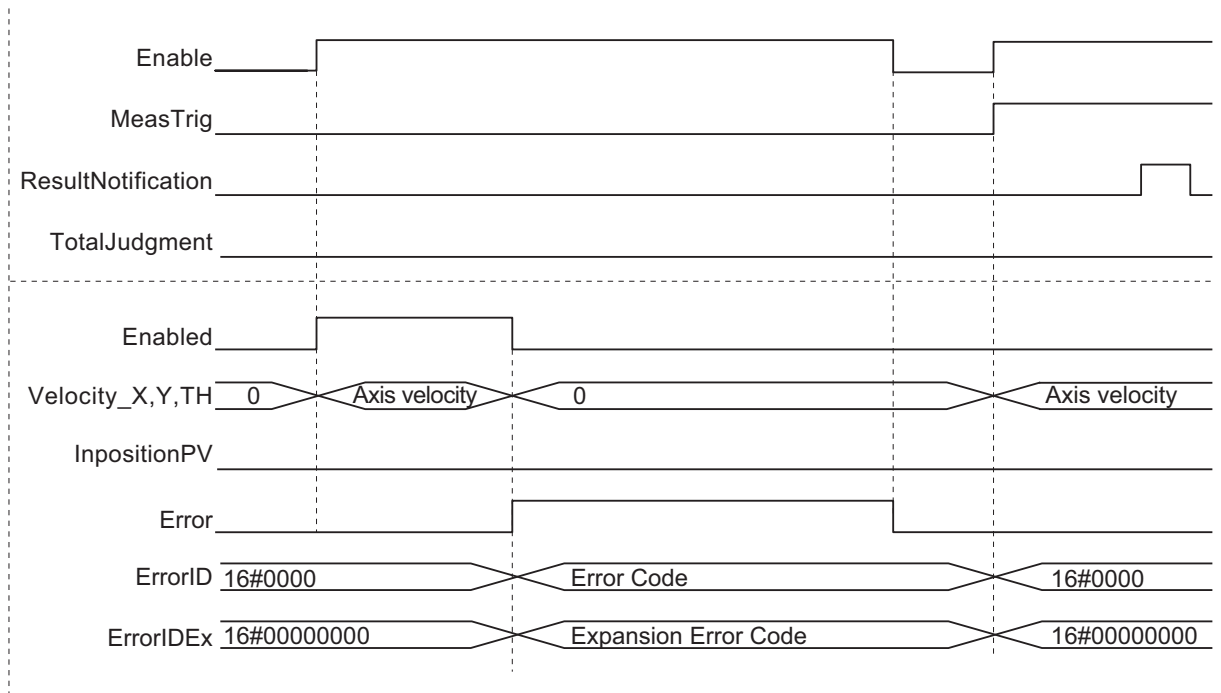
- Timing Chart for Normal End



- Timing Chart for Error End (Error at start of execution)



- Timing Chart for Error End (Error while execution is in progress)



Precautions for Correct Use

The following will result in an error.

TRUE is output for (Error) and InPositionPV FALSE is output when Velocity_X, Velocity_Y, Velocity_TH, CurveTime_X, CurveTime_Y, CurveTime_TH is 0.

- If an alignment control parameter outside of the valid range is set
- When the calculation result of axis velocity is non-numeric ($\pm\infty$)

Sample Programming

Please refer to *Sample Programming* on page 4 - 90.

Troubleshooting (Error Codes and Corrective Actions)

Error code	Expansion error code	Status	Description	Corrective action
16#0000	16#00000000	Normal end	—	—
16#3D0E	16#00000001	Invalid alignment control parameter	If an alignment control parameter outside of the valid range is set.	Correct the value entered for the alignment control parameter.
	16#00000002	Invalid axis speed calculation	The axis velocity calculation result is non-numeric ($\pm\infty$).	Revise such that the values for the axis travel distance, axis current position, XYθ axis motion parameter, and fifth order trajectory parameter can be calculated.

JudgeAlignmentComplete

In-Position Judgment performed for the measurement mark position with respect to the reference mark position. This is the means for judging completion of alignment control. Also, it is the judgment to determine whether to ultimately output the measurement trigger output from the Generate trigger FB (GenerateTrigger) to the vision sensor.

FB name	Name	FB/FUN	Graphic expression	ST expression
JudgeAlignmentComplete	Alignment control judgment	FB	<p style="text-align: center;">JudgeAlignmentComplete_instance</p>	<pre> JudgeAlignment- Complete_instance(Enable:=, TriggerInput:=, TriggerAck:=, TriggerBusy:=, ResultNotification:=, TotalJudgment:=, InpositionPV:=, InpositionAX:=, InputVelocityAX:=, Enabled=>, Trigger=>, CtrlComplete=>, OutputVelocityAX=>, Error=>, ErrorID=>, ErrorIDEx=>, TargetMarkPos- Stage:=, MeasMarkPos- Stage:=, AlignmentParams:=); </pre>

Function Block and Function Information

Item	Description
Library file name	OmronLib_VF_Alignment_Vx_x.slr (x indicates the version)
Namespace	OmronLib\VF_Alignment
Source code published/not published	Not Published
Function Block and Function number	00211

Variables

Input Variables

Input variable	Name	Data type	Default	Valid range	Description
Enable	Execute	BOOL	FALSE	TRUE, FALSE	TRUE: Execute FALSE: Do not execute
TriggerInput	Measurement trigger input	BOOL	FALSE	TRUE, FALSE	Enters the Trigger that is output by GenerateTrigger FB.
TriggerAck	Trigger acknowledged state	BOOL	FALSE	TRUE, FALSE	Input the Trigger acknowledged state signal (Trigger Ack) that is output from the vision sensor.
TriggerBusy	Busy	BOOL	FALSE	TRUE, FALSE	Input the Busy signal (Busy) that is output from the vision sensor.
ResultNotification	Data output complete	BOOL	FALSE	TRUE, FALSE	Enter the Data output complete signal (Result Notification) from the vision sensor.
TotalJudgment	Output Overall judgment	BOOL	FALSE	TRUE, FALSE	Enter the Overall judgment output signal (Total Judgment) from the vision sensor.
InpositionPV	Inposition according to the command position	BOOL	FALSE	TRUE, FALSE	Inputs CalcAxisVelocity.InpositionPV.
InpositionAX	Axis control In-Position	BOOL	FALSE	TRUE, FALSE	Inputs the logical product of all MC_SyncMoveAbsolute.InPosition that perform stage axis control.
InputVelocityAX	Input axis velocity	ARRAY[0..2] OF LREAL	0	Depends on data type.	Inputs the X, Y, θ axis velocities for the Virtual XY θ stage. For this input, enter the value of the output variable of CalcAxis Velocity Function.

Output Variables

Output variable	Name	Data type	Valid range	Description
Enabled	Executing	BOOL	TRUE, FALSE	It is TRUE while executing.
Trigger	Measurement trigger	BOOL	TRUE, FALSE	The output signal which becomes the input (Trigger) to the Execute measurement bit of the vision sensor.
CtrlComplete	Alignment control complete	BOOL	TRUE, FALSE	It becomes TRUE when Alignment control is completed based on the In-Position Judgment after Stop.
OutputVelocityAX	Output axis velocity	ARRAY[0..2] OF LREAL	Depends on data type.	The input axis velocity (InputVelocityAX) or zero [0] is output according to the Completion Judgment sequence.

Output variable	Name	Data type	Valid range	Description
Error	Error	BOOL	TRUE, FALSE	TRUE: Error end FALSE: Normal end, executing, or execution condition not met
ErrorID	Error Code	WORD	*1	This is the error ID for an error end. The value is 16#0000 for normal end.
ErrorIDEx	Expansion Error Code	DWORD	*1	This is the Extension Error ID at error end. The value is 16#00000000 for normal end.

*1. Refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 71 for details.

- The timing to refresh outputs

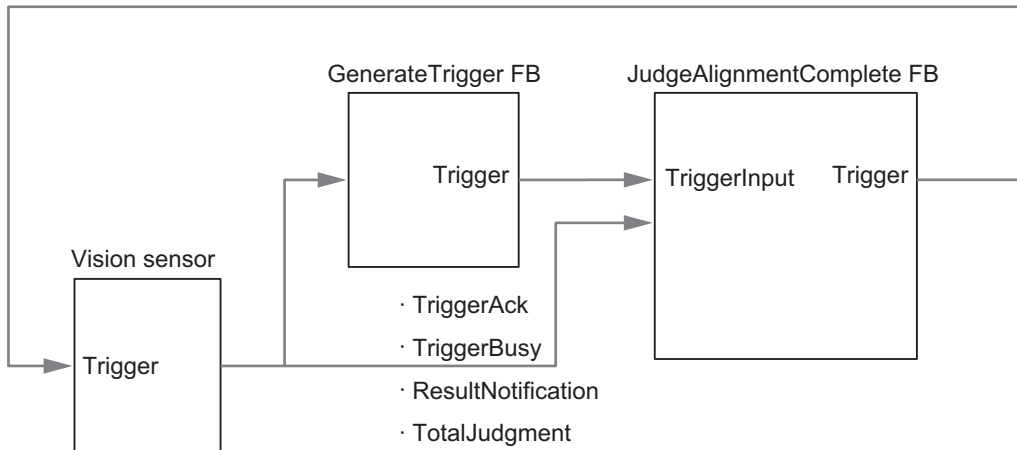
Variable	Condition for becoming TRUE	Condition for becoming FALSE
Enabled	<ul style="list-style-type: none"> • When Enable is TRUE 	<ul style="list-style-type: none"> • When Enable is FALSE • When CtrlComplete is TRUE • When Error is TRUE
Trigger	<ul style="list-style-type: none"> • When TriggerInput is TRUE • When the vision sensor is given command to measure 	<ul style="list-style-type: none"> • When Enable is FALSE • While TriggerInput is FALSE • When TriggerAck is FALSE • When Error is TRUE
CtrlComplete	<ul style="list-style-type: none"> • When In-Position Judgment is OK after Stop 	<ul style="list-style-type: none"> • When Enable is FALSE
Error	<ul style="list-style-type: none"> • At Error end 	<ul style="list-style-type: none"> • When Enable is FALSE

Input-Output Variables

Input-output variable	Name	Data type	Valid range	Description
Target-Mark-Pos-Stage	Reference mark position (Stage coordinates)	ARRAY[0..3] OF OmronLib \VF_Alignment\sPOSITION	—	Input the reference mark position using the stage coordinates. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array.
Meas-Mark-Pos-Stage	Measurement mark position (Stage coordinates)	ARRAY[0..3] OF OmronLib \VF_Alignment\sPOSITION	—	Input the measurement mark position using the stage coordinates. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array.
AlignmentParams	Alignment control parameter	OmronLib\VF_Alignment\sALIGNMENT_PARAMS	—	Input alignment control parameters. Refer to <i>Alignment Control Parameter (sALIGNMENT_PARAMS)</i> on page 4 - 2 for details.

Function

This FB is specially designed based on the vision sensor and Generate trigger FB (GenerateTrigger) and the following connection. When using this FB, make the following connection and use it.



In-Position Judgment performed for the measurement mark position with respect to the reference mark position. This is the means for judging completion of alignment control.

It performs Completion Judgment while execution (Enable) is TRUE. During execution, TRUE is output during execution (Enabled).

Execute this FB at the same time as alignment control is started and end alignment control when the Alignment control complete (CtrlComplete) output by this FB becomes TRUE.

The Completion Judgment consists of the following three stages.

- (1) In-Position Judgment before Stop
- (2) Stop Judgment
- (3) In-Position Judgment after Stop

The control and judgment functions at each stage are as follows.

- (1) In-Position Judgment before Stop

The judgment is started when InpositionPV judged by CalcAxisVelocity becomes TRUE.

For the conditions in which InpositionPV becomes TRUE, please refer to *Function* on page 4 - 57 in CalcAxisVelocity.

In-Position Judgment is performed from the reference mark position and the measurement mark position which is the judgment target (below) when the first measurement after the start of In-Position Judgment before Stop succeeds (ResultNotification = TRUE and CtrlComplete = FALSE).

If In-Position Judgment before Stop is OK, transition to Stop Judgment.

If the In-Position Judgment before Stop is NG, continue In-Position Judgment before Stop and wait for the measurement to succeed.

To the output axis velocity (OutputVelocityAX), the value of the input axis velocity (InputVelocityAX) is output as it is.

- In-Position Judgment Method

In-Position Judgment is done by two methods; Diameter / Angle judgment and XYθ position judgment.

Set this with sJUDGE_PARAMS.InPosMode.

Judgment is OK when all targets set to TRUE in the judgment target (sJUDGE_PARAMS.In-PosTarget) are within the In-Position range.

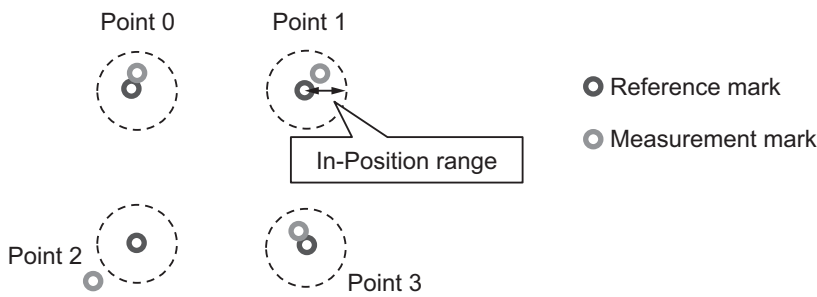
Please specify sJUDGE_PARAMS.TargetMark as the judgment target for reference / measurement marks for up to 4 points.

In-Position Judgment is performed only for points set to TRUE.

Points set to FALSE are not subject to the judgment and have no effect on the judgment result. If all are FALSE, a Parameter Error occurs.

As shown in the figure below, when alignment control is performed on an object in which only Point 2 of the measurement mark is different from the reference mark, it is possible to perform Stop Judgment using the setting below which takes Point 2 out of the Judgment target.

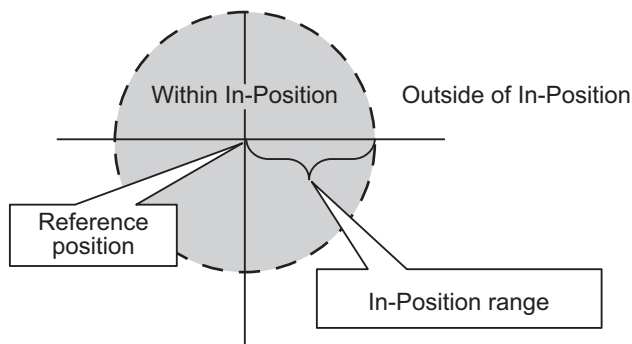
```
TargetMark[0]:=TRUE;
TargetMark[1]:=TRUE;
TargetMark[2]:=FALSE;
TargetMark[3]:=TRUE;
```



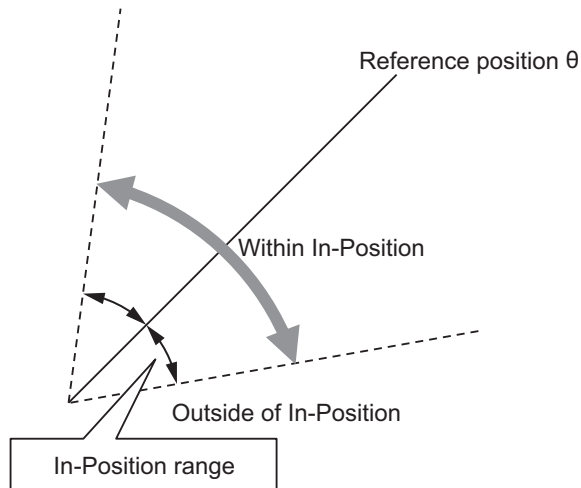
a) Diameter / Angle (InPosMode = 0)

Judgment done by diameter and angle.

For the In-Position Judgment of Diameter, when the XY coordinates of the measurement mark position are included in the circle whose center is the reference mark position XY coordinates and the In-Position range (sJUDGE_PARAMS.InPosRange [0]) is the radius, it is within the In-Position. If it is outside of that circle, it will be outside of the In-Position.

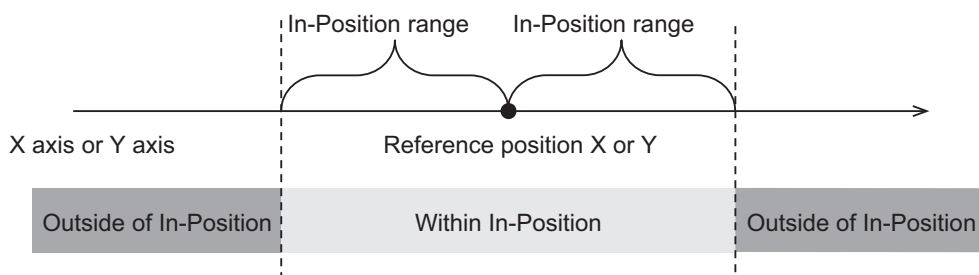


For the Angle In-Position Judgment, if the measurement mark position θ value is less than or equal to the \pm In-Position range (sJUDGE_PARAMS.InPosRange[1]) with respect to the reference mark θ position θ value, it is within the In-Position. If it is larger than that range, it will be outside of the In-Position.



b) XYθ Position (InPosMode = 1)

When the command position is included within \pm In-Position range around the reference mark position XY coordinates, it is within In-Position, and when it is outside the range, it will be outside the In-Position.



The example illustrating the θ axis is the same as the angle of InPos Mode = 0.

(2) Stop Judgment

It outputs zero to the output axis velocity (OutputVelocityAX) and waits for the axis stop.

For the Axis Stop judgment, use the In-Position Check function (setting: axis parameter - Operation Setting - In-Position range) of the MC Function Module.

Please refer to InPosition output of MC_SyncMoveAbsolute used for axis control and input TRUE to InPositionAX when InPosition of all axes becomes TRUE.

After Axis Stop is completed, TRUE is output to Trigger, and it transitions to In-Position Judgment after Stop.

(3) In-Position Judgment after Stop

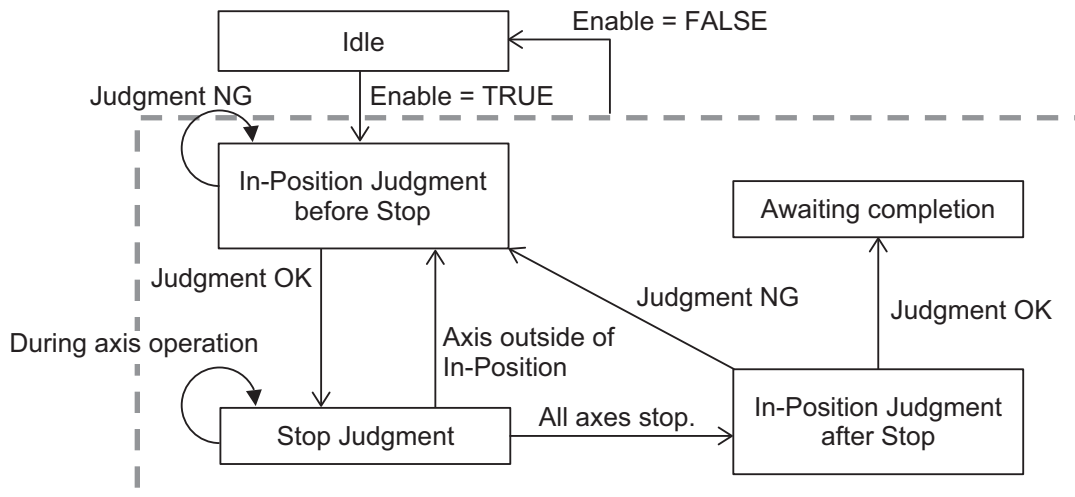
In-Position Judgment is performed from the reference mark position and the measurement mark position which is the judgment target when the first measurement after the start of In-Position Judgment after Stop is successful. (ResultNotification = TRUE and CtrlComplete = FALSE).

If Judgment is OK, TRUE is output to CtrlComplete.

Please terminate the alignment control.

In case of Judgment NG, (1) return to In-Position Judgment before Stop.

State transition diagram



After an OK judgment for In-Position Judgment after Stop, it will continue to wait until the execution (Enable) becomes FALSE.

During this time, even if the input variable changes, re-judgment will not be done.

To re-confirm the completion judgment, set execute (Enable) to FALSE and then set it to TRUE again.

When execute (Enable) is set to FALSE during the judgment, execution ends.

If execution (Enable) is set to TRUE again, the Completion Judgment is performed from Idle in the above state transition diagram to In-Position Judgment before Stop.

If the axis travel (InpositionPV = FALSE, InpositionAX = FALSE) is detected during the judgment of (2) and (3), it will transition to the In-Position Judgment before Stop and the judgment will be repeated from (1) In-Position Judgment before Stop.

- Measurement Trigger Output

While InpositionPV is FALSE, the value of TriggerInput input from Generate trigger FB (Generate-Trigger) is output to Trigger.

While InpositionPV is TRUE, the value of TriggerInput is ignored and Trigger becomes TRUE when measurement is required according to the above judgment processing.

- Parameter setting

Please refer to *Done Judgment Parameter (JUDGE_PARAMS)* on page 4 - 14 and *FB/FUN Structure Usage* on page 4 - 17 for the use of the members belonging to each structure.

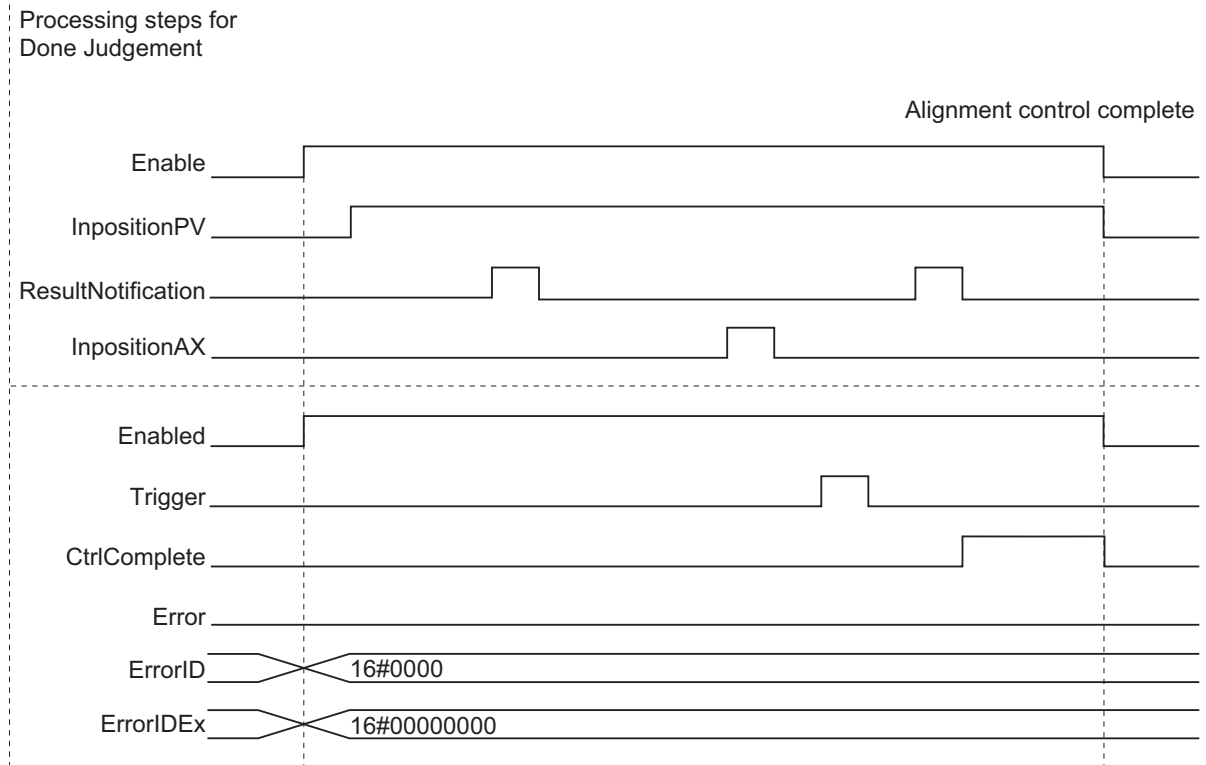
- Changing input parameters

The alignment control parameter retains the value at the start of execution.

Changes during execution are not reflected in the judgment.

If you want to make the change effective, set Execute (Enable) to FALSE once and then set it to TRUE again.

Timing Chart



Precautions for Correct Use

The following will result in an error.

TRUE is output for Error (Error) and FALSE is output for Enabled, Trigger and CtrlComplete.

- If an alignment control parameter outside of the valid range is set
- When all elements of the sJUDGE_PARAMS.TargetMark array are FALSE

Sample Programming

Please refer to *Sample Programming* on page 4 - 90.

Troubleshooting (Error Codes and Corrective Actions)

Error code	Ex-pan-sion error code	Status	Description	Corrective action
16#0000	16#00000000	Normal end	—	—
16#3D11	16#00000001	Invalid alignment control parameter	If an alignment control parameter outside of the valid range is set.	Correct the value entered for the alignment control parameter.

Error code	Expansion error code	Status	Description	Corrective action
	16#00 00000 2	Nothing to calculate	All elements of sJUDGE_PAR-AMS.TargetMark array are FALSE.	Please set so that at least one point can be calculated.

GenerateTrigger

A measurement trigger is generated and it instructs the vision sensor to perform a measurement.

FB name	Name	FB/FUN	Graphic expression	ST expression
GenerateTrigger	Trigger generation	FB		<pre>GenerateTrigger_instance(Enable:=, CurveTime_X:=, CurveTime_Y:=, CurveTime_TH:=, MeasDataRefresh:=, TriggerAck:=, TriggerBusy:=, ResultNotification:=, TotalJudgment:=,VirtualStagePos:= Enabled=>, Trigger=>, Error=>, ErrorID=>, ErrorIDEx=>, AlignmentParams:=);</pre>

Function Block and Function Information

Item	Description
Library file name	OmronLib_VF_Alignment_Vx_x.slr (x indicates the version)
Namespace	OmronLib_VF_Alignment
Source code published/not published	Not Published
Function Block and Function number	00212

Variables

Input Variables

Input variable	Name	Data type	Default	Valid range	Description
Enable	Execute	BOOL	FALSE	TRUE, FALSE	TRUE: Execute FALSE: Do not execute
CurveTime_X	X axis path time	LREAL	0.0	Depends on data type.	Unit: ms Input CalcAxisVelocity.CurveTime_X.

Input variable	Name	Data type	Default	Valid range	Description
CurveTime_Y	Y axis path time	LREAL	0.0	Depends on data type.	Unit: ms Input CalcAxisVelocity.CurveTime_Y.
CurveTime_TH	θ axis path time	LREAL	0.0	Depends on data type.	Unit: ms Input CalcAxisVelocity.CurveTime_TH.
MeasDataRefresh	Refresh measurement data	BOOL	FALSE	TRUE, FALSE	Input CalcAxisVelocity.MeasDataRefresh.
TriggerAck	Trigger acknowledged state	BOOL	FALSE	TRUE, FALSE	Input the Trigger acknowledged state signal (Trigger Ack) that is output from the vision sensor.
TriggerBusy	Busy	BOOL	FALSE	TRUE, FALSE	Input the Busy signal (Busy) that is output from the vision sensor.
ResultNotification	Data output complete	BOOL	FALSE	TRUE, FALSE	Enter the Data output complete signal (Result Notification) that is output from the vision sensor.
TotalJudgment	Output Overall judgment	BOOL	FALSE	TRUE, FALSE	Enter the Overall judgment output signal (Total Judgment) that is output from the vision sensor.
VirtualStagePos	Virtual stage position	OmronLib \\WF_Alignment\\sPOSITION	—	—	Enter the stage position in the Virtual XYθ Stage coordinates.

Output Variables

Output variable	Name	Data type	Valid range	Description
Enabled	Executing	BOOL	TRUE, FALSE	It is TRUE while executing.
Trigger	Measurement trigger	BOOL	TRUE, FALSE	The output signal which becomes the input (Trigger) to the Execute measurement bit of the vision sensor.
Error	Error	BOOL	TRUE, FALSE	TRUE: Error end FALSE: Normal end, executing, or execution condition not met
ErrorID	Error Code	WORD	*1	This is the error ID for an error end. The value is 16#0000 for normal end.
ErrorIDEx	Expansion Error Code	DWORD	*1	This is the Extension Error ID at error end. The value is 16#00000000 for normal end.

*1. Refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 77 for details.

- The timing to refresh outputs

Variable	Condition for becoming TRUE	Condition for becoming FALSE
Enabled	<ul style="list-style-type: none"> When Enable is TRUE 	<ul style="list-style-type: none"> When Enable is FALSE When Error is TRUE
Trigger	<ul style="list-style-type: none"> When measurement instruction is required and the Generate trigger condition is satisfied 	<ul style="list-style-type: none"> When Enable is FALSE When TriggerAck is TRUE When Error is TRUE
Error	<ul style="list-style-type: none"> At error end 	<ul style="list-style-type: none"> When Enable is FALSE

Input-Output Variables

Input-output variable	Name	Data type	Valid range	Description
AlignmentParams	Alignment control parameter	OmronLib\VF_Alignment\sALIGNMENT_PARAMS	—	Input alignment control parameters. Refer to <i>Alignment Control Parameter (sALIGNMENT_PARAMS)</i> on page 4 - 2 for details.

Function

A measurement trigger is generated and it instructs the vision sensor to perform a measurement.

Generate Trigger is done while Execute (Enable) is TRUE.

During execution, TRUE is output for Executing (Enabled).

If execution (Enable) is FALSE, FALSE is output to the measurement trigger (Trigger).

The Generate trigger condition changes depending on whether fifth order trajectory sequence is used (sCALCURVE_PARAMS.UseCurveSequence = TRUE) or not.

- When not using fifth order trajectory sequence

TRUE is output to Trigger when all of the following three Generate trigger conditions are satisfied.

Condition 1: The axis velocity of the XYθ axes is not more than the set value

All axis velocities on the X axis, Y axis, and θ axis calculated from the amount of change in the position on the Virtual XYθ Stage coordinates are less than the image capture speed limit (see table below) of each axis.

Axis	Image capture speed limit parameter
X	sCALCVIRTUALMOVE_PARAMS.MoveParams_X.ImagingLimitVel
Y	sCALCVIRTUALMOVE_PARAMS.MoveParams_Y.ImagingLimitVel
θ	sCALCVIRTUALMOVE_PARAMS.MoveParams_TH.ImagingLimitVel

For the image capture speed limit, please set the following calculated value as a guide against the target accuracy of alignment and camera exposure time (1/shutter speed).

Image capture speed limit (ImagingLimitVel) = target accuracy × 20/camera exposure time

Condition 2: TriggerAck = FALSE

Condition 3: TriggerBusy = FALSE

- When using fifth order trajectory sequence

When using the fifth order trajectory sequence, since it tends to accelerate until 1/2 of the path time elapses, limit imaging so it is not done during this time.

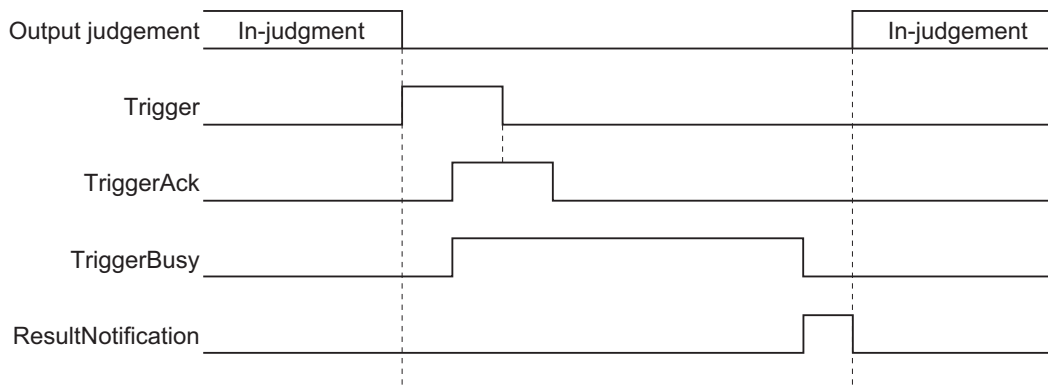
Therefore, in addition to the Generate trigger conditions 1, 2, and 3 when not using fifth order trajectory sequence, the following Generate trigger condition 4 is added.

Condition 4: The maximum value of the trajectory time and the minimum trajectory time (sCAL-CURVE_PARAMS.MinCurveTime) of each of the XYθ axes input to this FB is taken as the maximum trajectory time and one half of the maximum trajectory time has elapsed from the Refresh measurement data (MeasDataRefresh = TRUE)

- Re-generate triggers

After satisfying the trigger output condition, when TriggerAck = TRUE, set the output of the measurement trigger (Trigger) to FALSE.

After that, when ResultNotification = TRUE, return to judgment of the Generate trigger condition, and when the Generate trigger condition is satisfied again set the output of the measurement trigger (Trigger) to TRUE.



- Parameter setting

Please refer to *FB/FUN Structure Usage* on page 4 - 17 for the use of the members belonging to each structure.

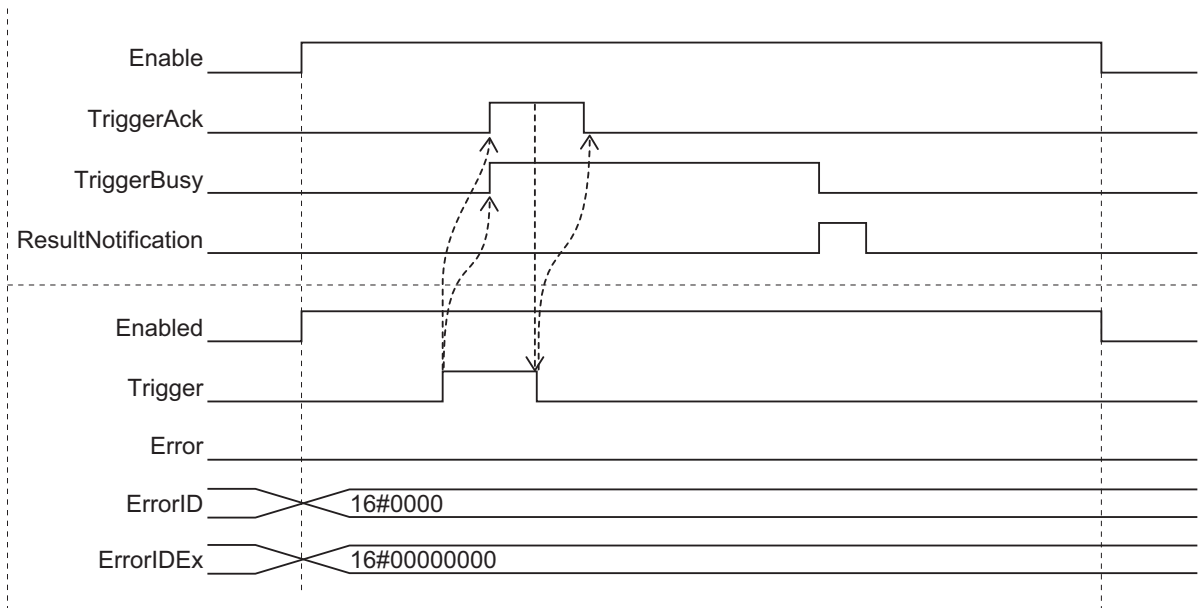
- Changing input parameters

The alignment control parameter retains the value at the start of execution.

Changes during execution are not reflected in the judgment.

If you want to make the change effective, set Execute (Enable) to FALSE once and then set it to TRUE again.

Timing Chart



Precautions for Correct Use

The following will result in an error.

TRUE is output for Error (Error) and FALSE is output for Trigger.

- If an alignment control parameter outside of the valid range is set

Sample Programming

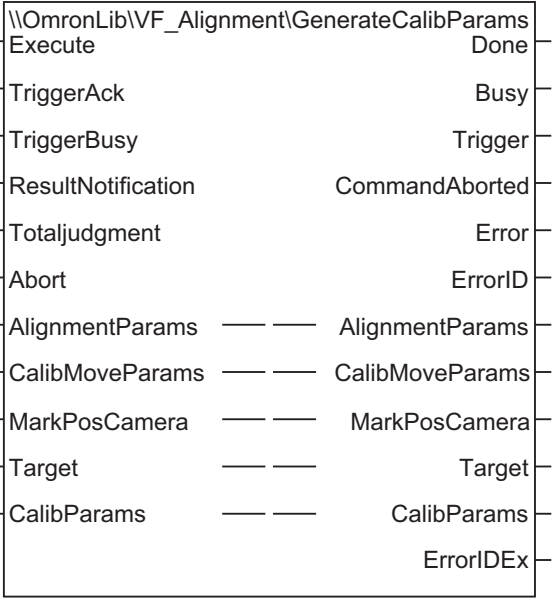
Please refer to *Sample Programming* on page 4 - 90.

Troubleshooting (Error Codes and Corrective Actions)

Error code	Ex-pan-sion error code	Status	Description	Corrective action
16#0000	16#0000000	Normal end	—	—
16#3D12	16#0000001	Invalid alignment control parameter	If an alignment control parameter outside of the valid range is set.	Correct the value entered for the alignment control parameter.

GenerateCalibParams

Generate calibration parameters used for coordinate conversion (AffineTrans).
By executing this instruction, it is possible to generate up to 4 mark positions.

FB name	Name	FB/FUN	Graphic expression	ST expression
GenerateCalibParams	Generate calibration parameter	FB		<pre>GenerateCalibParams_instance(Execute:=, TriggerAck:=, TriggerBusy:=, ResultNotification:=, TotalJudgment:=, Abort:=, Done=>, Busy=>, Trigger=>, CommandAborted=>, Error=>, ErrorID=>, ErrorIDEx=>, AlignmentParams:=, CalibMoveParams:=, MarkPosCamera:=, Target:=, CalibParams:=);</pre>

Function Block and Function Information

Item	Description
Library file name	OmronLib_VF_Alignment_Vx_x.slr (x indicates the version)
Namespace	OmronLib\F_VF_Alignment
Source code published/not published	Not Published
Function Block and Function number	00210

Variables

Input Variables

Input variable	Name	Data type	Default	Valid range	Description
Execute	Execute	BOOL	FALSE	TRUE, FALSE	TRUE: The function is executed

Input variable	Name	Data type	Default	Valid range	Description
TriggerAck	Trigger acknowledged state	BOOL	FALSE	TRUE, FALSE	Input the Trigger acknowledged state signal (Trigger Ack) that is output from the vision sensor.
TriggerBusy	Busy	BOOL	FALSE	TRUE, FALSE	Input the Busy signal (Busy) that is output from the vision sensor.
ResultNotification	Data output complete	BOOL	FALSE	TRUE, FALSE	Enter the Data output complete signal (Result Notification) that is output from the vision sensor.
TotalJudgment	Output Overall judgment	BOOL	FALSE	TRUE, FALSE	Enter the Overall judgment output signal (Total Judgment) that is output from the vision sensor.
Abort	Abort	BOOL	FALSE	TRUE, FALSE	When set to TRUE, generate calibration parameter is aborted.

Output Variables

Output variable	Name	Data type	Valid range	Description
Done	Done	BOOL	TRUE, FALSE	TRUE when Generate calibration parameter is completed.
Busy	Executing	BOOL	TRUE, FALSE	TRUE when the instruction is acknowledged.
Trigger	Measurement trigger	BOOL	TRUE, FALSE	The output signal which becomes the input (Trigger) to the Execute measurement bit of the vision sensor.
CommandAborted	Command Aborted	BOOL	TRUE, FALSE	TRUE when the instruction is aborted.
Error	Error	BOOL	TRUE, FALSE	TRUE: Error end FALSE: Normal end, executing, or execution condition not met
ErrorID	Error Code	WORD	*1	This is the error ID for an error end. The value is 16#0000 for normal end.
ErrorIDEx	Expansion Error Code	DWORD	*1	This is the Extension Error ID at error end. The value is 16#00000000 for normal end.

*1. Refer to *Troubleshooting (Error Codes and Corrective Actions)* on page 4 - 88 for details.

- The timing to refresh outputs

Variable	Condition for becoming TRUE	Condition for becoming FALSE
Done	<ul style="list-style-type: none"> • Generate calibration parameter completed 	<ul style="list-style-type: none"> • When Execute is FALSE When Execute is already FALSE on completion, after one task period.

Variable	Condition for becoming TRUE	Condition for becoming FALSE
Busy	<ul style="list-style-type: none"> When Execute is TRUE 	<ul style="list-style-type: none"> When Done is TRUE When CommandAborted is TRUE When Error is TRUE <p>However, if the axis is in motion, TRUE continues to be output until Axis stop is completed.</p>
Trigger	<ul style="list-style-type: none"> When the vision sensor is given command to measure 	<ul style="list-style-type: none"> When TriggerAck is TRUE When CommandAborted is TRUE When Error is TRUE
CommandAborted	<ul style="list-style-type: none"> When Abort is TRUE When TotalJudgment is TRUE When CommandAborted occurs in the internal axis control FB 	<ul style="list-style-type: none"> When Execute is FALSE <p>When Execute is already FALSE on completion, after one task period.</p>
Error	<ul style="list-style-type: none"> At Error end 	<ul style="list-style-type: none"> When Execute is FALSE <p>When Execute is already FALSE on completion, after one task period.</p>

Input-Output Variables

Input-output variable	Name	Data type	Valid range	Description
AlignmentParams	Alignment control parameter	OmronLib\VF_Alignment\sALIGNMENT_PARAMS	—	Input alignment control parameters. Refer to <i>Alignment Control Parameter (sALIGNMENT_PARAMS)</i> on page 4 - 2 for details.
CalibMoveParams	Calibration axis motion parameter	OmronLib\VF_Alignment\sCALIB_MOVE_PARAMS	—	Set calibration axis motion parameters. Refer to <i>Calibration Axis Motion Parameter (sCALIB_MOVE_PARAMS)</i> on page 4 - 15 for details.
MarkPosCamera	Mark position (Camera coordinates)	ARRAY[0..3] OF OmronLib\VF_Alignment\sPOSITION	—	Enter the mark position output from the vision sensor. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array.
Target	Calibration target	ARRAY[0..3] OF BOOL	TRUE, FALSE	Set the points for which to generate the calibration parameters. Set the point 0, point 1, point 2, point 3 in this order from the beginning of the array. TRUE: Calculate FALSE: Do not calculate
CalibParams	Calibration parameter	ARRAY[0..3] OF OmronLib\VF_Alignment\sCALIB_PARAMS	—	Outputs the calibration parameters for each point. It is output from the top of the array in order of point 0, point 1, point 2, point 3.

Function

Generate calibration parameters used for coordinate conversion (AffineTrans).

Since this FB performs axis control, it can not be executed simultaneously with alignment control.

This FB is an equivalent function to the "Vision Master Calibration" processing item of the vision sensor, except for the following differences.

Item		Vision Master Calibration	Generate calibration parameter FB
External device setting		Supports 8 stage types and 2 robot types	Supports 8 stage types
Number of calibration data segments		Maximum 8 data	Maximum 4 data
Input method for mark position (Camera coordinates)		Input by arithmetic expression	Use the input value as it is
Output method for amount of movement		Select from absolute position or relative position	No selection
Trapezoidal distortion, Lens distortion		Applicable	Cannot be used
Select homing method		Select Enable/Disable	No selection Operate disabled
Initial calibration		Yes Process for finding the range of movement for this calibration	Not supported The range of movement for this calibration is set according to the input parameter.
This calibration	Effective range of the field of view	1 to 100%	No setting It operates within the range on the axis coordinate specified by the calibration axis motion parameter (sCALIB_MOVE_PARAMS)
	Sampling method	Data one by one / All data at same time	No selection All data is applied at the same time By executing this FB multiple times, it is possible to realize the same operation as data one by one
	Parallel movement sampling setting, rotational movement sampling setting	Number of distribution rows (2 to 10), number of distribution columns (2 to 10), number of distribution points (2 to 100)	Total travel distance (X, Y, θ) Number of distribution rows at X axis & Y axis operation, Number of distribution columns at θ axis operation is 9 points, Number of distribution points fixed at 11 points
	Advanced settings	Stop criteria for rotation sampling No. of points for error calculation Upper limit of error evaluation value	None

A calibration parameter is generated for the point set as TRUE in the calibration target (Target). Generate is not done for points set to FALSE, and the calibration parameters do not change.

Start (Execute) is set to TRUE to start Generate calibration parameter.

It is TRUE while executing (Busy).

When Generate is completed, Busy is FALSE and Completed (Done) is TRUE.

Even if FALSE (Execute) is set to FALSE or TRUE again during execution, this FB will not be canceled or re-executed. It will continue to run until it is done (Done), aborted (CommandAborted), or there is an error end (Error).

- Generate calibration parameter

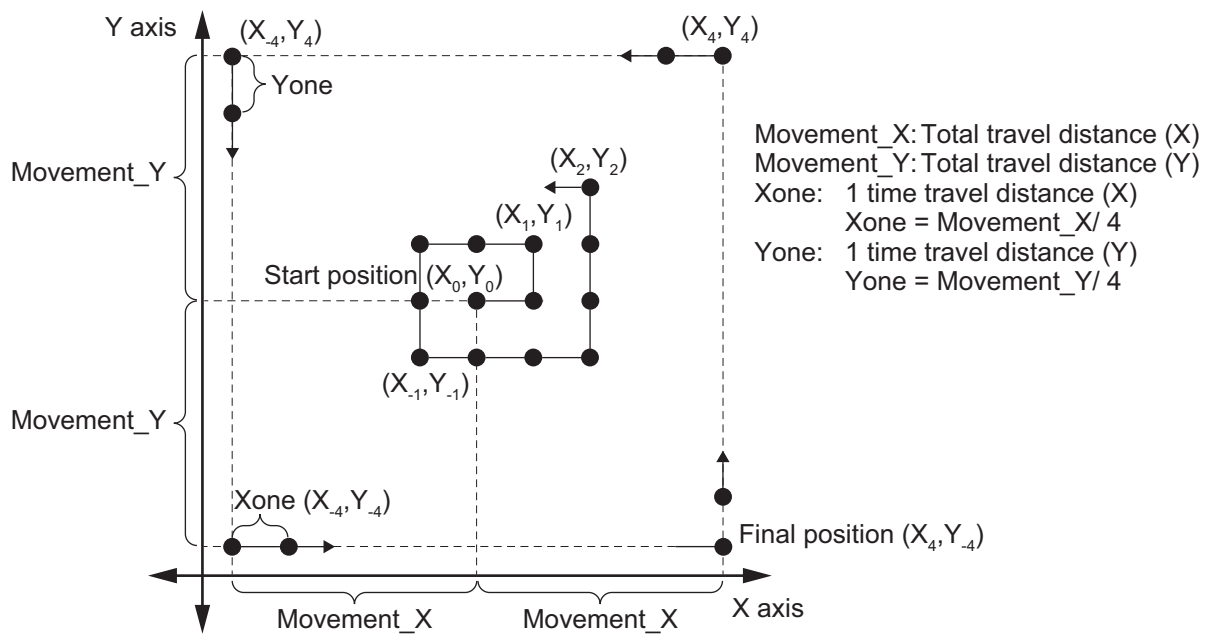
Based on the starting position, move the X axis, Y axis, and θ axis as the relative amount of each axis specified by the calibration axis motion parameter (sCALIB_MOVE_PARAMS) as the relative amount.

Start position → X axis & Y axis movement → Start position → θ axis movement → Start position.

This is the movement.

By setting CalibMoveParams.WaitTime, it is possible to reduce the time from after travel operation complete to the imaging, attenuate the vibration caused by the movement, and to reduce the influence of vibration on the image.

- X axis & Y axis movement

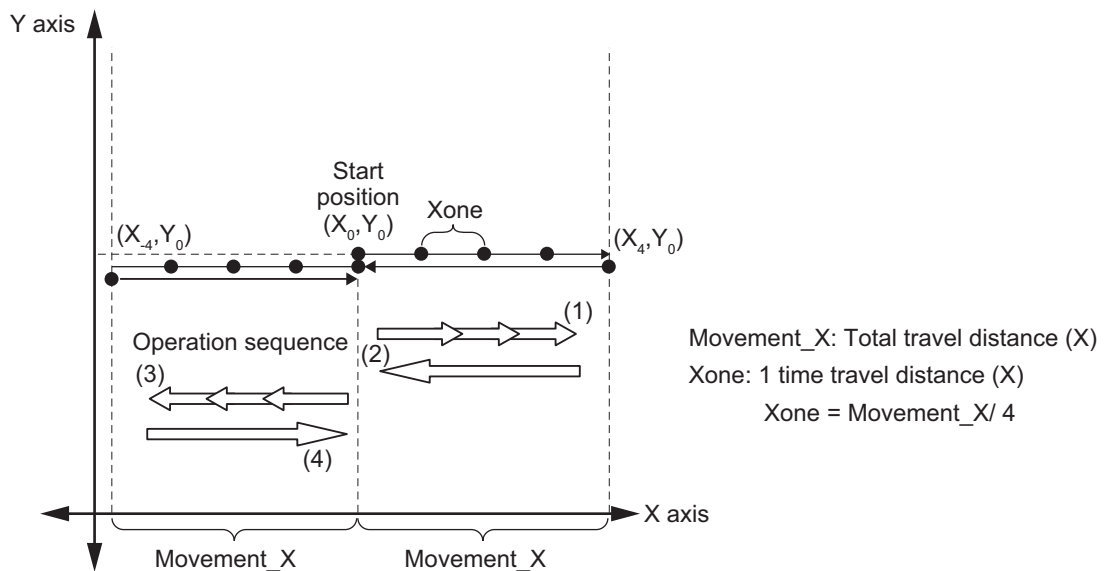


Axis coordinates	Relative amount based on starting position		Notes
	X axis	Y axis	
(X_0, Y_0)	0	0	Start position
(X_1, Y_0)	Xone	0	
(X_1, Y_1)	Xone	Yone	
(X_0, Y_1)	0	Yone	
(X_{-1}, Y_1)	-Xone	Yone	
(X_{-1}, Y_0)	-Xone	0	
(X_{-1}, Y_{-1})	-Xone	-Yone	
(X_0, Y_{-1})	0	-Yone	
(X_1, Y_{-1})	Xone	-Yone	
(X_2, Y_{-1})	$Xone \times 2$	-Yone	
(X_2, Y_0)	$Xone \times 2$	0	
(X_2, Y_1)	$Xone \times 2$	Yone	
(X_2, Y_2)	$Xone \times 2$	$Yone \times 2$	
:			

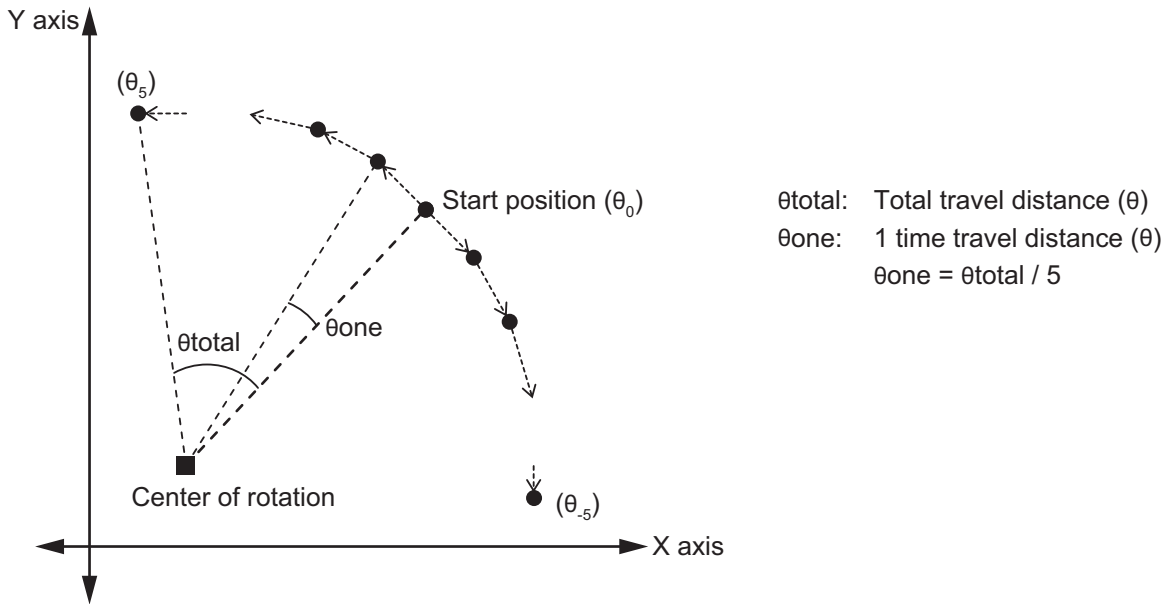
Axis coordinates	Relative amount based on starting position		Notes
	X axis	Y axis	
(X ₄ , Y ₋₃)	Xtotal	-Yone × 3	
:			
(X ₄ , Y ₄)	Xtotal	Ytotal	X positive direction end, Y positive direction end
(X ₃ , Y ₄)	Xone × 3	Ytotal	
:			
(X ₋₄ , Y ₄)	-Xtotal	Ytotal	X negative direction end, Y positive direction end
(X ₋₄ , Y ₃)	-Xtotal	Yone × 3	
:			
(X ₋₄ , Y ₋₄)	-Xtotal	-Ytotal	X negative direction end, Y negative direction end
(X ₋₃ , Y ₋₄)	-Xone × 3	-Ytotal	
:			
(X ₄ , Y ₋₄)	Xtotal	-Ytotal	Final position X positive direction end, Y negative direction end
(X ₀ , Y ₀)	0	0	Return to Start position

a) When the selected stage is an X(Y) stage (X, Y, Xθ, Yθ, θX, θY)

The operation of designating the stage (X, Xθ, θX) where only the X axis exists is described below.



- θ axis movement



Axis coordinates	Relative amount based on starting position	Notes
	θ axis	
(θ_0)	0	Start position
(θ_1)	θ_{one}	
(θ_2)	$\theta_{one} \times 2$	
:		
(θ_k)	θ_{total}	Positive direction end
(θ_0)	0	
(θ_{-1})	$-\theta_{one}$	
(θ_{-2})	$-\theta_{one} \times 2$	
:		
(θ_{-k})	$-\theta_{total}$	Final position Negative direction end
(θ_0)	0	Return to Start position

- a) Output data
Outputs a measurement trigger (Trigger) to be input to the vision sensor at the timing necessary for generating the calibration parameters.
- b) Command Aborted due to measurement result (Overall Judgment NG)
When the input to the Output Overall judgment (TotalJudgment) is TRUE, execution of this FB is aborted, all controlled axes are decelerated to a stop, and TRUE is output as Command Aborted (CommandAborted).
- c) Arbitrary Command Aborted
If you want to abort the execution of this FB, set Abort to TRUE.
Decelerates all controlled axes to a stop, aborts Generate calibration parameters, and outputs TRUE to Command Aborted (CommandAborted).
Command Aborted (CommandAborted) is output while Execute (Execute) is TRUE.
When Execute (Execute) is FALSE, TRUE is output only after one task period.
To re-execute, set Execute (Execute) once to FALSE and then set it to TRUE.
- d) Command Aborted initiated by other axis control commands

- During execution of this FB, do not execute other axis control commands on the axis used in the stage.
- When axis control is performed by another command, it decelerates all controlled axes to a stop, aborts Generate calibration parameters, and outputs TRUE to Command Aborted (CommandAborted).
- Command Aborted (CommandAborted) is output while Execute (Execute) is TRUE.
- When Execute (Execute) is FALSE, TRUE is output only after one task period.
- To re-execute, set Execute (Execute) once to FALSE and then set it to TRUE.
- e) Command Aborted due to error condition
- When an error occurs, it decelerates all controlled axes to a stop, aborts Generate calibration parameters, and outputs TRUE to error (Error).
- Please refer to *Precautions for Correct Use* on page 4 - 87 for the conditions that cause errors.
- To re-execute, set Execute (Execute) to FALSE, correct the cause of the error and set Execute (Execute) to TRUE again.
- f) Axis deceleration stop
- If a Command Aborted occurs during execution, this FB uses the MC_Stop command to decelerate the controlled axes to a stop.
- SCALIB_MOVE_PARAMS.StopDec is applied to the deceleration rate (MC_Stop.Deceleration) at this time.
- If an error (Error) or Command Aborted (CommandAborted) occurs in the MC_MoveAbsolute instruction and the MC_Stop instruction during deceleration stop, immediately stop with the MC_Stop instruction.
- g) Parameter setting
- Please refer to *FB/FUN Structure Usage* on page 4 - 17 for the use of the members belonging to each structure.
- h) Changing input parameters
- Alignment control parameters, calibration axis motion parameters, and calibration targets at the start of execution are retained.
- Changes during execution are not reflected in the calculation.
- If you want to make changes effective, please set it to TRUE again after completion.

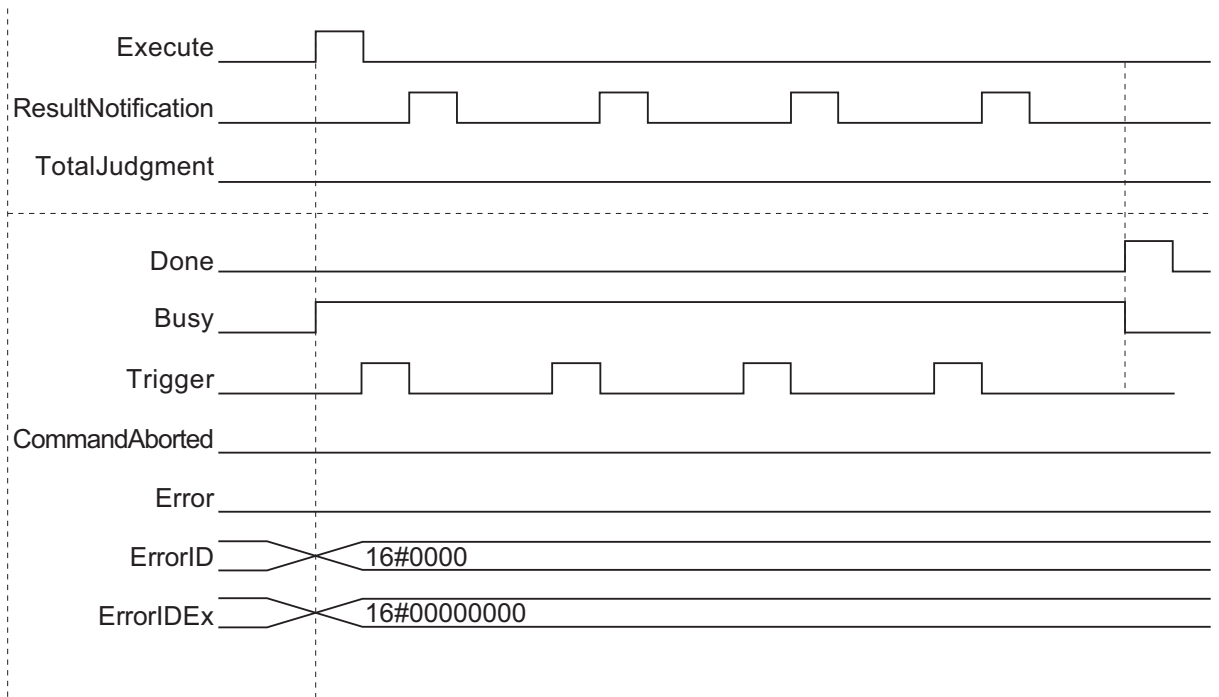
Timing Charts

The timing charts are shown below.

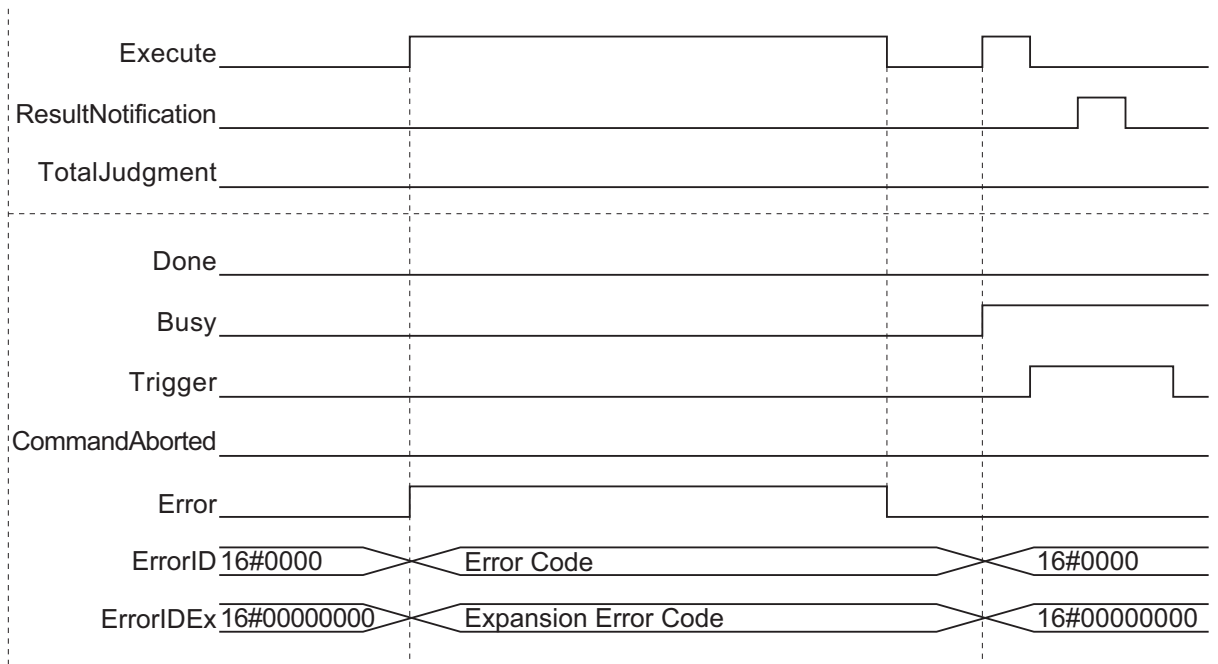
- Executing (Busy) becomes TRUE at the start of Execute (Execute), and generate calibration parameters is started.
 - When generate calibration parameters is completed, Done (Done) becomes TRUE.
 - If an error occurs during execution of this function block, end calculation and Error changes to TRUE.
- Generate calibration parameter will not be performed while in an error state.
- You can find out the cause of the error by accessing the values output to Error Code (ErrorID) and Expansion Error Code (ErrorIDEx).
- While error (Error) is TRUE, it will stay in the error (Error) state.
- The Error Code (ErrorID) or Expansion Error Code (ErrorIDEx) is maintained until Execute is performed again.
- If another axis control command is executed while this FB is being executed, Command Aborted (CommandAborted) becomes TRUE.

- While Execute (Execute) is TRUE, Command Aborted (CommandAborted) will be maintained. Generate calibration parameter will not be performed while in Command Aborted (CommandAborted) state.

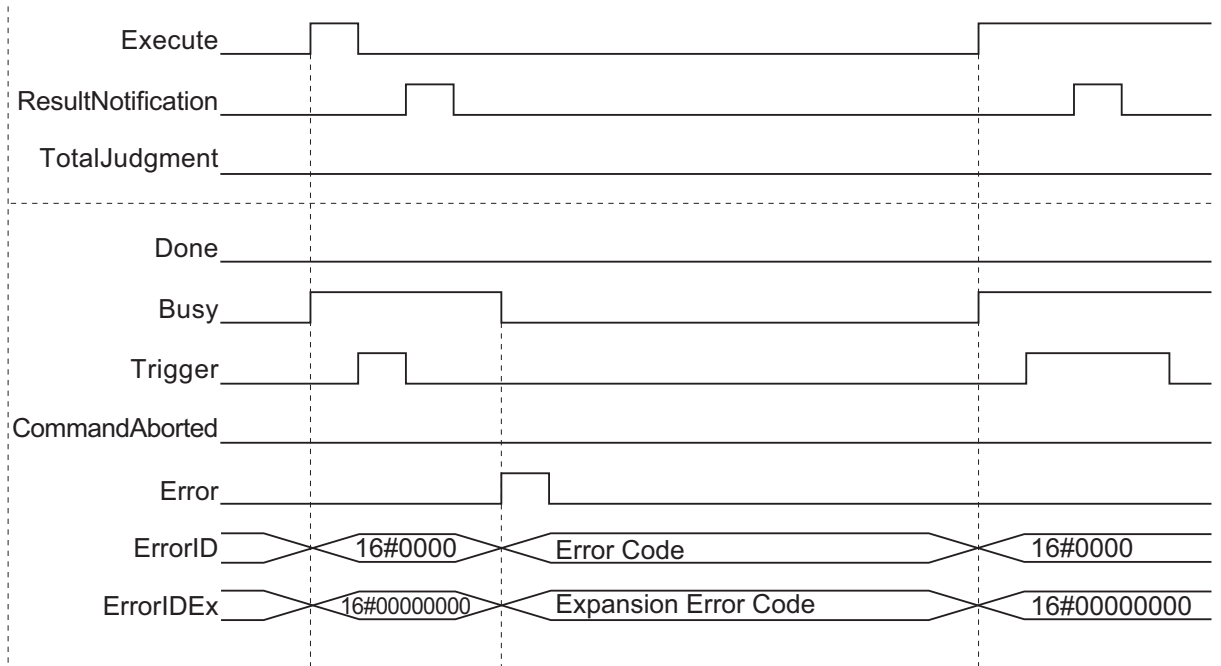
- Timing Chart for Normal End



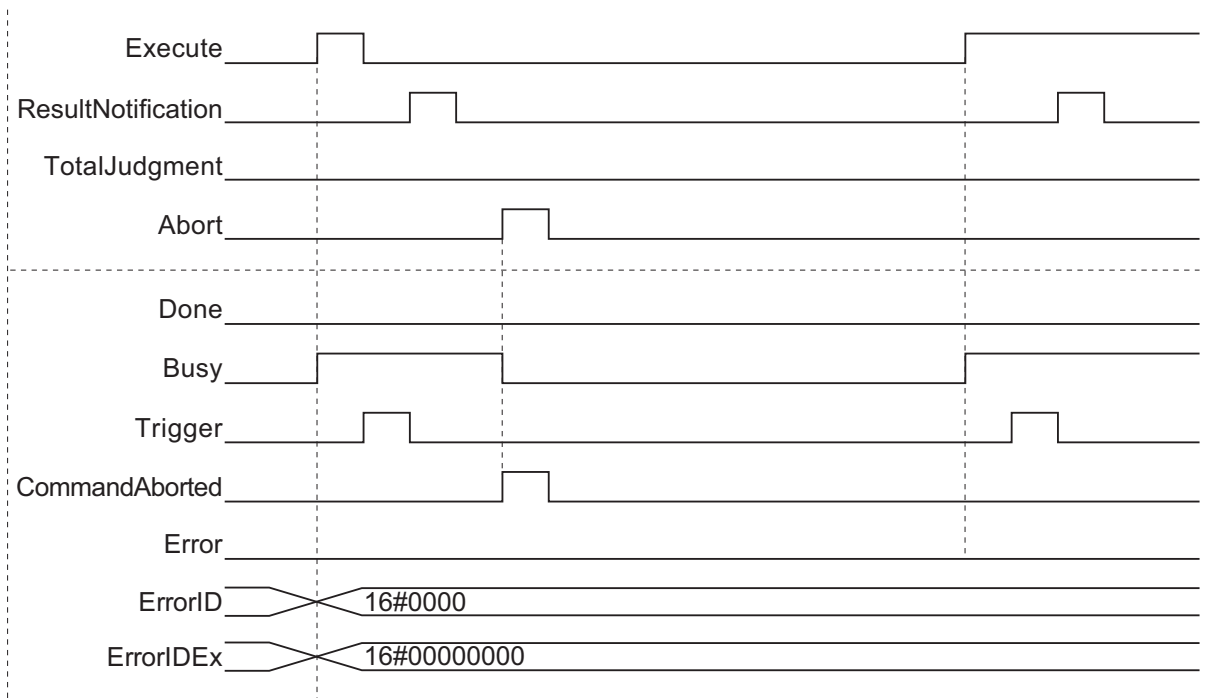
- Timing Chart for Error End (Error at start of execution)



- Timing Chart for Error End (Error while execution is in progress)



- Timing Chart for Command Aborted (Command Aborted while execution is in progress, or when entering Abort input variable)



Precautions for Correct Use

The following will result in an error.

TRUE is output for Error (Error) and the calibration parameter is not updated.

- If an alignment control parameter outside of the valid range is set
- If the value set for the calibration axis motion parameter is outside the valid range
- When all elements of the Target array are FALSE

- When the result of calibration parameter calculation is non-numeric ($\pm\infty$)
- When the FB/FUN terminates with an error end

Within this FB, there is no monitoring of errors common to MC Function Module (MC common errors) and errors occurring on each axis (axis errors). Therefore, monitor during execution of this FB in the program and, in case of any error, abort execution of this FB and stop the axis movement. For details on monitoring and stopping methods, refer to the following manuals.

- *NJ/NX-series CPU Unit Motion Control User's Manual (Cat. No. W507)*
- *NJ/NX-series Motion Control Instructions Reference Manual (Cat. No. W508)*
- *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Motion Control User's Manual (Cat. No. W559)*
- *NY Series Motion Control Instructions Reference Manual (Cat. No. W561)*

Within this FB, there is no verification process on its own prerequisite conditions for operation (such as verification that there is no error caused by EtherCAT communication with the vision sensor and Servo Drive, or that there is no error beyond a minor fault on the controlled axis, that it is in the Servo ON state, or that it is in the home defined state). For this reason, please check each state before executing this FB in any program.

For each state, refer to the following manual.

- *NJ/NX-series CPU Unit Motion Control User's Manual (Cat. No. W507)*
- *NJ/NX-series Motion Control Instructions Reference Manual (Cat. No. W508)*
- *NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Motion Control User's Manual (Cat. No. W559)*
- *NY Series Motion Control Instructions Reference Manual (Cat. No. W561)*

Sample Programming

Please refer to *Sample Programming* on page 4 - 90.

Troubleshooting (Error Codes and Corrective Actions)

Error code	Ex-pansion error code	Status	Description	Corrective action
16#0000	16#000000	Normal end	—	—
16#3D10	16#0000001	Invalid axis movement parameter for calibration	The value set for the calibration axis motion parameter is outside the valid range.	Correct and reset the calibration axis motion parameter.
	16#0000002	Invalid alignment control parameter	If an alignment control parameter outside of the valid range is set.	Correct the value entered for the alignment control parameter.
	16#0000004	Nothing to calculate	All elements of the Target array are FALSE.	Please set so that at least one point can be calculated.

Error code	Expansion error code	Status	Description	Corrective action
	16#00 00000 5	Invalid calculation of calibration parameter	The result of calibration parameter calculation is non-numeric ($\pm\infty$).	Revise such that the magnification, calibration axis motion parameter, stage parameter, and camera coordinate position can be calculated.
	16#20 00000 x	CalcInverseKinematics error	There is an error in CalcInverseKinematics.	Check the value for x against the Out value from CalcInverseKinematics and correct as needed.
	16#30 00000 x	CalcForwardKinematics error	There is an error in CalcForwardKinematics.	Check the value for x against the Out value from CalcForwardKinematics and correct as needed.
	16#70 00xxxx	MC_MoveAbsolute error	There is an error in MC_SyncMoveAbsolute.	Check the xxxx value against the MC_SyncMoveAbsolute error code (ErrorID) and correct as needed. *1
	16#90 00xxxx	MC_Stop Error	There is an error in MC_Stop.	Check the xxxx value against the MC_Stop error code (ErrorID) and correct as needed. *1

- *1. For error codes of MC_MoveVelocity, MC_Stop, refer to the error code list in *NJ/NX-series Motion Control Instructions Reference Manual (Cat. No. W508)* or *NY Series Motion Control Instructions Reference Manual (Cat. No. W561)*

Sample Programming

Sample programming for executing alignment control of XY θ stage using the Visual Feedback Alignment Library is shown.

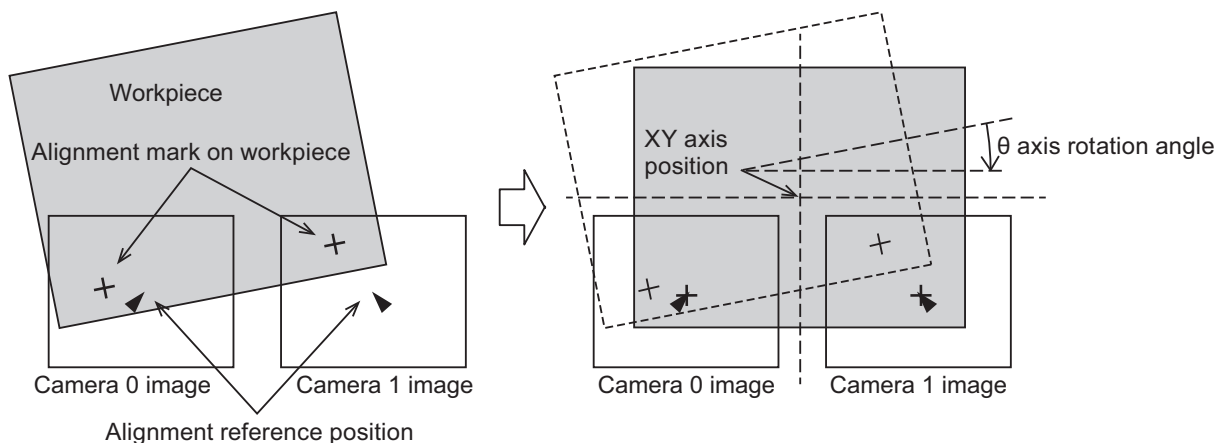
The sample programming is provided as a Sysmac Studio project file. The descriptions in this section are based on this project file. Ask your OMRON representative for details on obtaining the project file.

Overview

Overview of Operation

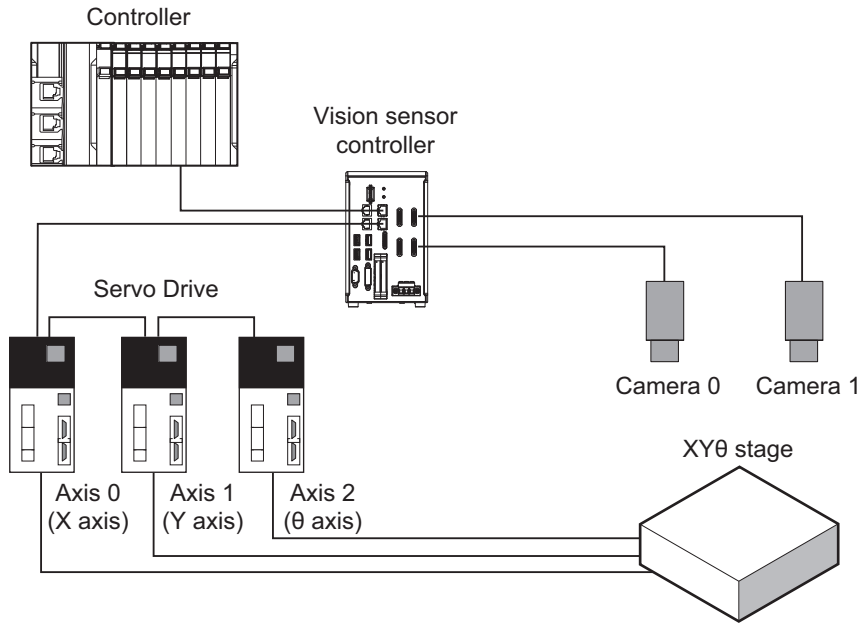
Alignment control of XY positions and θ angle on a workpiece, or XY axis position and θ axis rotation angle of a stage is performed based on two alignment marks made on the workpiece object.

In this sample programming, the position of each alignment mark is measured with two cameras connected to the vision sensor controller, and the stage is controlled so that the alignment mark aligns with a predetermined reference position on each camera control screen.



System Configuration

This sample programming is for the following system configuration.



Configura- tion element	Name	Model	Notes
Controller	Machine Automation Control- ler NJ/NX Series	NJ301-1100	Ver. 1.08 or later
Vision sensor controller	Vision System FH Series	FH-3050-20	Ver. 5.50 or later
Camera	High speed digital CMOS camera	FH-SM02	Monochrome 2 Megapixel *1
Lens	C Mount High resolution tele- centric lens for 2/3-inch imag- ing element	3Z4S-LE VS-TCH4-65-O	4x Telecentric lens *1
Servo Drive	AC Servo Driver (G5 Series)	R88D-KNA5L-ECT	Ver. 1.0 or later
Stage	XYθ Stage	CTLH120M-1010 + AT120 (IKO)	An XYθ configuration com- prised of a combination of CTLH120M (XY Stage) and AT120 (θ Stage)
Software	Sysmac Studio Automation Software	SYSMAC-SE2□□□	Ver. 1.18 or higher

*1. The resolutions based on this combination of cameras and lenses are as follows.

Configuration	Item	Performance
Camera (Standalone)	Imaging element size (H × V)	11.26 × 5.98 mm
	Effective pixels (H × V)	2040 × 1088 pixel
	Pixel resolution (H × V)	5.52 × 5.50 um/pixel
Camera + Lens	Field of view size (H × V)	2.82 × 1.50 mm
	Pixel resolution (H × V)	1.38 × 1.37 um/pixel

Controller, vision sensor controller and Servo Drive are connected by EtherCAT communication. With this sample program, the vision sensor controller, given the imaging instructions from the control-
ler, measures the position of the alignment mark on an image and outputs it to the controller. Based on the measured position and the reference position information (the target position of the alignment), the controller calculates the stage movement for alignment control and gives an operation command to the Servo Drive that drives the stage.

System Settings

The primary system settings for this sample programming are given below. Unless otherwise specified, use the factory setting or the default value of the software.

Configura- tion element	Item	Setting item	Setting value	Notes
Controller	EtherCAT (Network con- figuration)	Node address 1	Device name: E001 Model: R88D-KNA5L-ECT	Assign to axis 0 (MC_Ax- is000).
		Node address 2	Device name: E002 Model: R88D-KNA5L-ECT	Assign to axis 1 (MC_Ax- is001).
		Node address 3	Device name: E003 Model: R88D-KNA5L-ECT	Assign to axis 2 (MC_Ax- is002).
		Node address 4	Device name: E004 Model: FH3050-20	Generate device variables us- ing the I/O map.
	Motion Con- trol Setup	Axis Settings	Add axes 0 to 2.	Set each axis according to the specifications of the Servomo- tor and stage used.
	Task settings	Primary peri- odic task	1 ms	The sample programming is executed in the primary peri- odic task.
Vision sensor controller	Startup set- ting	Communica- tion module	Fieldbus : EtherCAT	Refer to <i>Vision System FH/FZ5 Series User's Manual for Communications Settings (Cat. No. Z365)</i> for details.
	EtherCAT Communica- tions	Number of data to output	Result Data Format 0 (DINT8)	For this sample programming, only Line 0 can be used.
	Editing the flow	Fieldbus Data Output	Output data No.0: Camera 0 measurement coordinate X No.1: Camera 0 measurement coordinate Y No.2: Camera 1 measurement coordinate X No.3: Camera 1 measurement coordinate Y No.4: Camera 0 reference co- ordinate X No.5: Camera 0 reference co- ordinate Y No.6: Camera 1 reference co- ordinate X No.7: Camera 1 reference co- ordinate Y	Assigns a measurement unit output to each camera image input. Refer to <i>Vision System FH/FZ5 Series Processing Item Function Reference Manual (Cat. No. Z341)</i> . for details.

Sample Programming Types

This manual includes the following sample programs.

No.	Program name	Description	FB/FUN used	Reference
1	Auto-calibration	Use Auto-generate calibration parameter FB (GenerateCalibParams) to generate calibration parameters to be used for coordinate conversion FUN (AffineTrans).	Auto-generate calibration parameter FB (GenerateCalibParams)	<i>Auto-calibration</i> on page 4 - 93
2	Alignment Control	Alignment control is performed using Stage Control FB (CtrlStage).	Coordinate conversion FUN (AffineTrans) Position and Angle Calculation FUN (CalcPosAngle) Stage Control FB (CtrlStage)	<i>Alignment Control</i> on page 4 - 106

In this sample programming, only the basic operation parts are described. In order to actually run this program, you need to add the following programming.

- The Servo lock circuit of each axis to be used (MC_Power instruction)
- The Homing circuit for each axis to be used (MC_Home instruction, etc.)
- Movement circuit to reference position during auto-calibration (MC_MoveAbsolute instruction, etc.)
- Movement circuit to position at alignment start (MC_MoveAbsolute instruction, etc.)

The controller reads and writes process data on the vision sensor controller, Servo Drive and EtherCAT communication via device variables and axis variables.

When EtherCAT communication is not established, such as immediately after turning on the power of the controller, the values of the process data are invalid, so the values of the device variables and axis variables input / output process data are invalid. Therefore, it is necessary to read and write device variables and axis variables that input and output process data after confirming that the process data becomes valid.

For details on how to verify the validity of process data programmatically, refer to the sample programming for checking the validity of all slave process data in *NJ/NX-series CPU Unit Built-in EtherCAT® Port User's Manual (Cat. No. W505)*.

After checking the validity of the process data, turn on the Servo Drive using the MC_Power command. In addition, if the process data becomes invalid, or if other causes of errors are detected, execute the MC_Stop command to stop the axis movement, or otherwise handle error processing according to machine specifications.

Auto-calibration

In this program, the Auto-generate calibration parameter FB (GenerateCalibParams) moves the stage with the set movement amount and instructs measurement to the vision sensor controller.

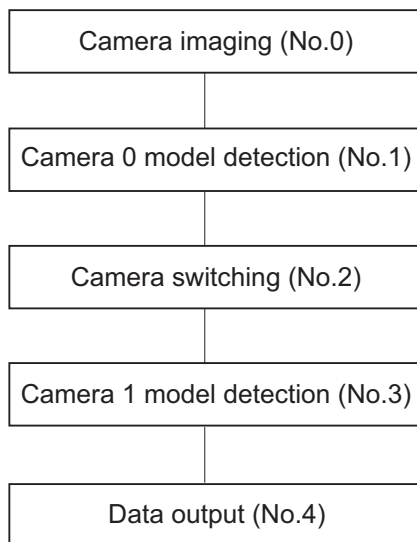
Samples the current position of the stage and the measurement position of the alignment mark measured by the vision sensor controller, and generates a calibration parameter to be used for the coordinate conversion FUN (AffineTrans).

In the following sample programming for alignment control, the calibration parameters generated by this programming are used.

Vision Sensor Controller Processing Flow

The vision sensor controller images the alignment mark with two cameras in response to the imaging instruction given from the controller. On the captured image, it detects the shape of the pre-registered alignment mark and measures its detection point coordinates.

The measurement result (measurement position) is output to the controller via EtherCAT communication.

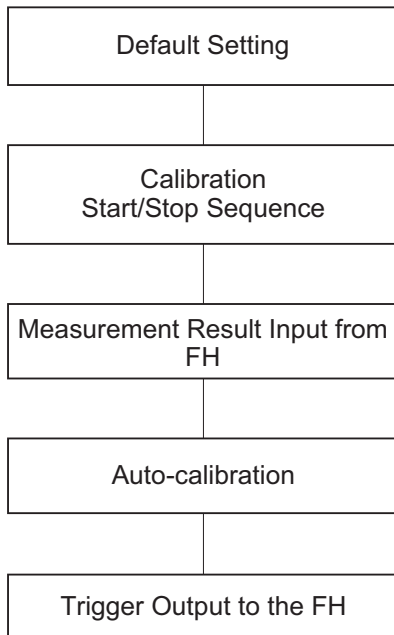


No.	Item	Unit	Function	Settings
0	Camera imaging	Camera Image Input FH	Image capture of a mark using camera 0 and camera 1. The object to be measured at this time is from camera 0.	Set the Shutter speed and Gain for camera 0 and camera 1.
1	Camera 0 model detection	Shape Search III	Measure the position of the feature (mark) registered from the image of camera 0.	The feature (mark) to be detected by the camera 0 is registered as a model.
2	Camera switching	Camera switching	Switch the measurement target to camera 1.	Set camera 1 in the selection setting.
3	Camera 1 model detection	Shape Search III	Measure the position of the feature (mark) registered from the image of camera 1.	The feature (mark) to be detected by the camera 1 is registered as a model.
4	Data Output	Fieldbus Data Output	In the EtherCAT communication, the measurement position of Mark 0 and Mark 1 are output.	Set the data to be output (Measurement position, Overall Judgment).

Overview of Sample Programming

The sample programming for auto-calibration is roughly divided into the following processes.

Section Structure



N o.	Process	Description	Ladder diagram	Structured text (ST)
1	Default Setting	Set various parameters of FB / FUN used in this sample programming as variables.	Program name: CtrlStage_LD_Samp Section name: Settings Rung number: 0 to 5	Program name: CtrlStage_ST_Samp Row number: 2 to 81
2	Calibration Start/ Stop Sequence	The circuit for the Auto-Calibration Start/Stop sequence. It accepts Start/ Stop instructions from a high level sequence program, or display device.	Program name: CtrlStage_LD_Samp Section name: AutoCalibration Rung number: 0 to 7	Program name: CtrlStage_ST_Samp Row number: 281 to 324
3	Measurement Result Input from FH	Converts the measurement result (measurement position) from the vision sensor controller so that it can be input to FB / FUN and assigns it to the variable. Measurement results from the vision sensor controller are input via device variables. This sample programming shows the case where the measurement result is output with DINT. In the FH Series vision sensor controller, in the case of DINT output, since the measurement result has a fixed-point output format of three digits after the decimal point, use the value divided by 1000 as shown in this circuit.	Program name: CtrlStage_LD_Samp Section name: AutoCalibration Rung number: 8	Program name: CtrlStage_ST_Samp Row number: 326 to 332

N o.	Process	Description	Ladder diagram	Structured text (ST)
4	Auto-calibration	Based on the default setting, it gives imaging instruction to the vision sensor controller while moving the stage position. Converts the measurement position obtained from the vision sensor controller to the stage coordinate system, calculates the calibration parameter by correspondence to the stage's current position. The following FB/FUN are used. Auto-generate calibration parameter FB (GenerateCalibParams)	Program name: CtrlStage_LD_Samp Section name: AutoCalibration Rung number: 9	Program name: CtrlStage_ST_Samp Row number: 334 to 353
5	Trigger Output to the FH	Sends the measurement sensor trigger output from the Auto-generate calibration parameter FB (GenerateCalibParams) to the vision sensor controller. The measurement trigger output to the vision sensor controller is output using a device variable.	Program name: CtrlStage_LD_Samp Section name: FH_Control Rung number: 0	Program name: CtrlStage_ST_Samp Row number: 355 to 356

Primary Variables

Name	Data type	Default	Comment
CalibParams	ARRAY[0..3] OF OmronLib\VF_Alignment\sCALIB_PARAMS	—	Variables for storing the calibration parameters.
AlignmentParams	OmronLib\VF_Alignment\sALIGNMENT_PARAMS	—	Variables for storing the alignment parameters.
CalibMoveParams	OmronLib\VF_Alignment\sCALIB_MOVE_PARAMS	—	Variables for storing the movement parameters used for calibration.
CalibStart	BOOL	FALSE	Alignment control starts when this variable changes from FALSE to TRUE.
CalibStop	BOOL	FALSE	Alignment control stops when this variable changes from FALSE to TRUE.
Reset	BOOL	FALSE	Fault Reset is executed when this variable changes from FALSE to TRUE.
ServoON_X	BOOL	FALSE	This variable is TRUE when the X axis Servo is ON.
ServoON_Y	BOOL	FALSE	This variable is TRUE when the Y axis Servo is ON.
ServoON_TH	BOOL	FALSE	This variable is TRUE when the θ axis Servo is ON.

Name	Data type	Default	Comment
AutoCalib-HomePos	BOOL	FALSE	An internal relay indicating when the Calibration Control Sequence can be started.
AutoCalib-Start	BOOL	FALSE	An internal variable that starts the Calibration Control Sequence.
AutoCalib-Stop	BOOL	FALSE	An internal variable that stops the Calibration Control Sequence.
AutoCalibEnd	BOOL	FALSE	An internal variable that ends the Calibration Control Sequence.
AL_AutoCalibT-OVER	BOOL	FALSE	This is an alarm flag indicating that a Calibration Control timeout occurred.
AL_GenerateCalibParams	BOOL	FALSE	This is an alarm flag indicating that an error occurred in GenerateCalibParams.
AL_MC_EC_Error	BOOL	FALSE	This is an alarm flag that indicates that an error occurred in the Motion Control Function Module of the CPU Unit or in the EtherCAT Master Function Module.
CalibAlarm	BOOL	FALSE	This becomes TRUE when there is an Auto-calibration alarm.
GenerateCalibParams_TOver	TON	—	This becomes TRUE when the time for Auto-calibration exceeds the set time.
GenerateCalibParams_Done	BOOL	FALSE	This is a variable for storing Done output from GenerateCalibParams.
GenerateCalibParams_Busy	BOOL	FALSE	This is a variable for storing Busy output from GenerateCalibParams.
GenerateCalibParams_Error	BOOL	FALSE	This is a variable for storing Error output from GenerateCalibParams.
GenerateCalibParams_Trigger	BOOL	FALSE	This is a variable that outputs the imaging trigger to send to the vision sensor.
MeasMarkPosCamera	ARRAY[0..3] OF OmronLib\VF_Alignment\POSITION	—	Variables for storing the measurement position for cameras 0 and 1. Index 0 of the array is camera 0, and Index 1 is the measurement position of camera 1.
CalcPos-Target	ARRAY[0..3] OF BOOL	FALSE	Set the points for which to generate the calibration parameters. In this sample programming, the array elements [0] and [1] are set to TRUE because they are for point 0 and point 1.
GenerateCalibParams_instance	\\Omronlib\VF_Alignment\GenerateCalibParams	—	This is an instance of calibration parameter automatic generation FB (GenerateCalibParams).

Ladder Diagram

Default setting

```

0 Stage parameter settings
P_First_Run
1 AlignmentParams.StageParams.StageType := 1; // Stage type : XYθ stage
2 AlignmentParams.StageParams.AxNo[0] := 0; // X axis settings
3 AlignmentParams.StageParams.AxNo[1] := 1; // Y axis settings
4 AlignmentParams.StageParams.AxNo[2] := 2; // θ axis settings
5 AlignmentParams.StageParams.StopDec[0] := LREAL#100.0; // X stop of deceleration
6 AlignmentParams.StageParams.StopDec[1] := LREAL#100.0; // Y stop of deceleration
7 AlignmentParams.StageParams.StopDec[2] := LREAL#55.664195; // θ stop of deceleration
8 AlignmentParams.StageParams.YXTHStageParams.Rotation := UINT#0; // θ direction : positive
9 AlignmentParams.StageParams.YXTHStageParams.TH_Type := UINT#1; // θ type : linear
10 AlignmentParams.StageParams.YXTHStageParams.TH_Length := LREAL#100.0; // θ length
11 AlignmentParams.StageParams.YXTHStageParams.TH_Offset := LREAL#0.0; // θ offset of angle

```

```

1 Axis movement parameter settings
P_First_Run
1 AlignmentParams.CalcVirtualMoveParams.MoveParams_X.Kp := LREAL#16.0; // X proportio
2 AlignmentParams.CalcVirtualMoveParams.MoveParams_X.UseMaxVel := FALSE; // X velocity l
3 AlignmentParams.CalcVirtualMoveParams.MoveParams_X.MaxVel := LREAL#5.0; // X velocity l
4 AlignmentParams.CalcVirtualMoveParams.MoveParams_X.UseAccDec := FALSE; // X accelerat
5 AlignmentParams.CalcVirtualMoveParams.MoveParams_X.AccDec := LREAL#100.0; // X accelerat
6 AlignmentParams.CalcVirtualMoveParams.MoveParams_X.ImagingLimitVel := LREAL#1.0; // X velocity l
7 AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.Kp := LREAL#16.0; // Y proportio
8 AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.UseMaxVel := FALSE; // Y velocity l
9 AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.MaxVel := LREAL#5.0; // Y velocity l
10 AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.UseAccDec := FALSE; // Y accelerat
11 AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.AccDec := LREAL#100.0; // Y accelerat
12 AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.ImagingLimitVel := LREAL#1.0; // Y velocity l
13 AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.Kp := LREAL#16.0; // θ proportio
14 AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.UseMaxVel := FALSE; // θ velocity l
15 AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.MaxVel := LREAL#2.78320975; // θ velocity l
16 AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.UseAccDec := FALSE; // θ accelerat
17 AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.AccDec := LREAL#55.664195; // θ accelerat
18 AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.ImagingLimitVel := LREAL#0.55664195; // θ velocity l
19 AlignmentParams.CalcVirtualMoveParams.InPosMode := UINT#0; // judgement
20 AlignmentParams.CalcVirtualMoveParams.InPosRange[0] := LREAL#0.001; // threshold c
21 AlignmentParams.CalcVirtualMoveParams.InPosRange[1] := LREAL#0.5; // threshold c
22 AlignmentParams.CalcVirtualMoveParams.InPosRange[2] := LREAL#0.0; // threshold c
23 AlignmentParams.CalcVirtualMoveParams.InPosTarget[0] := TRUE; // target of ra
24 AlignmentParams.CalcVirtualMoveParams.InPosTarget[1] := TRUE; // target of ar
25 AlignmentParams.CalcVirtualMoveParams.InPosTarget[2] := FALSE; // target of ar

```

```

2 Curve parameter settings
P_First_Run
1 AlignmentParams.CalcCurveParams.UseCurve := TRUE; // use curve
2 AlignmentParams.CalcCurveParams.UseCurveSequence := TRUE; // use curve sequence
3 AlignmentParams.CalcCurveParams.UseConnectingVel := TRUE; // use speed connect
4 AlignmentParams.CalcCurveParams.MinCurveTime := LREAL#32.0; // min. curve time : 32ms
5 AlignmentParams.CalcCurveParams.MaxCurveTime := LREAL#1000.0; // max. curve time : 1000ms
6 AlignmentParams.CalcCurveParams.DistNumber := 1; // dist. number : 1

```

```

3 Imaging parameter settings
P_First_Run
1 AlignmentParams.ImagingParams.ShutterSpeed := LREAL#10.0; // shutter speed
2 CalcPosTarget[0] := TRUE; // target of camera 0 settings
3 CalcPosTarget[1] := TRUE; // target of camera 1 settings
4 CalcPosTarget[2] := FALSE; // target of camera 2 settings
5 CalcPosTarget[3] := FALSE; // target of camera 3 settings

```

```

4 Judgement parameter settings
P_First_Run
1 AlignmentParams.JudgeParams.InPosMode := UINT#0; // judgement mode : radius, angle
2 AlignmentParams.JudgeParams.InPosRange[0] := LREAL#0.001; // threshold of radius
3 AlignmentParams.JudgeParams.InPosRange[1] := LREAL#0.5; // threshold of angle
4 AlignmentParams.JudgeParams.InPosRange[2] := LREAL#0.000; // threshold of angle
5 AlignmentParams.JudgeParams.InPosTarget[0] := TRUE; // target of radius
6 AlignmentParams.JudgeParams.InPosTarget[1] := TRUE; // target of angle
7 AlignmentParams.JudgeParams.InPosTarget[2] := FALSE; // target of radius
8 AlignmentParams.JudgeParams.TargetMark[0] := CalcPosTarget[0]; // target of camera 0 : TRUE
9 AlignmentParams.JudgeParams.TargetMark[1] := CalcPosTarget[1]; // target of camera 1 : TRUE
10 AlignmentParams.JudgeParams.TargetMark[2] := CalcPosTarget[2]; // target of camera 2 : FALSE
11 AlignmentParams.JudgeParams.TargetMark[3] := CalcPosTarget[3]; // target of camera 3 : FALSE

```

```

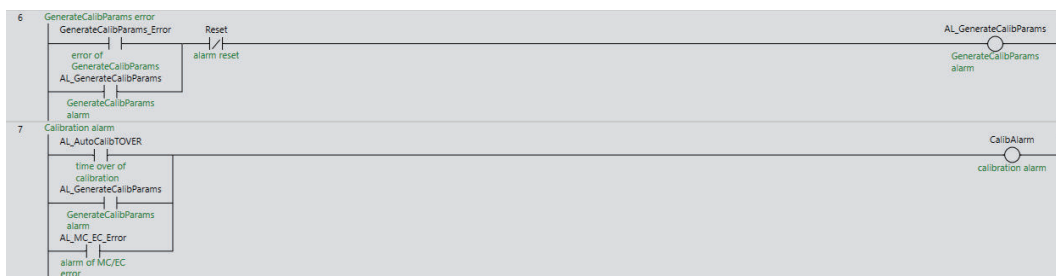
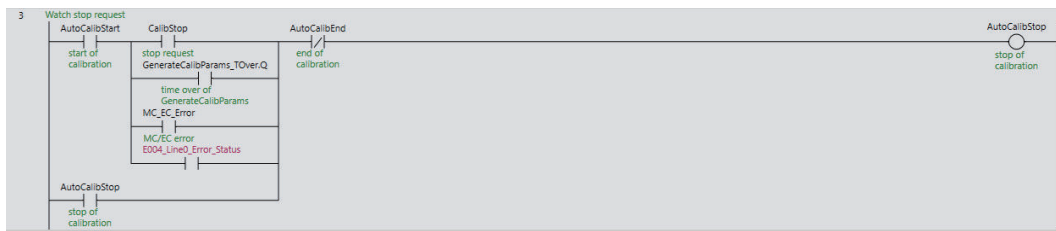
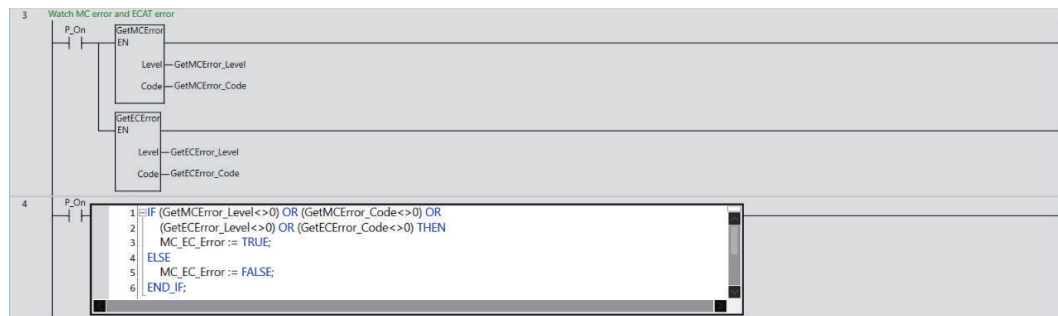
5 Calibration parameter settings
P_First_Run
1 CalibMoveParams.Movement_X := LREAL#1.0; // X amount
2 CalibMoveParams.Velocity_XU := LREAL#20.0; // X velocity
3 CalibMoveParams.AccDec_XU := LREAL#200.0; // X acceleration
4 CalibMoveParams.Movement_Y := LREAL#0.5; // Y amount
5 CalibMoveParams.Velocity_YV := LREAL#10.0; // Y velocity
6 CalibMoveParams.AccDec_YV := LREAL#100.0; // Y acceleration
7 CalibMoveParams.Movement_TH := LREAL#0.625; // θ amount
8 CalibMoveParams.Velocity_THW := LREAL#5.0; // θ velocity
9 CalibMoveParams.AccDec_THW := LREAL#50.0; // θ acceleration
10 CalibMoveParams.StopDec[0] := LREAL#200.0; // X stop of deceleration
11 CalibMoveParams.StopDec[1] := LREAL#100.0; // Y stop of deceleration
12 CalibMoveParams.StopDec[2] := LREAL#50.0; // θ stop of deceleration
13 CalibMoveParams.WaitTime := UINT#0; // Imaging wait time

```

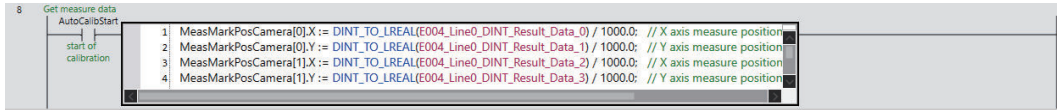
Calibration Control Start/Stop sequence



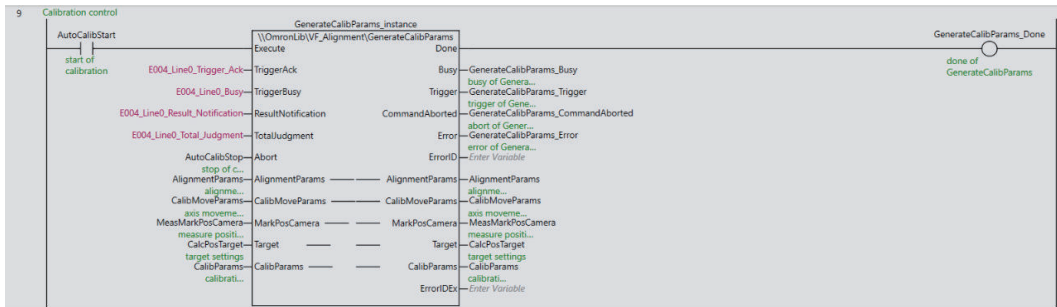
This circuit in the alignment control sample programming is a common circuit for both the Motion Control Function Module of the CPU Unit and the Fault Monitoring circuit of the EtherCAT Master Function Module.



Measurement Result Input from FH

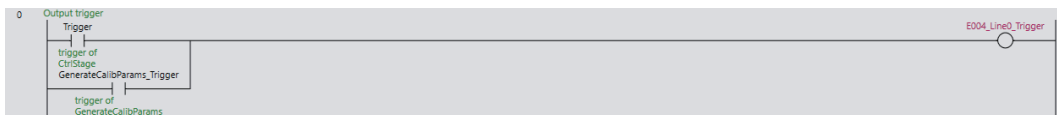


Auto-calibration



Trigger Output to the FH

It is the same circuit as the alignment control sample programming.



Axis Control Error Reset Processing

It is the same circuit as the alignment control sample programming.



Structured Text (ST)

Default setting

```
// Initial settings
IF P_First_Run THEN
    // Stage parameter settings
    AlignmentParams.StageParams.StageType := 1; // Stage type : XYθ stage
    AlignmentParams.StageParams.AxNo[0] := 0; // X axis settings
```

```

AlignmentParams.StageParams.AxNo[1] := 1;      // Y axis settings
AlignmentParams.StageParams.AxNo[2] := 2;      // θ axis settings
AlignmentParams.StageParams.StopDec[0] := LREAL#100.0;
                                                // X stop of deceleration
AlignmentParams.StageParams.StopDec[1] := LREAL#100.0;
                                                // Y stop of deceleration
AlignmentParams.StageParams.StopDec[2] := LREAL#55.664195;
                                                // θ stop of deceleration
AlignmentParams.StageParams.XYTHStageParams.Rotation := UINT#0;
                                                // θ rotation : positive
AlignmentParams.StageParams.XYTHStageParams.TH_Type := UINT#1;
                                                // θ type : linear
AlignmentParams.StageParams.XYTHStageParams.TH_Length := LREAL#100.0;
                                                // θ length
AlignmentParams.StageParams.XYTHStageParams.TH_Offset := LREAL#0.0;
                                                // θ offset of angle

// Axis movement parameter settings
AlignmentParams.CalcVirtualMoveParams.MoveParams_X.Kp := LREAL#16.0;
                                                // X proportional gain
AlignmentParams.CalcVirtualMoveParams.MoveParams_X.UseMaxVel := FALSE;
                                                // X velocity limit select
AlignmentParams.CalcVirtualMoveParams.MoveParams_X.MaxVel := LREAL#5.0;
                                                // X velocity limit
AlignmentParams.CalcVirtualMoveParams.MoveParams_X.UseAccDec := FALSE;
                                                // X acceleration limit select
AlignmentParams.CalcVirtualMoveParams.MoveParams_X.AccDec := LREAL#100.0;
                                                // X acceleration limit
AlignmentParams.CalcVirtualMoveParams.MoveParams_X.ImagingLimitVel
:= LREAL#1.0;
                                                // X velocity limit of imaging
AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.Kp := LREAL#16.0;
                                                // Y proportional gain
AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.UseMaxVel := FALSE;
                                                // Y velocity limit select
AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.MaxVel := LREAL#5.0;
                                                // Y velocity limit
AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.UseAccDec := FALSE;
                                                // Y acceleration limit select
AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.AccDec := LREAL#100.0;
                                                // Y acceleration limit
AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.ImagingLimitVel
:= LREAL#1.0;
                                                // Y velocity limit of imaging
AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.Kp := LREAL#16.0;
                                                // θ proportional gain
AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.UseMaxVel := FALSE;
                                                // θ velocity limit select
AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.MaxVel := LREAL#2.78320975;

```

```

// θ velocity limit
AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.UseAccDec := FALSE;
// θ acceleration limit select
AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.AccDec := LREAL#55.664195;
// θ acceleration limit
AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.ImagingLimitVel
:= LREAL#0.55664195; // θ velocity limit of imaging
AlignmentParams.CalcVirtualMoveParams.InPosMode := UINT#0;
// judgment mode : radius, angle
AlignmentParams.CalcVirtualMoveParams.InPosRange[0] := LREAL#0.001;
// threshold of radius
AlignmentParams.CalcVirtualMoveParams.InPosRange[1] := LREAL#0.5;
AlignmentParams.CalcVirtualMoveParams.InPosRange[2] := LREAL#0.0;
// threshold of angle
AlignmentParams.CalcVirtualMoveParams.InPosTarget[0] := TRUE;
AlignmentParams.CalcVirtualMoveParams.InPosTarget[1] := TRUE;
// target of radius
AlignmentParams.CalcVirtualMoveParams.InPosTarget[2] := FALSE;
// target of angle

// Curve parameter settings
AlignmentParams.CalcCurveParams.UseCurve := TRUE;
// use curve
AlignmentParams.CalcCurveParams.UseCurveSequence := TRUE;
// use curve sequence
AlignmentParams.CalcCurveParams.UseConnectingVel := TRUE;
// use speed connect
AlignmentParams.CalcCurveParams.MinCurveTime := LREAL#32.0;
// min. curve time : 32ms
AlignmentParams.CalcCurveParams.MaxCurveTime := LREAL#1000.0;
// max. curve time : 1000ms
AlignmentParams.CalcCurveParams.DistNumber := 1;
// dist. number : 1

// Image parameter settings
AlignmentParams.ImagingParams.ShutterSpeed := LREAL#10.0; // shutter speed
CalcPosTarget[0] := TRUE; // target of camera 0 settings
CalcPosTarget[1] := TRUE; // target of camera 1 settings
CalcPosTarget[2] := FALSE; // target of camera 2 settings
CalcPosTarget[3] := FALSE; // target of camera 3 settings

// Judgment parameter settings
AlignmentParams.JudgeParams.InPosMode := UINT#0;
// judgment mode : radius, angle
AlignmentParams.JudgeParams.InPosRange[0] := LREAL#0.001;
// threshold of radius
AlignmentParams.JudgeParams.InPosRange[1] := LREAL#0.5;

```



```

AlignmentParams.JudgeParams.InPosRange[2] := LREAL#0.000;
// threshold of angle

AlignmentParams.JudgeParams.InPosTarget[0] := TRUE;
// target of radius

AlignmentParams.JudgeParams.InPosTarget[1] := TRUE;
AlignmentParams.JudgeParams.InPosTarget[2] := FALSE;
// target of angle

AlignmentParams.JudgeParams.TargetMark[0] := CalcPosTarget[0];
// target of camera 0 : TRUE

AlignmentParams.JudgeParams.TargetMark[1] := CalcPosTarget[1];
// target of camera 1 : TRUE

AlignmentParams.JudgeParams.TargetMark[2] := CalcPosTarget[2];
// target of camera 2 : FALSE

AlignmentParams.JudgeParams.TargetMark[3] := CalcPosTarget[3];
// target of camera 3 : FALSE

// Axis movement parameter settings
CalibMoveParams.Movement_X := LREAL#1.0; // X amount
CalibMoveParams.Velocity_XU := LREAL#20.0; // X velocity
CalibMoveParams.AccDec_XU := LREAL#200.0; // X acceleration
CalibMoveParams.Movement_Y := LREAL#0.5; // Y amount
CalibMoveParams.Velocity_YV := LREAL#10.0; // Y velocity
CalibMoveParams.AccDec_YV := LREAL#100.0; // Y acceleration
CalibMoveParams.Movement_TH := LREAL#0.625; //  $\theta$  amount
CalibMoveParams.Velocity_THW := LREAL#5.0; //  $\theta$  velocity
CalibMoveParams.AccDec_THW := LREAL#50.0; //  $\theta$  acceleration
CalibMoveParams.StopDec[0] := LREAL#200.0; // X stop of deceleration
CalibMoveParams.StopDec[1] := LREAL#100.0; // Y stop of deceleration
CalibMoveParams.StopDec[2] := LREAL#50.0; //  $\theta$  stop of deceleration
CalibMoveParams.WaitTime := UINT#0; // Imaging wait time
END_IF;

```

Calibration Control Start/Stop sequence

```

// Watch start condition
IF ServoON_X AND ServoON_Y AND ServoON_TH AND NOT(GenerateCalibParams_Busy)
AND NOT(CalibAlarm) THEN
    AutoCalibHomePos := TRUE;
ELSE
    AutoCalibHomePos := FALSE;
END_IF;

// Start of calibration
R_TRIG_CalibStart(Clk:=CalibStart);
IF R_TRIG_CalibStart.Q AND AutoCalibHomePos AND NOT(AutoCalibEnd) THEN
    AutoCalibStart := TRUE;
END_IF;

// Watch calibration time

```

```

GenerateCalibParams_TOver(In:=AutoCalibStart, PT:=T#20s);
// Watch MC error and EC error
GetMCError(
    GetMCError_Level,
    GetMCError_Code
);
GetECError(
    GetECError_Level,
    GetECError_Code
);
IF (GetMCError_Level<>0) OR (GetMCError_Code<>0) OR (GetECError_Level<>0)
    OR (GetECError_Code<>0) THEN
    MC_EC_Error := TRUE;
ELSE
    MC_EC_Error := FALSE;
END_IF;
// Watch stop request
IF AutoCalibStart AND (Stop OR GenerateCalibParams_TOver.Q OR MC_EC_Error
    OR E004_Line0_Error_Status) THEN
    AutoCalibStop := TRUE;
END_IF;
// End of calibration
IF AutoCalibEnd THEN
    AutoCalibStart := FALSE;
    AutoCalibStop := FALSE;
    AutoCalibEnd := FALSE;
END_IF;
IF AutoCalibStart AND (GenerateCalibParams_Done
    OR GenerateCalibParams_CommandAborted OR GenerateCalibParams_Error) THEN
    AutoCalibEnd := TRUE;
END_IF;
// Watch error
IF GenerateCalibParams_TOver.Q THEN // Time over of calibration
    AL_AutoCalibTOVER := TRUE;
END_IF;
IF GenerateCalibParams_Error THEN // GenerateCalibParams alarm
    AL_GenerateCalibParams := TRUE;
END_IF;
IF MC_EC_Error THEN
    AL_MC_EC_Error := TRUE;
ELSE
    AL_MC_EC_Error := FALSE;
END_IF;
IF AL_AutoCalibTOVER OR AL_GenerateCalibParams OR AL_MC_EC_Error THEN
    CalibAlarm := TRUE;
ELSE
    CalibAlarm := FALSE;

```

```

END_IF;
// Alarm reset
IF Reset AND NOT(AutoCalibStart) THEN
    AL_AutoCalibTOVER := FALSE;
    AL_GenerateCalibParams := FALSE;
END_IF;

```

Measurement Result Input from FH

```

// Get measure data
IF AutoCalibStart THEN
    MeasMarkPosCamera[0].X := DINT_TO_LREAL(E004_Line0_DINT_Result_Data_0)
    / LREAL#1000.0; // X axis measure position of camera 0
    MeasMarkPosCamera[0].Y := DINT_TO_LREAL(E004_Line0_DINT_Result_Data_1)
    / LREAL#1000.0; // Y axis measure position of camera 0
    MeasMarkPosCamera[1].X := DINT_TO_LREAL(E004_Line0_DINT_Result_Data_2)
    / LREAL#1000.0; // X axis measure position of camera 1
    MeasMarkPosCamera[1].Y := DINT_TO_LREAL(E004_Line0_DINT_Result_Data_3)
    / LREAL#1000.0; // Y axis measure position of camera 1
END_IF;

```

Auto-calibration

```

// Calibration control
R_TRIG_AutoCalibStart(Clk:=AutoCalibStart);
GenerateCalibParams_instance(
    Execute := R_TRIG_AutoCalibStart.Q,
    TriggerAck := E004_Line0_Trigger_Ack,
    TriggerBusy := E004_Line0_Busy,
    ResultNotification := E004_Line0_Result_Notification,
    TotalJudgment := E004_Line0_Total_Judgment,
    Abort := AutoCalibStop,
    AlignmentParams := AlignmentParams,
    CalibMoveParams := CalibMoveParams,
    MarkPosCamera := MeasMarkPosCamera,
    Target := CalcPosTarget,
    CalibParams := CalibParams,
    Done => GenerateCalibParams_Done,
    Busy => GenerateCalibParams_Busy,
    Trigger => CalibTrigger,
    CommandAborted => GenerateCalibParams_CommandAborted,
    Error => GenerateCalibParams_Error
);

```

Trigger Output to the FH

It is the same circuit as the alignment control sample programming.

```

// Output trigger
E004_Line0_Trigger := (Trigger OR CalibTrigger);

```

Axis Control Error Reset Processing

It is the same circuit as the alignment control sample programming.

```
// Reset
ResetAxisX_instance(
    Execute := (AL_MC_EC_Error AND R_TRIG_Reset.Q AND
                _MC_AX[AlignmentParams.StageParams.AxNo[0]].Status.ErrorStop),
    Axis := _MC_AX[AlignmentParams.StageParams.AxNo[0]]
);
ResetAxisY_instance(
    Execute := (AL_MC_EC_Error AND R_TRIG_Reset.Q AND
                _MC_AX[AlignmentParams.StageParams.AxNo[1]].Status.ErrorStop),
    Axis := _MC_AX[AlignmentParams.StageParams.AxNo[1]]
);
ResetAxisTH_instance(
    Execute := (AL_MC_EC_Error AND R_TRIG_Reset.Q AND
                _MC_AX[AlignmentParams.StageParams.AxNo[2]].Status.ErrorStop),
    Axis := _MC_AX[AlignmentParams.StageParams.AxNo[2]]
);
```

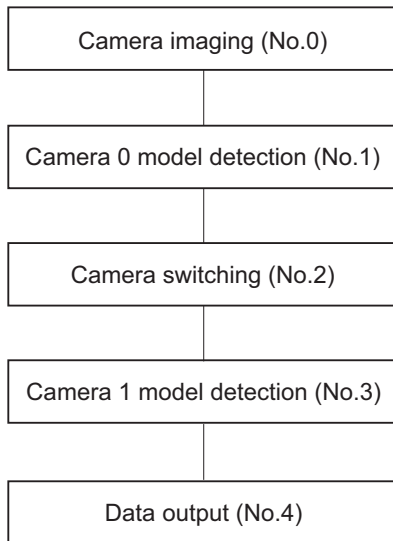
Alignment Control

This program performs the following two processes. First, an imaging instruction is given to the vision sensor controller, and the measurement position obtained from the captured image and the alignment target position (reference position) set in advance by the vision sensor controller are converted to the stage coordinates by the coordinate conversion FUN (AffineTrans). Then, alignment control is performed using the Position and Angle Calculation FUN (CalcPosAngle) and the Stage Control FB (CtrlStage) based on the measurement position and reference position in the stage coordinates.

Vision Sensor Controller Processing Flow

The vision sensor controller images the alignment mark with two cameras in response to the imaging instruction given from the controller. On the captured image, it detects the shape of the pre-registered alignment mark and measures its detection point coordinates.

The measurement result (measurement position) and the reference position (the target position for the alignment) are output to the controller via EtherCAT communication.

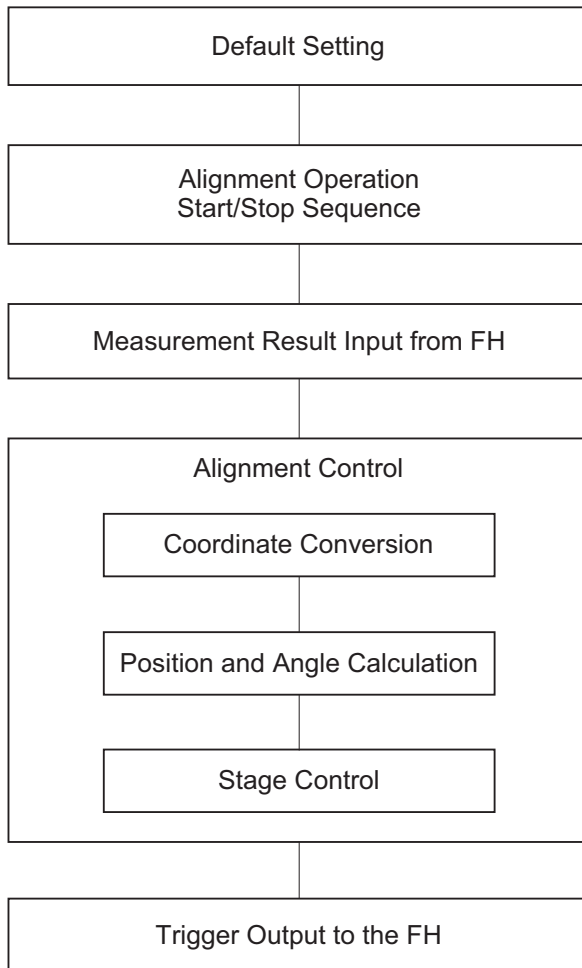


No.	Item	Unit	Function	Settings
0	Camera imaging	Camera Image Input FH	Image capture of a mark using camera 0 and camera 1. The object to be measured at this time is from camera 0.	Set the Shutter speed and Gain for camera 0 and camera 1.
1	Camera 0 model detection	Shape Search III	Measure the position of the feature (mark) registered from the image of camera 0.	The feature (mark) to be detected by the camera 0 is registered as a model.
2	Camera switching	Camera switching	Switch the measurement target to camera 1.	Set camera 1 in the selection setting.
3	Camera 1 model detection	Shape Search III	Measure the position of the feature (mark) registered from the image of camera 1.	The feature (mark) to be detected by the camera 1 is registered as a model.
4	Data Output	Fieldbus Data Output	In the EtherCAT communication, the reference position and measurement position of Mark 0 and Mark 1 are output.	Set the data to be output (Reference position, Measurement position, Overall Judgment).

Overview of Sample Programming

The sample programming for alignment control is roughly divided into the following processes.

Section Structure



N o.	Process	Description	Ladder diagram	Structured text (ST)
1	Default Setting	Set various parameters of FB / FUN used in this sample programming as variables.	Program name: CtrlStage_LD_Samp Section name: Settings Rung number: 0 to 4	Program name: CtrlStage_ST_Samp Row number: 2 to 66
2	Alignment Operation Start/Stop Sequence	This is a sequence circuit for Alignment control Start/Stop. It accepts Start/Stop instructions from a high level sequence program, or display device. If the alignment operation is not completed after 2 seconds from the start of alignment, a stop instruction is given as a timeout.	Program name: CtrlStage_LD_Samp Section name: StageControl Rung number: 0 to 10	Program name: CtrlStage_ST_Samp Row number: 83 to 150

No.	Process	Description	Ladder diagram	Structured text (ST)
3	Measurement Result Input from FH	<p>Converts the measurement result (reference position, measurement position) from the vision sensor controller so that it can be input to FB / FUN and assigns it to a variable. Measurement results from the vision sensor controller are input via device variables. This sample programming shows the case where the measurement result is output with DINT. In the FH Series vision sensor controller, in the case of DINT output, since the measurement result has a fixed-point output format of three digits after the decimal point, use the value divided by 1000 as shown in this circuit.</p>	<p>Program name: CtrlStage_LD_Samp Section name: StageControl Rung number: 11</p>	<p>Program name: CtrlStage_ST_Samp Row number: 152 to 162</p>
4	Alignment Control	<p>Converts the reference position and measurement position obtained from the vision sensor controller to the stage coordinates and calculates the position and angle. The stage is moved based on the obtained position and angle information. The following FB/FUN are used. Coordinate conversion FUN (AffineTrans) Position and Angle Calculation FUN (CalcPosAngle) Stage Control FB (CtrlStage)</p>	<p>Program name: CtrlStage_LD_Samp Section name: StageControl Rung number: 12 to 18</p>	<p>Program name: CtrlStage_ST_Samp Row number: 164 to 227</p>
5	Trigger Output to the FH	<p>Sends the measurement trigger output that the Stage Control FB (CtrlStage) outputs to the vision sensor controller. The measurement trigger output to the vision sensor controller is output using a device variable.</p>	<p>Program name: CtrlStage_LD_Samp Section name: FH_Control Rung number: 0</p>	<p>Program name: CtrlStage_ST_Samp Row number: 355 to 356</p>

Primary Variables

Name	Data type	Default	Comment
CalibParams	ARRAY[0..3] OF OmronLib\VF_Alignment\sCALIB_PARAMS	—	Variables for storing the calibration parameters.
Alignment-Params	OmronLib\VF_Alignment\sALIGNMENT_PARAMS	—	Variables for storing the alignment parameters.
Start	BOOL	FALSE	Alignment control starts when this variable changes from FALSE to TRUE.
Stop	BOOL	FALSE	Alignment control stops when this variable changes from FALSE to TRUE.
Reset	BOOL	FALSE	Fault Reset is executed when this variable changes from FALSE to TRUE.
ServoON_X	BOOL	FALSE	This variable is TRUE when the X axis Servo is ON.
ServoON_Y	BOOL	FALSE	This variable is TRUE when the Y axis Servo is ON.
ServoON_TH	BOOL	FALSE	This variable is TRUE when the θ axis Servo is ON.
AlignmentHomePos	BOOL	FALSE	An internal variable indicating when alignment control sequence can be started.
Alignment-Start	BOOL	FALSE	An internal variable that starts the alignment control sequence.
Alignment-Stop	BOOL	FALSE	An internal variable that stops the alignment control sequence.
AlignmentEnd	BOOL	FALSE	An internal variable that ends the alignment control sequence.
AL_AlignmentT-OVER	BOOL	FALSE	This is an alarm flag indicating that an alignment control timeout occurred.
AL_CtrlStage	BOOL	FALSE	This is an alarm flag indicating that an error occurred in CtrlStage.
AL_MC_EC_Error	BOOL	FALSE	This is an alarm flag that indicates that an error occurred in the Motion Control Function Module of the CPU Unit or in the EtherCAT Master Function Module.
AlignmentAlarm	BOOL	FALSE	This variable is TRUE when there is an alignment control alarm.
CtrlStage_T Over	TON	—	When the alignment time exceeds the set time, a timeout occurs and alignment control is stopped.
CtrlStage_Done	BOOL	FALSE	This is a variable for storing Done output from CtrlStage.
CtrlStage_Busy	BOOL	FALSE	This is a variable for storing Busy output from CtrlStage.
CtrlStage_CommandAborted	BOOL	FALSE	This is a variable for storing CommandAborted output from CtrlStage.
CtrlStage_Error	BOOL	FALSE	This is a variable for storing Error output from CtrlStage.

Name	Data type	Default	Comment
Trigger	BOOL	FALSE	This is a variable that outputs the imaging trigger to send to the vision sensor.
TargetMark-PosCamera	ARRAY[0..3] OF OmronLib\VF_Alignment\POSITION	—	Variables for storing the reference position for cameras 0 and 1. Index 0 of the array is camera 0, and Index 1 is the reference position of camera 1.
MeasMark-PosCamera	ARRAY[0..3] OF OmronLib\VF_Alignment\POSITION	—	Variables for storing the measurement position for cameras 0 and 1. Index 0 of the array is camera 0, and Index 1 is the measurement position of camera 1.
TargetMark-PosStage	ARRAY[0..3] OF OmronLib\VF_Alignment\POSITION	—	Variables for storing the reference position after Affine conversion.
MeasMark-PosStage	ARRAY[0..3] OF OmronLib\VF_Alignment\POSITION	—	Variables for storing the measurement position after Affine conversion.
TargetMark-PosAlign	OmronLib\VF_Alignment\POSITION	—	Variables for storing the reference position of the position angle calculation.
MeasMark-PosAlign	OmronLib\VF_Alignment\POSITION	—	Variables for storing the measurement position of the position angle calculation.
ret_Target_AffineTrans	UINT	0	Variable to store the returned value of AffineTrans when calculating the reference position.
ret_Meas_AffineTrans	UINT	0	Variable to store the returned value of AffineTrans when calculating the measurement position.
ret_Target_CalcPosAngle	UINT	0	Variable to store the returned value of CalcPosAngle when calculating the reference position.
ret_Meas_CalcPosAngle	UINT	0	Variable to store the returned value of CalcPosAngle when calculating the measurement position.
TargetMark-Pos-Stage_OK	ARRAY[0..3] OF OmronLib\VF_Alignment\POSITION	—	Variables for storing the reference position after Affine conversion ends normally.
MeasMark-Pos-Stage_OK	ARRAY[0..3] OF OmronLib\VF_Alignment\POSITION	—	Variables for storing the measurement position after Affine conversion ends normally.
TargetMark-PosA-align_OK	OmronLib\VF_Alignment\POSITION	—	Variables for storing the reference position when position angle calculation ends normally.
MeasMark-PosA-align_OK	OmronLib\VF_Alignment\POSITION	—	Variables for storing the measurement position when position angle calculation ends normally.
CalcPos-Target	ARRAY[0..3] OF BOOL	FALSE	Set the points for position angle calculation. Since this sample programming applies to points 0 and 1, set Index 0 and Index 1 of the array to TRUE.
CtrlStage_instance	\\OmronLib\VF_Alignment\CtrlStage	—	Instance of the Stage Control FB (CtrlStage).

Ladder Diagram

Default setting

0 Stage parameter settings

```

1 | P_First_Run | 1 | AlignmentParams.StageParams.StageType := 1; // Stage type : XYθ stage
2 | | 2 | AlignmentParams.StageParams.AxNo[0] := 0; // X axis settings
3 | | 3 | AlignmentParams.StageParams.AxNo[1] := 1; // Y axis settings
4 | | 4 | AlignmentParams.StageParams.AxNo[2] := 2; // θ axis settings
5 | | 5 | AlignmentParams.StageParams.StopDec[0] := LREAL#100.0; // X stop of deceleration
6 | | 6 | AlignmentParams.StageParams.StopDec[1] := LREAL#100.0; // Y stop of deceleration
7 | | 7 | AlignmentParams.StageParams.StopDec[2] := LREAL#55.664195; // θ stop of deceleration
8 | | 8 | AlignmentParams.StageParams.YXTHStageParams.Rotation := UINT#0; // θ direction : positive
9 | | 9 | AlignmentParams.StageParams.YXTHStageParams.TH_Type := UINT#1; // θ type : linear
10 | | 10 | AlignmentParams.StageParams.YXTHStageParams.TH_Length := LREAL#100.0; // θ length
11 | | 11 | AlignmentParams.StageParams.YXTHStageParams.TH_Offset := LREAL#0.0; // θ offset of angle

```

1 Axis movement parameter settings

```

1 | P_First_Run | 1 | AlignmentParams.CalcVirtualMoveParams.MoveParams_X.Kp := LREAL#16.0; // X proportio
2 | | 2 | AlignmentParams.CalcVirtualMoveParams.MoveParams_X.UseMaxVal := FALSE; // X velocity l
3 | | 3 | AlignmentParams.CalcVirtualMoveParams.MoveParams_X.MaxVel := LREAL#5.0; // X velocity l
4 | | 4 | AlignmentParams.CalcVirtualMoveParams.MoveParams_X.UseAccDec := FALSE; // X accelerat
5 | | 5 | AlignmentParams.CalcVirtualMoveParams.MoveParams_X.AccDec := LREAL#100.0; // X accelerat
6 | | 6 | AlignmentParams.CalcVirtualMoveParams.MoveParams_X.ImagingLimitVel := LREAL#1.0; // X velocity l
7 | | 7 | AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.Kp := LREAL#16.0; // Y proportio
8 | | 8 | AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.UseMaxVal := FALSE; // Y velocity l
9 | | 9 | AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.MaxVel := LREAL#5.0; // Y velocity l
10 | | 10 | AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.UseAccDec := FALSE; // Y accelerat
11 | | 11 | AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.AccDec := LREAL#100.0; // Y accelerat
12 | | 12 | AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.ImagingLimitVel := LREAL#1.0; // Y velocity l
13 | | 13 | AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.Kp := LREAL#16.0; // θ proportio
14 | | 14 | AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.MaxVel := FALSE; // θ velocity l
15 | | 15 | AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.MaxVel := LREAL#2.78320975; // θ velocity l
16 | | 16 | AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.UseAccDec := FALSE; // θ accelerat
17 | | 17 | AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.AccDec := LREAL#55.664195; // θ accelerat
18 | | 18 | AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.ImagingLimitVel := LREAL#0.55664195; // θ velocity l
19 | | 19 | AlignmentParams.CalcVirtualMoveParams.InPosMode := UINT#0; // judgement
20 | | 20 | AlignmentParams.CalcVirtualMoveParams.InPosRange[0] := LREAL#0.001; // threshold c
21 | | 21 | AlignmentParams.CalcVirtualMoveParams.InPosRange[1] := LREAL#0.5; // threshold c
22 | | 22 | AlignmentParams.CalcVirtualMoveParams.InPosRange[2] := LREAL#0.0; // threshold c
23 | | 23 | AlignmentParams.CalcVirtualMoveParams.InPosTarget[0] := TRUE; // target of ra
24 | | 24 | AlignmentParams.CalcVirtualMoveParams.InPosTarget[1] := TRUE; // target of ai
25 | | 25 | AlignmentParams.CalcVirtualMoveParams.InPosTarget[2] := FALSE; // target of ai

```

2 Curve parameter settings

```

1 | P_First_Run | 1 | AlignmentParams.CalcCurveParams.UseCurve := TRUE; // use curve
2 | | 2 | AlignmentParams.CalcCurveParams.UseCurveSequence := TRUE; // use curve sequence
3 | | 3 | AlignmentParams.CalcCurveParams.UseConnectingVel := TRUE; // use speed connect
4 | | 4 | AlignmentParams.CalcCurveParams.MinCurveTime := LREAL#32.0; // min. curve time : 32ms
5 | | 5 | AlignmentParams.CalcCurveParams.MaxCurveTime := LREAL#1000.0; // max. curve time : 1000ms
6 | | 6 | AlignmentParams.CalcCurveParams.DistNumber := 1; // dist. number : 1

```

3 Imaging parameter settings

```

1 | P_First_Run | 1 | AlignmentParams.ImagingParams.ShutterSpeed := LREAL#10.0; // shutter speed
2 | | 2 | CalcPosTarget[0] := TRUE; // target of camera 0 settings
3 | | 3 | CalcPosTarget[1] := TRUE; // target of camera 1 settings
4 | | 4 | CalcPosTarget[2] := FALSE; // target of camera 2 settings
5 | | 5 | CalcPosTarget[3] := FALSE; // target of camera 3 settings

```

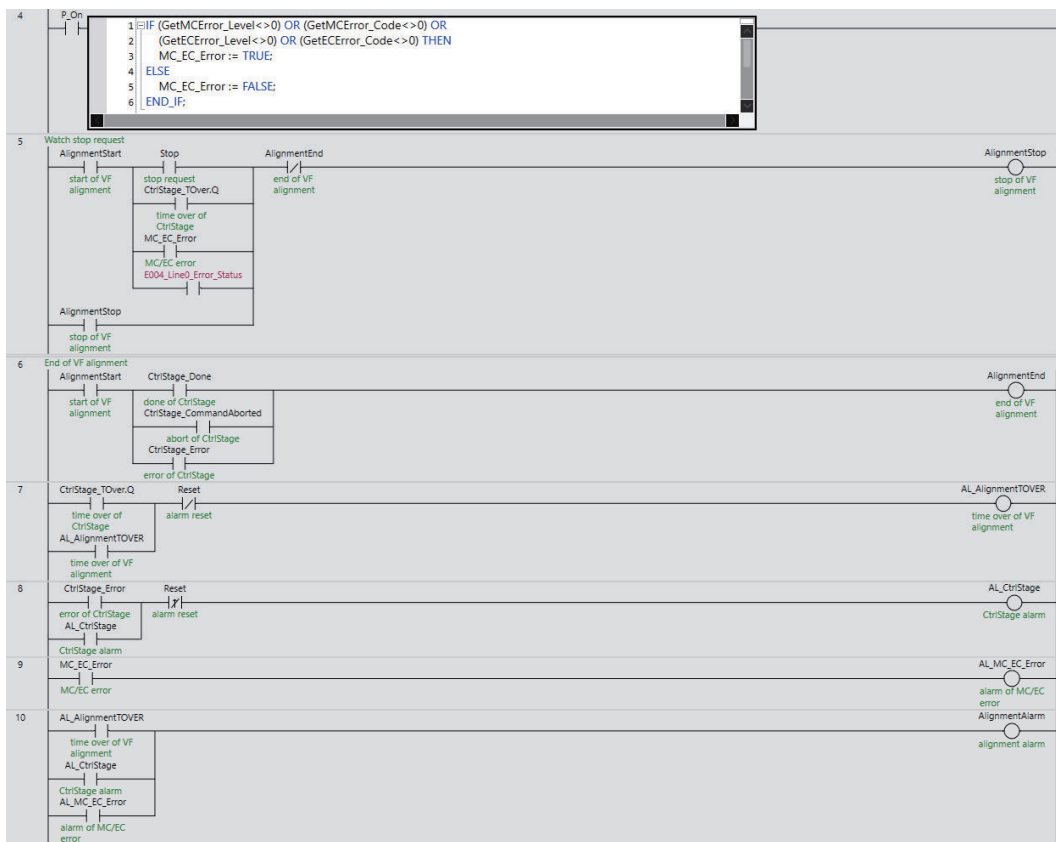
4 Judgement parameter settings

```

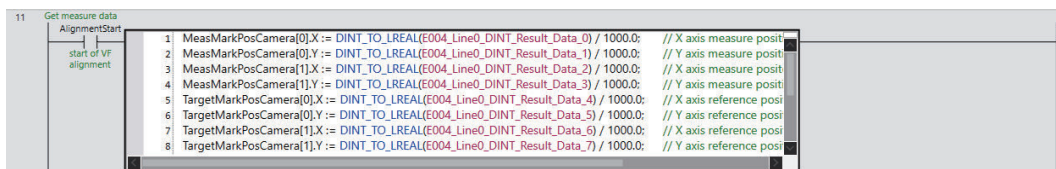
1 | P_First_Run | 1 | AlignmentParams.JudgeParams.InPosMode := UINT#0; // judgement mode : radius, angle
2 | | 2 | AlignmentParams.JudgeParams.InPosRange[0] := LREAL#0.001; // threshold of radius
3 | | 3 | AlignmentParams.JudgeParams.InPosRange[1] := LREAL#0.5; // threshold of angle
4 | | 4 | AlignmentParams.JudgeParams.InPosRange[2] := LREAL#0.000; // threshold of angle
5 | | 5 | AlignmentParams.JudgeParams.InPosTarget[0] := TRUE; // target of radius
6 | | 6 | AlignmentParams.JudgeParams.InPosTarget[1] := TRUE; // target of angle
7 | | 7 | AlignmentParams.JudgeParams.InPosTarget[2] := FALSE; // target of angle
8 | | 8 | AlignmentParams.JudgeParams.TargetMark[0] := CalcPosTarget[0]; // target of camera 0 : TRUE
9 | | 9 | AlignmentParams.JudgeParams.TargetMark[1] := CalcPosTarget[1]; // target of camera 1 : TRUE
10 | | 10 | AlignmentParams.JudgeParams.TargetMark[2] := CalcPosTarget[2]; // target of camera 2 : FALSE
11 | | 11 | AlignmentParams.JudgeParams.TargetMark[3] := CalcPosTarget[3]; // target of camera 3 : FALSE

```

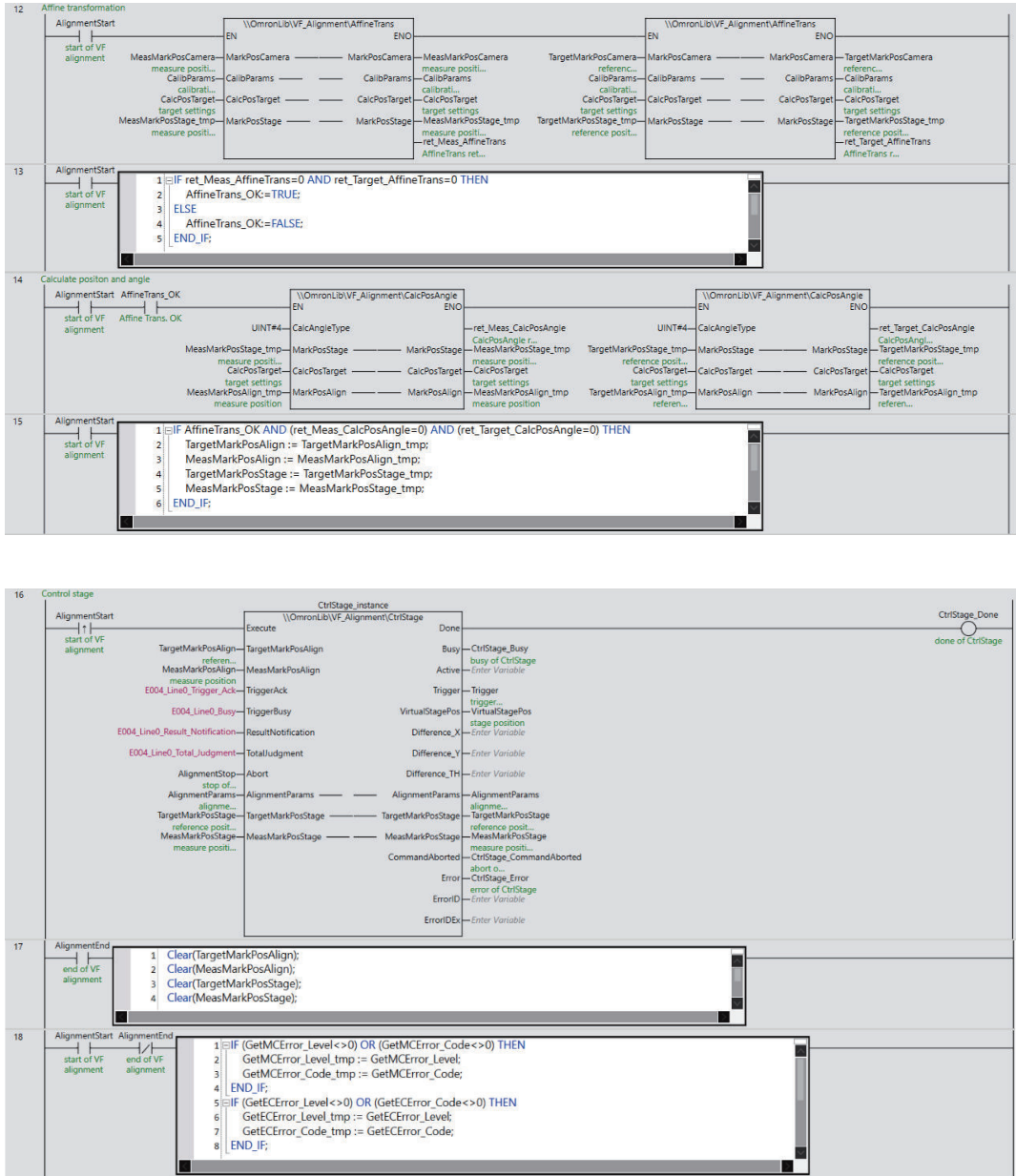
Alignment Control Start/Stop Sequence



Measurement Result Input from FH



Alignment Control



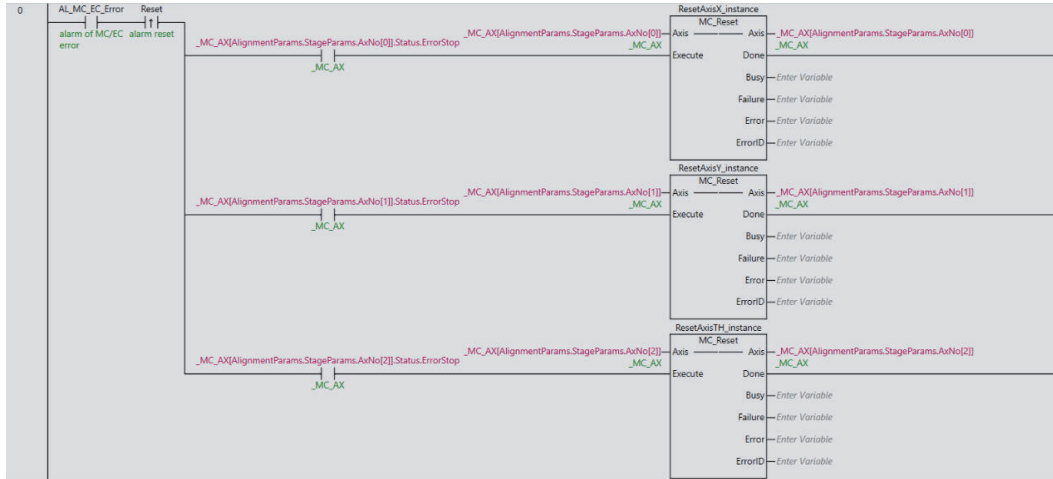
Trigger Output to the FH

It is the same circuit as the auto-calibration sample programming.



Axis Control Error Reset Processing

It is the same circuit as the auto-calibration sample programming.



Structured Text (ST)

Default setting

```
// Initial settings
IF P_First_Run THEN
  // Stage parameter settings
  AlignmentParams.StageParams.StageType := 1;    // Stage type : XYθ stage
  AlignmentParams.StageParams.AxNo[0] := 0;      // X axis settings
  AlignmentParams.StageParams.AxNo[1] := 1;      // Y axis settings
  AlignmentParams.StageParams.AxNo[2] := 2;      // θ axis settings
  AlignmentParams.StageParams.StopDec[0] := LREAL#100.0;
                                                    // X stop of deceleration
  AlignmentParams.StageParams.StopDec[1] := LREAL#100.0;
                                                    // Y stop of deceleration
  AlignmentParams.StageParams.StopDec[2] := LREAL#55.664195;
                                                    // θ stop of deceleration
  AlignmentParams.StageParams.XYTHStageParams.Rotation := UINT#0;
                                                    // θ rotation : positive
  AlignmentParams.StageParams.XYTHStageParams.TH_Type := UINT#1;
                                                    // θ type : linear
  AlignmentParams.StageParams.XYTHStageParams.TH_Length := LREAL#100.0;
                                                    // θ length
  AlignmentParams.StageParams.XYTHStageParams.TH_Offset := LREAL#0.0;
                                                    // θ offset of angle

  // Axis movement parameter settings
  AlignmentParams.CalcVirtualMoveParams.MoveParams_X.Kp := LREAL#16.0;
                                                    // X proportional gain
  AlignmentParams.CalcVirtualMoveParams.MoveParams_X.UseMaxVel := FALSE;
                                                    // X velocity limit select
  AlignmentParams.CalcVirtualMoveParams.MoveParams_X.MaxVel := LREAL#5.0;
                                                    // X velocity limit
  AlignmentParams.CalcVirtualMoveParams.MoveParams_X.UseAccDec := FALSE;
```

```

// X acceleration limit select
AlignmentParams.CalcVirtualMoveParams.MoveParams_X.AccDec := LREAL#100.0;
// X acceleration limit
AlignmentParams.CalcVirtualMoveParams.MoveParams_X.ImagingLimitVel
:= LREAL#1.0; // X velocity limit of imaging
AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.Kp := LREAL#16.0;
// Y proportional gain
AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.UseMaxVel := FALSE;
// Y velocity limit select
AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.MaxVel := LREAL#5.0;
// Y velocity limit
AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.UseAccDec := FALSE;
// Y acceleration limit select
AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.AccDec := LREAL#100.0;
// Y acceleration limit
AlignmentParams.CalcVirtualMoveParams.MoveParams_Y.ImagingLimitVel
:= LREAL#1.0; // Y velocity limit of imaging
AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.Kp := LREAL#16.0;
// θ proportional gain
AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.UseMaxVel := FALSE;
// θ velocity limit select
AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.MaxVel := LREAL#2.78320975;
// θ velocity limit
AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.UseAccDec := FALSE;
// θ acceleration limit select
AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.AccDec := LREAL#55.664195;
// θ acceleration limit
AlignmentParams.CalcVirtualMoveParams.MoveParams_TH.ImagingLimitVel
:= LREAL#0.55664195; // θ velocity limit of imaging
AlignmentParams.CalcVirtualMoveParams.InPosMode := UINT#0;
// judgment mode : radius, angle
AlignmentParams.CalcVirtualMoveParams.InPosRange[0] := LREAL#0.001;
// threshold of radius
AlignmentParams.CalcVirtualMoveParams.InPosRange[1] := LREAL#0.5;
AlignmentParams.CalcVirtualMoveParams.InPosRange[2] := LREAL#0.0;
// threshold of angle
AlignmentParams.CalcVirtualMoveParams.InPosTarget[0] := TRUE;
// target of radius
AlignmentParams.CalcVirtualMoveParams.InPosTarget[1] := TRUE;
AlignmentParams.CalcVirtualMoveParams.InPosTarget[2] := FALSE;
// target of angle

// Curve parameter settings
AlignmentParams.CalcCurveParams.UseCurve := TRUE;
// use curve
AlignmentParams.CalcCurveParams.UseCurveSequence := TRUE;

```

```

// use curve sequence
AlignmentParams.CalcCurveParams.UseConnectingVel := TRUE;
// use speed connect
AlignmentParams.CalcCurveParams.MinCurveTime := LREAL#32.0;
// min. curve time : 32ms
AlignmentParams.CalcCurveParams.MaxCurveTime := LREAL#1000.0;
// max. curve time : 1000ms
AlignmentParams.CalcCurveParams.DistNumber := 1;
// dist. number : 1

// Image parameter settings
AlignmentParams.ImagingParams.ShutterSpeed := LREAL#10.0; // shutter speed
CalcPosTarget[0] := TRUE; // target of camera 0 settings
CalcPosTarget[1] := TRUE; // target of camera 1 settings
CalcPosTarget[2] := FALSE; // target of camera 2 settings
CalcPosTarget[3] := FALSE; // target of camera 3 settings

// Judgment parameter settings
AlignmentParams.JudgeParams.InPosMode := UINT#0;
// judgment mode : radius, angle
AlignmentParams.JudgeParams.InPosRange[0] := LREAL#0.001;
// threshold of radius
AlignmentParams.JudgeParams.InPosRange[1] := LREAL#0.5;
// threshold of angle
AlignmentParams.JudgeParams.InPosRange[2] := LREAL#0.000;
AlignmentParams.JudgeParams.InPosTarget[0] := TRUE;
// target of radius
AlignmentParams.JudgeParams.InPosTarget[1] := TRUE;
AlignmentParams.JudgeParams.InPosTarget[2] := FALSE;
// target of angle
AlignmentParams.JudgeParams.TargetMark[0] := CalcPosTarget[0];
// target of camera 0 : TRUE
AlignmentParams.JudgeParams.TargetMark[1] := CalcPosTarget[1];
// target of camera 1 : TRUE
AlignmentParams.JudgeParams.TargetMark[2] := CalcPosTarget[2];
// target of camera 2 : FALSE
AlignmentParams.JudgeParams.TargetMark[3] := CalcPosTarget[3];
// target of camera 3 : FALSE

END_IF;

```

Alignment Control Start/Stop Sequence

```

// VF alignment Start/Stop sequence
R_TRIG_Start(Clk:=Start);
// Watch start condition
IF ServoOn_X AND ServoOn_Y AND ServoOn_TH AND NOT(CtrlStage_Busy)
AND NOT(AlignmentAlarm) THEN
AlignmentHomePos := TRUE;

```

```

ELSE
    AlignmentHomePos := FALSE;
END_IF;
// Start of VF alignment
IF R_TRIG_Start.Q AND AlignmentHomePos AND NOT(AlignmentEnd) THEN
    AlignmentStart := TRUE;
END_IF;
// Watch VF alignment time
CtrlStage_TOver(In:=AlignmentStart, PT:=T#2.0s);
// Watch MC error and ECAT error
GetMCError(
    GetMCError_Level,
    GetMCError_Code
);
GetECError(
    GetECError_Level,
    GetECError_Code
);
IF (GetMCError_Level<>0) OR (GetMCError_Code<>0) OR (GetECError_Level<>0)
    OR (GetECError_Code<>0) THEN
    MC_EC_Error := TRUE;
ELSE
    MC_EC_Error := FALSE;
END_IF;
// Watch stop request
IF AlignmentStart AND (Stop OR CtrlStage_TOver.Q OR MC_EC_Error
    OR E004_Line0_Error_Status) AND NOT(AlignmentEnd) THEN
    AlignmentStop := TRUE;
END_IF;
IF AlignmentEnd THEN
    AlignmentStart := FALSE;
    AlignmentStop := FALSE;
    AlignmentEnd := FALSE;

    Clear(TargetMarkPosAlign);
    Clear(MeasMarkPosAlign);
    Clear(TargetMarkPosStage);
    Clear(MeasMarkPosStage);
END_IF;
// End of VF alignment
IF AlignmentStart AND (CtrlStage_Done OR CtrlStage_CommandAborted
    OR CtrlStage_Error) THEN
    AlignmentEnd := TRUE;
END_IF;
// Watch error
IF CtrlStage_TOver.Q THEN
    AL_AlignmentTOVER := TRUE; // Time over of VF alignment

```



```

END_IF;
IF CtrlStage_Error THEN
    AL_CtrlStage := TRUE;      // CtrlStage alarm
END_IF;
IF MC_EC_Error THEN
    AL_MC_EC_Error := TRUE;
ELSE
    AL_MC_EC_Error := FALSE;
END_IF;
IF AL_AlignmentTOVER OR AL_CtrlStage OR AL_MC_EC_Error THEN
    AlignmentAlarm := TRUE;
END_IF;
// Alarm reset
R_TRIG_Reset(Clk:=Reset);
IF R_TRIG_Reset.Q AND NOT(AlignmentStart) THEN
    AL_AlignmentTOVER := FALSE;
    AL_CtrlStage := FALSE;
    AlignmentAlarm := FALSE;
END_IF;

```

Measurement Result Input from FH

```

// Get measure data
IF AlignmentStart THEN
    MeasMarkPosCamera[0].X := DINT_TO_LREAL(E004_Line0_DINT_Result_Data_0)
    / LREAL#1000.0;      // X axis measure position of camera 0
    MeasMarkPosCamera[0].Y := DINT_TO_LREAL(E004_Line0_DINT_Result_Data_1)
    / LREAL#1000.0;      // Y axis measure position of camera 0
    MeasMarkPosCamera[1].X := DINT_TO_LREAL(E004_Line0_DINT_Result_Data_2)
    / LREAL#1000.0;      // X axis measure position of camera 1
    MeasMarkPosCamera[1].Y := DINT_TO_LREAL(E004_Line0_DINT_Result_Data_3)
    / LREAL#1000.0;      // Y axis measure position of camera 1
    TargetMarkPosCamera[0].X := DINT_TO_LREAL(E004_Line0_DINT_Result_Data_4)
    / LREAL#1000.0;      // X axis reference position of camera 0
    TargetMarkPosCamera[0].Y := DINT_TO_LREAL(E004_Line0_DINT_Result_Data_5)
    / LREAL#1000.0;      // Y axis reference position of camera 0
    TargetMarkPosCamera[1].X := DINT_TO_LREAL(E004_Line0_DINT_Result_Data_6)
    / LREAL#1000.0;      // X axis reference position of camera 1
    TargetMarkPosCamera[1].Y := DINT_TO_LREAL(E004_Line0_DINT_Result_Data_7)
    / LREAL#1000.0;      // Y axis reference position of camera 1
END_IF;

```

Alignment Control

```

// Alignment control
IF AlignmentStart THEN
    // Affine transformation
    ret_Meas_AffineTrans := \\OmronLib\VF_Alignment\AffineTrans(
        MarkPosCamera := MeasMarkPosCamera,

```

```

    CalibParams := CalibParams,
    CalcPosTarget := CalcPosTarget,
    MarkPosStage := MeasMarkPosStage_tmp
  );
ret_Target_AffineTrans := \\OmronLib\VF_Alignment\AffineTrans(
  MarkPosCamera := TargetMarkPosCamera,
  CalibParams := CalibParams,
  CalcPosTarget := CalcPosTarget,
  MarkPosStage := TargetMarkPosStage_tmp
);
IF (ret_Meas_AffineTrans=0) AND (ret_Target_AffineTrans=0) THEN
  AffineTrans_OK := TRUE;
ELSE
  AffineTrans_OK := FALSE;
END_IF;
IF AffineTrans_OK THEN
  // Calculate position and angle
  ret_Meas_CalcPosAngle := \\OmronLib\VF_Alignment\CalcPosAngle(
    CalcAngleType := UINT#4,
    MarkPosStage := MeasMarkPosStage_tmp,
    CalcPosTarget := CalcPosTarget,
    MarkPosAlign := MeasMarkPosAlign_tmp
  );
  ret_Target_CalcPosAngle := \\OmronLib\VF_Alignment\CalcPosAngle(
    CalcAngleType := UINT#4,
    MarkPosStage := TargetMarkPosStage_tmp,
    CalcPosTarget := CalcPosTarget,
    MarkPosAlign := TargetMarkPosAlign_tmp
  );
END_IF;
IF AffineTrans_OK AND (ret_Meas_CalcPosAngle=0) AND (ret_Target_CalcPosAngle=0)
  THEN
  TargetMarkPosAlign := TargetMarkPosAlign_tmp;
  MeasMarkPosAlign := MeasMarkPosAlign_tmp;
  TargetMarkPosStage := TargetMarkPosStage_tmp;
  MeasMarkPosStage := MeasMarkPosStage_tmp;
END_IF;
END_IF;

// Control stage
R_TRIG_AlignmentStart(Clk:=AlignmentStart);
CtrlStage_instance(
  Execute := R_TRIG_AlignmentStart.Q,
  TargetMarkPosAlign := TargetMarkPosAlign_OK,
  MeasMarkPosAlign := MeasMarkPosAlign_OK,
  TriggerAck := E004_Line0_Trigger_Ack,
  TriggerBusy := E004_Line0_Busy,

```

```

ResultNotification := E004_Line0_Result_Notification,
TotalJudgment := E004_Line0_Total_Judgment,
Abort := AlignmentStop,
AlignmentParams := AlignmentParams,
TargetMarkPosStage := TargetMarkPosStage_OK,
MeasMarkPosStage := MeasMarkPosStage_OK,
Done => CtrlStage_Done,
Busy => CtrlStage_Busy,
Trigger => Trigger,
VirtualStagePos => VirtualStagePos,
CommandAborted => CtrlStage_CommandAborted,
Error => CtrlStage_Error
);

```

Trigger Output to the FH

It is the same circuit as the auto-calibration sample programming.

```

// Output trigger
E004_Line0_Trigger := (Trigger OR CalibTrigger);

```

Axis Control Error Reset Processing

It is the same circuit as the auto-calibration sample programming.

```

// Reset
ResetAxisX_instance(
    Execute := (AL_MC_EC_Error AND R_TRIG_Reset.Q AND
                _MC_AX[AlignmentParams.StageParams.AxNo[0]].Status.ErrorStop),
    Axis := _MC_AX[AlignmentParams.StageParams.AxNo[0]]
);
ResetAxisY_instance(
    Execute := (AL_MC_EC_Error AND R_TRIG_Reset.Q AND
                _MC_AX[AlignmentParams.StageParams.AxNo[1]].Status.ErrorStop),
    Axis := _MC_AX[AlignmentParams.StageParams.AxNo[1]]
);
ResetAxisTH_instance(
    Execute := (AL_MC_EC_Error AND R_TRIG_Reset.Q AND
                _MC_AX[AlignmentParams.StageParams.AxNo[2]].Status.ErrorStop),
    Axis := _MC_AX[AlignmentParams.StageParams.AxNo[2]]
);

```



Additional Information

In this program, the Position and Angle Calculation FUN (CalcPosAngle) is used to calculate the position and rotation angle with respect to the measurement position and reference position. When using two or more alignment marks, an alignment control program can be constructed by replacing Position and Angle Calculation FUN (CalcPosAngle) with Multi-point Position and Angle Calculation FUN (CalcMultiPosAngle). Position and Angle Calculation FUN (CalcPosAngle) uses functions for each measurement position and reference position, but when using Multi-point Position and Angle Calculation FUN (CalcMultiPosAngle), both the measurement position and reference position variables are given to one function.



Appendix

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

A-1	Referring to Library Information	A - 2
A-1-1	Library Attributes, and FB or FUN Attributes	A - 2
A-1-2	Referring to Attributes of Libraries, Function Blocks, and Functions	A - 3
A-2	Referring to Function Block and Function Source Codes	A - 5



A-1 Referring to Library Information

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

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

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

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

Use the Sysmac Studio to access the library information.

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

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

Library Attributes

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

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

*2. It is provided in English and Japanese.

Attributes of Function Blocks and Functions

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

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

*2. It is provided in English and Japanese.

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

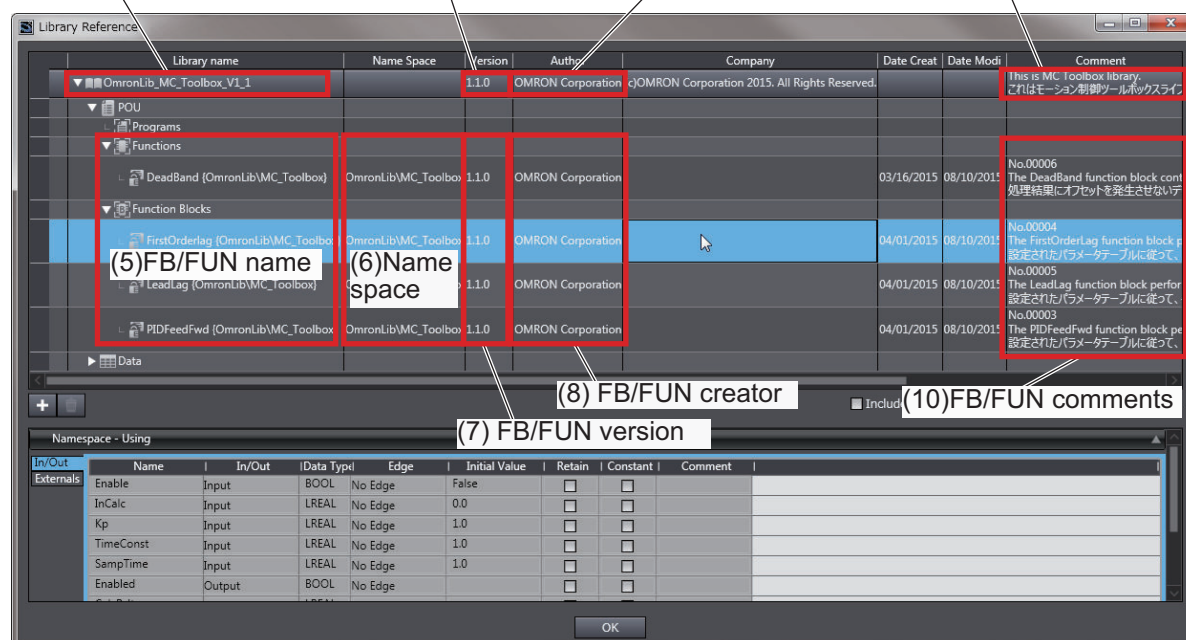
You can refer to the library attributes of library information, and FB or FUN attributes at the following Sysmac Studio locations.

- Library Reference Dialog Box
- Toolbox
- Programming screen

Library Reference Dialog Box

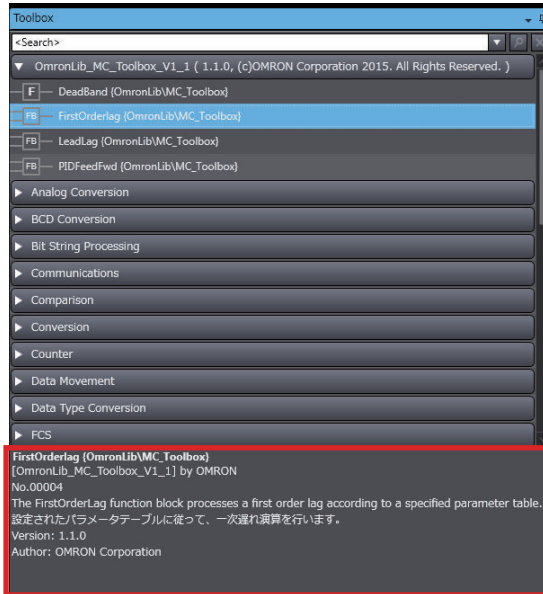
When you refer to the libraries, the library information is displayed at the locations shown below.

- (1) Library file name (2) Library version (3) Library creator (4) Library comments



Toolbox

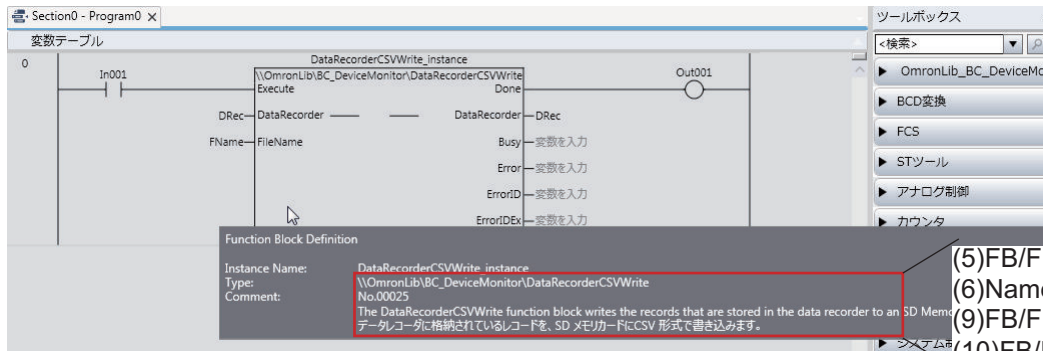
Select a function block or function to display its library information at the bottom of the Toolbox Pane. The text "**by OMRON**" which is shown on the right of the library name (1) indicates that this library was provided by OMRON.



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

Programming Screen

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



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

A-2 Referring to Function Block and Function Source Codes

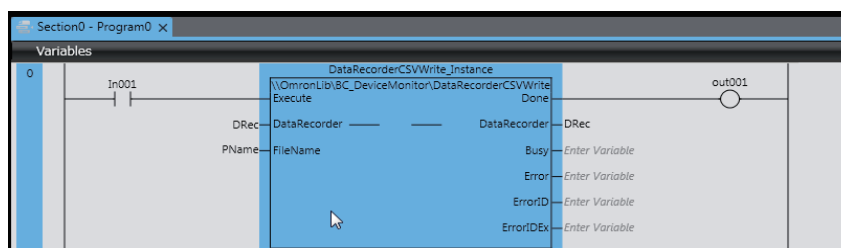
You can refer to the source codes of function blocks and functions provided by OMRON to customize them to suit the user's environment.

User function blocks and user functions can be created based on the copies of these source codes. The following are the examples of items that you may need to customize.

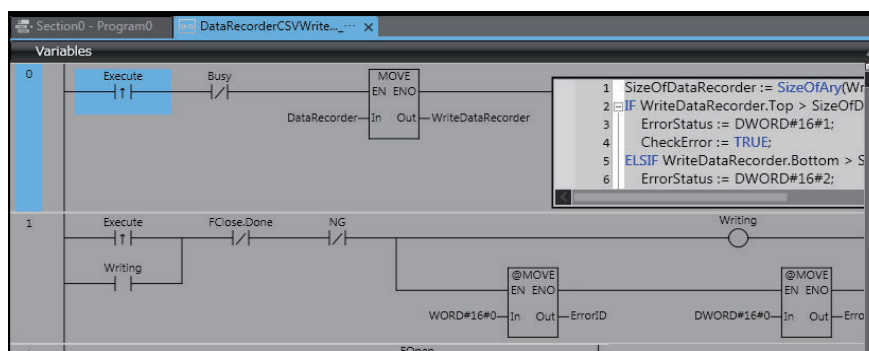
- Customizing the "Array Size" to suit the memory capacity of the user's Controller
- Customizing the "Data Type" to suit the user-defined data types

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

- 1 Select a function block or function in the program.



- 2 Double-click or right-click and select **To Lower Layer** from the menu. The source code is displayed.





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.





Index



Index

A

AffineTrans..... 4 - 22

C

CalcAxisVelocity..... 4 - 54

CalcForwardKinematics..... 4 - 51

CalcInverseKinematics..... 4 - 48

CalcMovement..... 4 - 45

CalcMultiPosAngle..... 4 - 30

CalcPosAngle..... 4 - 25

CtrlStage..... 4 - 35

G

GenerateCalibParams..... 4 - 78

GenerateTrigger..... 4 - 73

J

JudgeAlignmentComplete..... 4 - 64

S

sALIGNMENT_PARAMS..... 4 - 2

sAXIS_MOVE_PARAMS..... 4 - 11

sCALCURVE_PARAMS..... 4 - 13

sCALCVIRTUALMOVE_PARAMS..... 4 - 10

sCALIB_MOVE_PARAMS..... 4 - 15

sCALIB_PARAMS..... 4 - 2

sIMAGING_PARAMS..... 4 - 14

sJUDGE_PARAMS..... 4 - 14

sKINEMATICS_PARAMS..... 4 - 9

sPOSITION..... 4 - 2

sSTAGE_PARAMS..... 4 - 3

sUVWR_PARAMS..... 4 - 6

sUVWR_STAGE_PARAMS..... 4 - 5

sXYTH_STAGE_PARAMS..... 4 - 4

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:

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

Cat. No. W608-E1-02

0119