OMRON

Sysmac Library

User's Manual
for Dimension Measurement Library
SYSMAC-XR014

SYSMAC
always in control

W574-E1-04

NOTE

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.

No patent liability is assumed with respect to the use of the information contained herein. Moreover, because OMRON is constantly striving to improve its high-quality products, the information contained in this manual is subject to change without notice. Every precaution has been taken in the preparation of this manual. Nevertheless, OMRON assumes no responsibility for errors or omissions. Neither is any liability assumed for damages resulting from the use of the information contained in this publication.

Trademarks

- 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 registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.
- ODVA, CIP, CompoNet, DeviceNet, and EtherNet/IP are trademarks of ODVA.
- The SD and SDHC logos are trademarks of SD-3C, LLC.

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

Copyrights

Microsoft product screen shots reprinted with permission from Microsoft Corporation.

Introduction

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

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

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

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

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

Features of the Library

Dimension Measurement Library is the system which connects the NJ/NX/NY-series Controller, ZW-8000/7000/5000 Confocal Fiber Displacement Sensor Controller, and E9NC-TA0 Contact-Type Smart Sensor through EtherCAT communications. It is used when you perform various dimension measurements. When you use the Dimension Measurement Library, you can improve measurement accuracy, enlarge measurement range, and reduce programming work.

Intended Audience

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

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

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

Applicable Products

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

Manual Structure

Special Information

Special information in this manual is classified as follows:

Precautions for Safe Use

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

Precautions for Correct Use

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

Additional Information

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

Version Information

Information on differences in specifications and functionality for CPU Units and Industrial PCs with different unit versions and for different versions of the Sysmac Studio are given.

Note References are provided to more detailed or related information.

CONTENTS

Introduction 1
Features of the Library 1
Intended Audience 1
Applicable Products 2

Manual Structure 3
Special Information 3

CONTENTS 4

Terms and Conditions Agreement 6
Warranty, Limitations of Liability 6
Application Considerations 7
Disclaimers 7

Safety Precautions 8

Precautions for Safe Use 10

Related Manuals 11

Revision History 14

Procedure to Use Sysmac Libraries 15
Procedure to Use Sysmac Libraries Installed Using the Installer 16
Procedure to Use Sysmac Libraries Uploaded from a CPU Unit or an Industrial PC 20

Common Specifications of Function Blocks 23
Common Variables 24
Precautions 30

Individual Specifications of Function Blocks 31
CalcPointMeasurement 32
PointMeasure_Deviation 48
PointMeasure_Torsion 63
PointMeasure_Curve 71
PointMeasure_Thickness 79
ZW_ZeroResetControl 86
ZW_CmdControl 92
LineMeasure_Cartesian 108
LineMeasure_Cartesian2 151
LineMeasure_CreateShape2D_Master 176
LineMeasure_CreateShape2D_Master2 195
LineMeasure_CreateShape2D 215
LineMeasure_CreateShape2D2 226
Shape2D_Height 238
Shape2D_Height2 253
Shape2D_Edge 267
Shape2D_Edge2 277
Shape2D_InflectionPoint 288
Shape2D_InflectionPoint2 296
Shape2D_Angle 304
Shape2D_Angle2 312
Shape2D_Area 320
Shape2D_Area2 329
Shape2D_Compare 337
Shape2D_Compare2 346

Appendix..... 357
Referring to Library Information.....358
Referring to Function Block and Function Source Codes 361

Terms and Conditions Agreement

Warranty, Limitations of Liability

Warranties

- **Exclusive Warranty**

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

- **Limitations**

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

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

- **Buyer Remedy**

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

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

Limitation on Liability; Etc

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

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

Application Considerations

Suitability of Use

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

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

Programmable Products

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

Disclaimers

Performance Data

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

Change in Specifications

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

Errors and Omissions

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

Safety Precautions

Definition of Precautionary Information

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

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

The following notation is used.

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

Symbols

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

Cautions

⚠ Caution	
When you use the touch trigger prove sensor, secure the measured object to be fixed.	
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 so that the actual device operates as intended.	
You must confirm that the user program and parameter values are appropriate to the specifications and operation methods of the devices.	
In the 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.	
Read all related manuals carefully before you use this library.	
The Sysmac Library and manuals are assumed to be used by personnel that is given in Intended Audience in this manual. Otherwise, do not use them.	
The sample programming shows only the portion of a program that uses the function or function block from the library.	
When using actual devices, also program safety circuits, device 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.	
When you apply a value of an output parameter related to measurement to a target position for motion control, proceed with the design considering that the value is in the movable range of the mechanical device and that sensor and work do not interfere.	
When you use the ZW-8000/7000/5000 Series, fix the target to measure or displacement sensor correctly so that the distance between the target and displacement sensor will not change unexpectedly.	
Do not execute multi-execution of motion control instructions for the assigned X axis and Z axis when you execute the LineMeasure_Cartesian and LineMeasure_Cartesian2 function blocks.	
When you execute the LineMeasure_Cartesian and LineMeasure_Cartesian2 function blocks, confirm in advance that the work and sensor are not interference when a servo axis is moved to the Z-axis search end position.	

Precautions for Safe Use

Operation

- Appropriately set the axis resolution and increasing direction of the servo axis actual position.
- Do not use the filter (median filter, low-pass filter, or moving average filter) that is set in the ZW-8000/7000/5000 Series when you execute the LineMeasure_Cartesian and LineMeasure_Cartesian2 function blocks. If you use the filter in the tracer control, operations may become unstable.
- Select the work to measure for which the height is less than the measurement center distance of the ZW-8000/7000/5000 Series when you execute the LineMeasure_Cartesian and LineMeasure_Cartesian2 function blocks. Refer to the *ZW-8000/7000/5000 Confocal Fiber Type Displacement Sensor User's Manual* (Cat. No. Z362) for the measurement center distance.
- When you execute the LineMeasure_Cartesian and LineMeasure_Cartesian2 function blocks, use the Sysmac Studio or the ZW_CmdControl function block and do not change the setting parameters in the ZW-8000/7000/5000 Series. Unexpected operations may happen.

Related Manuals

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

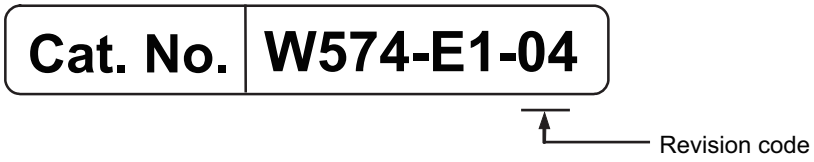
Manual name	Cat. No.	Model numbers	Application	Description
NX-series CPU Unit Hardware User's Manual	W535	NX701-□□□□	Learning the basic specifications of the NX-series NX701 CPU Units, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided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

Manual name	Cat. No.	Model numbers	Application	Description
NY-series IPC Machine Controller Industrial Box PC Hardware User's Manual	W556	NY512-□□□□	Learning the basic specifications of the NY-series Industrial Box PCs, including introductory information, designing, installation, and maintenance. Mainly hardware information is provided	An introduction to the entire NY-series system is provided along with the following information on the Industrial Box PC. Features and system configuration Overview Part names and functions General specifications Installation and wiring Maintenance and inspection
NJ/NX-series CPU Unit Software User's Manual	W501	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning how to program and set up an NJ/NX-series CPU Unit. Mainly software information is provided	The following information is provided on a Controller built with an NJ/NX-series CPU Unit. CPU Unit operation CPU Unit features Initial settings Programming based on IEC 61131-3 language specifications
NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Software User's Manual	W558	NY532-□□□□ NY512-□□□□	Learning how to program and set up the Controller functions of an NY-series Industrial PC	The following information is provided on NY-series Machine Automation Control Software. Controller operation Controller features Controller settings Programming based on IEC 61131-3 language specifications
NJ/NX-series Instructions Reference Manual	W502	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning detailed specifications on the basic instructions of an NJ/NX-series CPU Unit	The instructions in the instruction set (IEC 61131-3 specifications) are described.
NY-series Instructions Reference Manual	W560	NY532-□□□□ NY512-□□□□	Learning detailed specifications on the basic instructions of an NY-series Industrial PC	The instructions in the instruction set (IEC 61131-3 specifications) are described.
NJ/NX-series CPU Unit Motion Control User's Manual	W507	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning about motion control settings and programming concepts of an NJ/NX-series CPU Unit.	The settings and operation of the CPU Unit and programming concepts for motion control are described.
NY-series IPC Machine Controller Industrial Panel PC / Industrial Box PC Motion Control User's Manual	W559	NY532-□□□□ NY512-□□□□	Learning about motion control settings and programming concepts of an NY-series Industrial PC.	The settings and operation of the Controller and programming concepts for motion control are described.
NJ/NX-series Motion Control Instructions Reference Manual	W508	NX701-□□□□ NX102-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	Learning about the specifications of the motion control instructions of an NJ/NX-series CPU Unit.	The motion control instructions are described.
NY-series Motion Control Instructions Reference Manual	W561	NY532-□□□□ NY512-□□□□	Learning about the specifications of the motion control instructions of an NY-series Industrial PC.	The motion control instructions are described.
NJ/NY-series NC Integrated Controller User's Manual	O030	NJ501-5300 NY532-5400	Performing numerical control with NJ/NY-series Controllers.	Describes the functionality to perform the numerical control. Use this manual together with the *NJ/NY-series G code Instructions Reference Manual* (Cat. No. O031) when programming.

Manual name	Cat. No.	Model numbers	Application	Description
NJ/NY-series G code Instructions Reference Manual	O031	NJ501-5300 NY532-5400	Learning about the specifications of the G code/M code instructions.	The G code/M code instructions are described. Use this manual together with the *NJ/NY-series NC Integrated Controller User's Manual* (Cat. No. O030) when programming.
Sysmac Studio Version 1 Operation Manual	W504	SYSMAC -SE2□□□	Learning about the operating procedures and functions of the Sysmac Studio.	Describes the operating procedures of the Sysmac Studio.
CNC Operator Operation Manual	O032	SYSMAC -RTNC0□□□D	Learning an introduction of the CNC Operator and how to use it.	An introduction of the CNC Operator, installation procedures, basic operations, connection operations, and operating procedures for main functions are described.
ZW-8000/7000/5000 series Confocal Fiber Type Displacement Sensor User's Manual'	Z362	ZW-8000/7000/5000	Learning how to use the ZW-8000/7000/5000 series Confocal Fiber Type Displacement Sensors.	Describes the hardware, setup methods and functions of the ZW-8000/7000/5000 series Confocal Fiber Type Displacement Sensors.

Revision History

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



Revision code	Date	Revised content
01	November 2016	Original production
02	December 2016	Corrected mistakes.
03	March 2017	Added compatible models, Change library file name
04	January 2019	• Added compatible models. • Added FBs/FUNs that support variable-length arrays.

Procedure to Use Sysmac Libraries

Procedure to Use Sysmac Libraries Installed Using the Installer

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

There are two ways to use libraries.

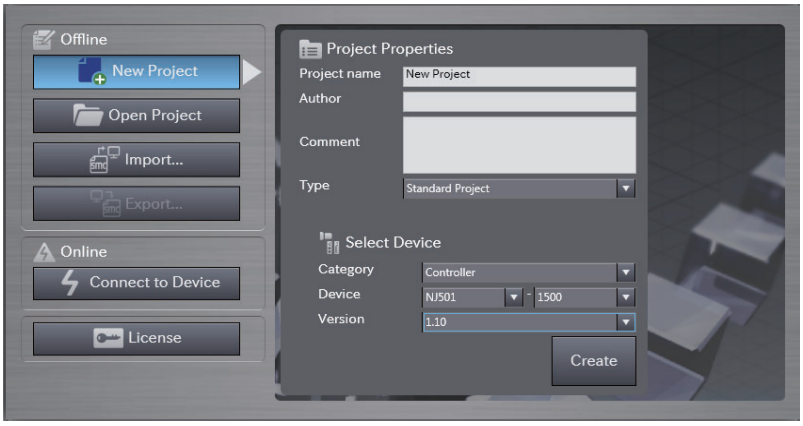
- Using newly installed Sysmac Libraries
- Using upgraded Sysmac Libraries

Version Information

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

Using Newly Installed Libraries

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

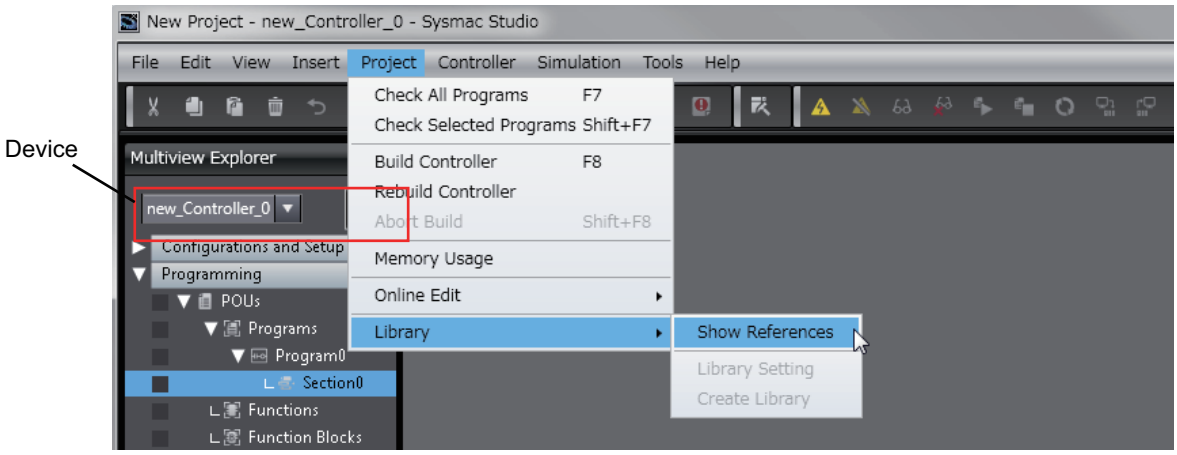


Precautions for Correct Use

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

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

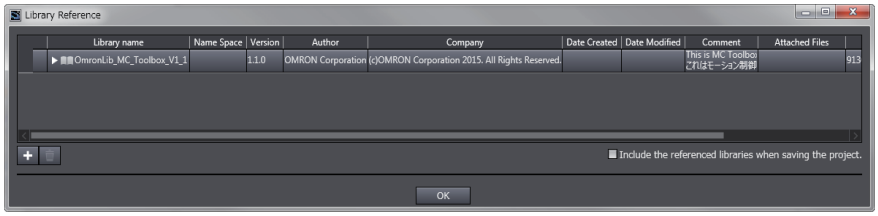
2 Select **Project** – **Library** – **Show References**.



Precautions for Correct Use

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

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



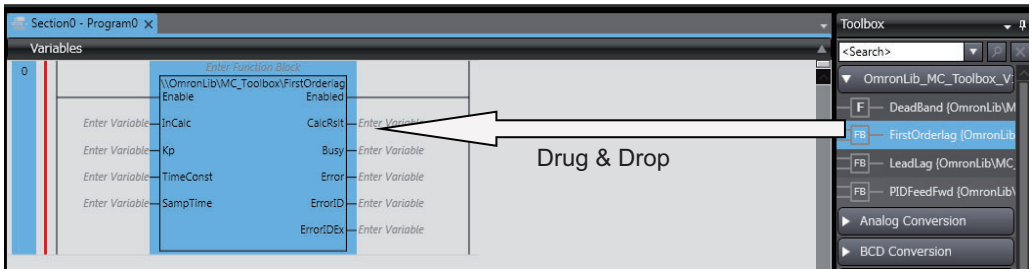
The Sysmac Library file is read into the project.

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

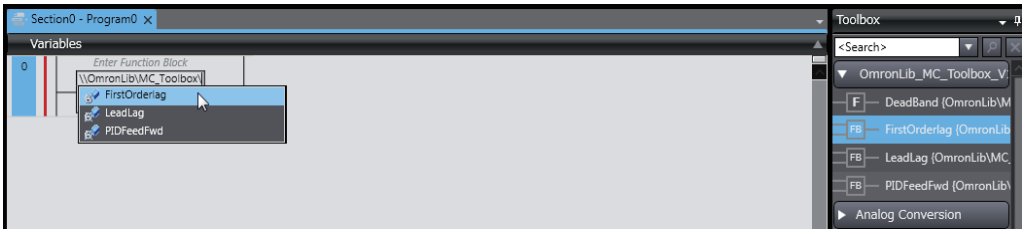
For the procedure for adding and setting libraries in the above screen, refer to the *Sysmac Studio Version 1 Operation Manual* (Cat. No. W504).

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

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



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



Precautions for Correct Use

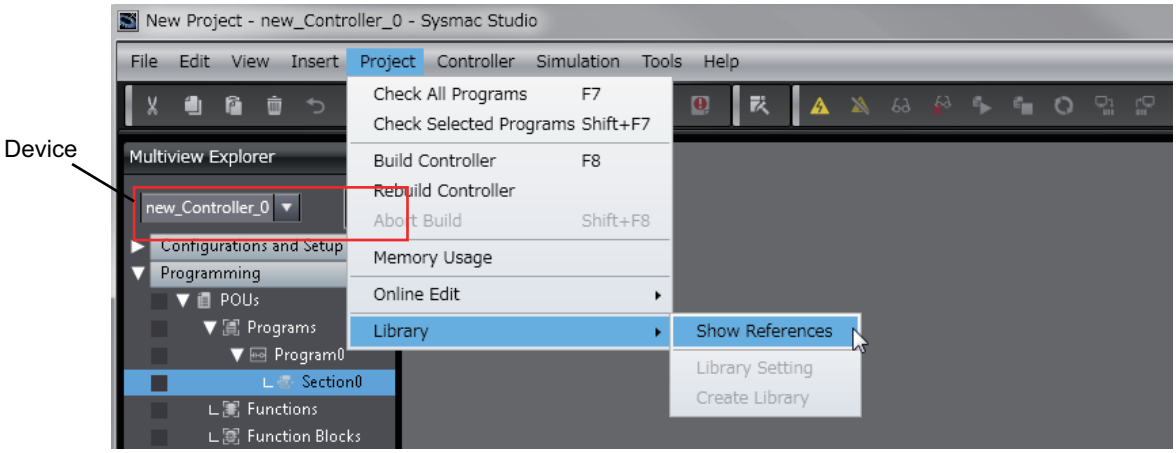
After you upgrade the Sysmac Studio, check all programs and make sure that there is no error of the program check results on the Build Tab Page.

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

Using Upgraded Libraries

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

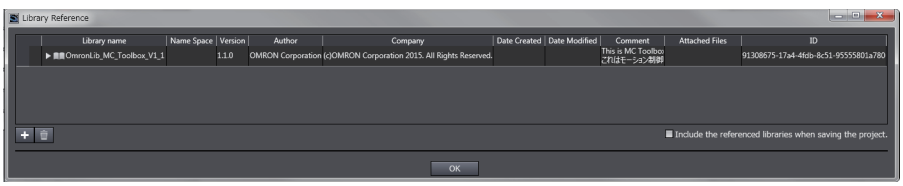
2 Select **Project** – **Library** – **Show References**.



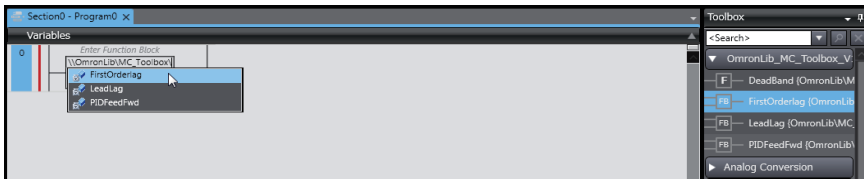
Precautions for Correct Use

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

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



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



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

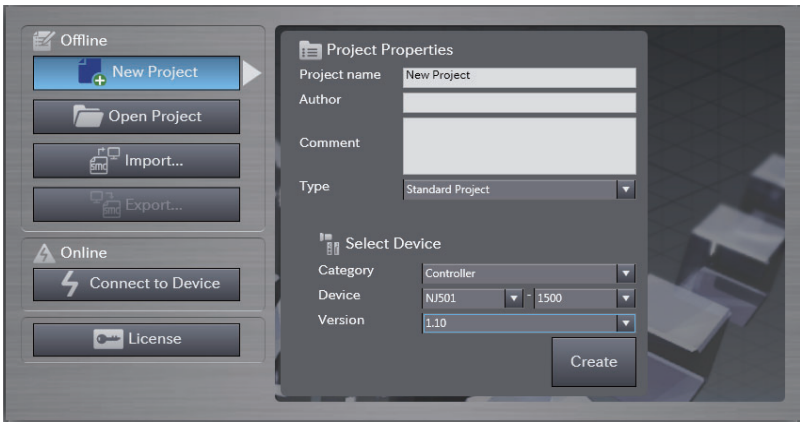
You can use Sysmac Libraries uploaded from a CPU Unit or an Industrial PC to your computer if they are not installed.

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

Version Information

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

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



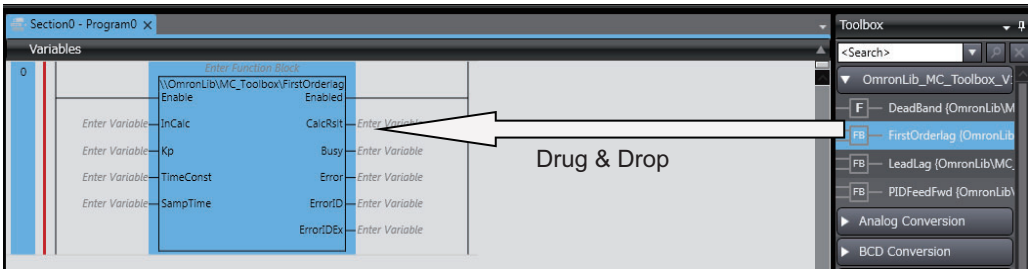
2 Connect the computer to the CPU Unit or the Industrial PC and place it online.

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

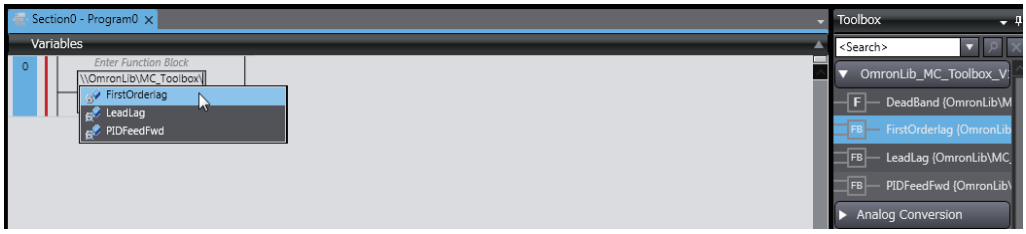
Now, when you select the Ladder Editor or ST Editor, the function blocks and functions included in the Sysmac Library used in the uploaded POUs appear in the Toolbox.

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

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



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



Precautions for Correct Use

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

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

Common Specifications of Function Blocks

Common Variables

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

Definition of Input Variables and Output Variables

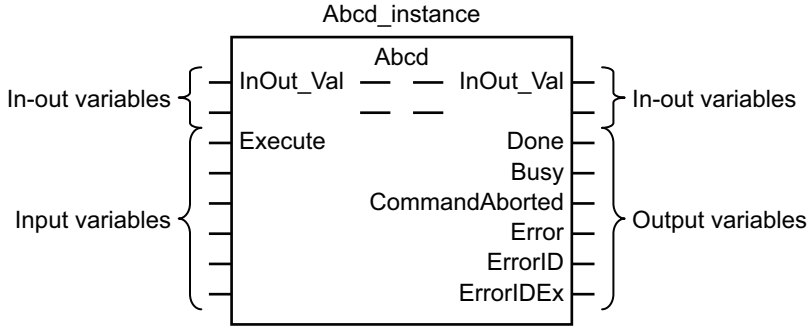
Common input variables and output variables used in functions and function blocks are as follows.

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

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

Execute-type Function Blocks

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

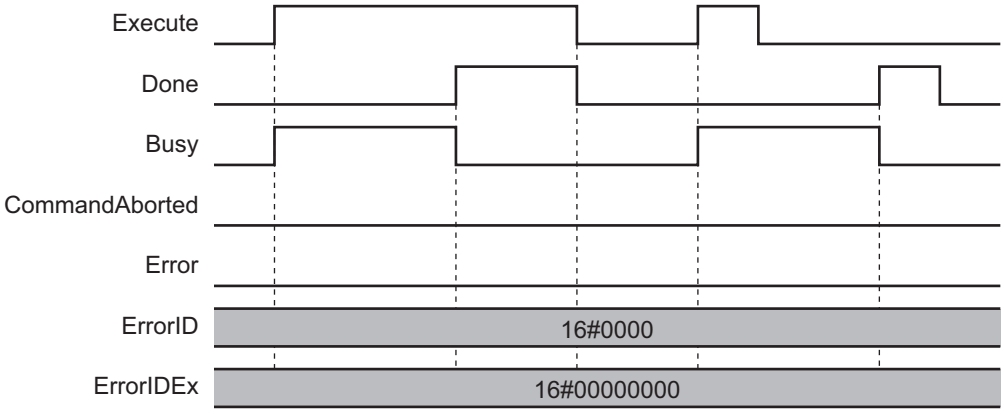


- If *Execute* is TRUE and *Done*, *CommandAborted*, or *Error* changes to TRUE, *Done*, *Command-Aborted*, and *Error* changes to FALSE when *Execute* is changed to FALSE.
- If *Execute* is FALSE and *Done*, *CommandAborted*, or *Error* changes to TRUE, *Done*,*Command-Aborted*, and *Error* changes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

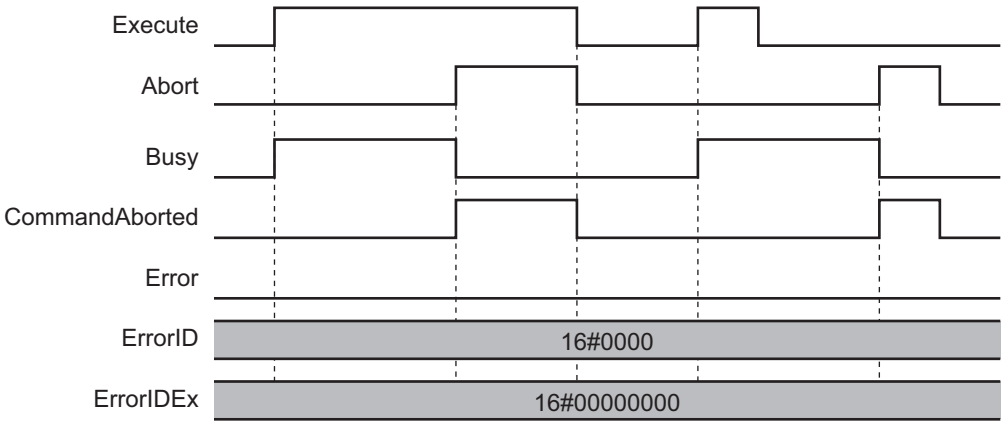
Timing Charts

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

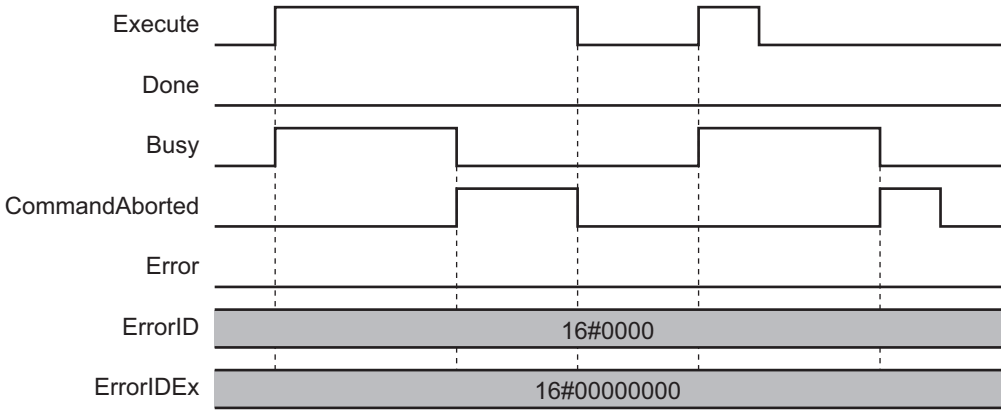
● Normal End



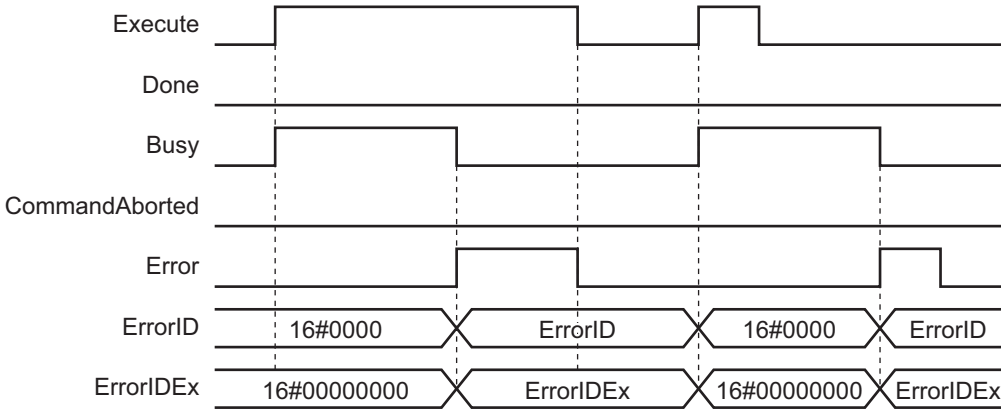
● Canceled Execution



Aborted Execution

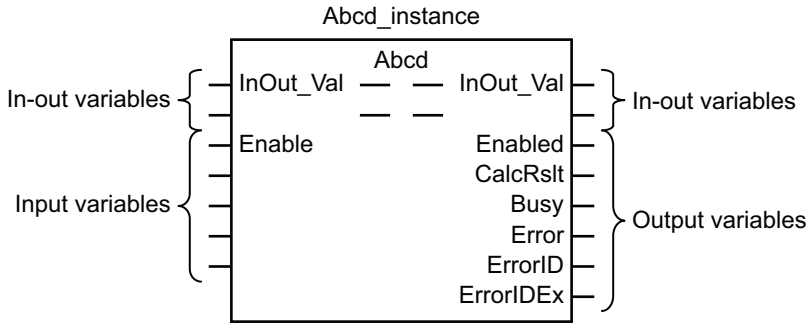


Errors



Enable-type Function Blocks

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

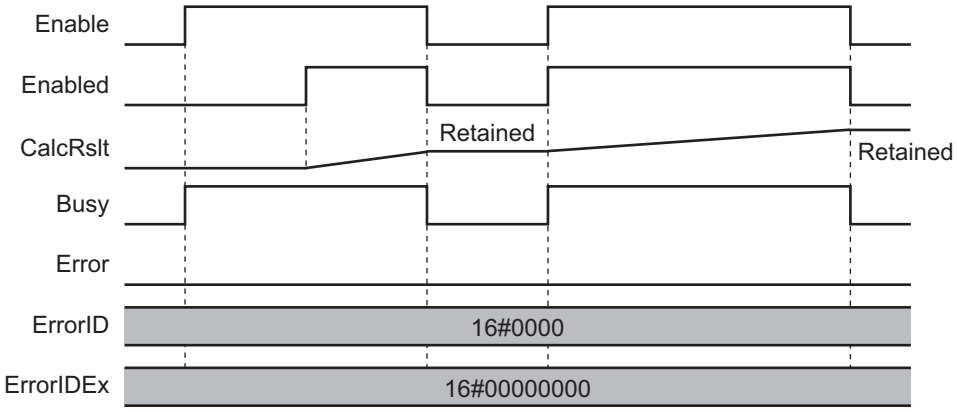


- If an error occurs, the relevant error code and expansion error code are set in *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code). The error codes are retained even after *Error* changes to FALSE, but *ErrorID* is set to 16#0000 and *ErrorIDEx* is set to 16#0000 0000 when *Enable* changes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

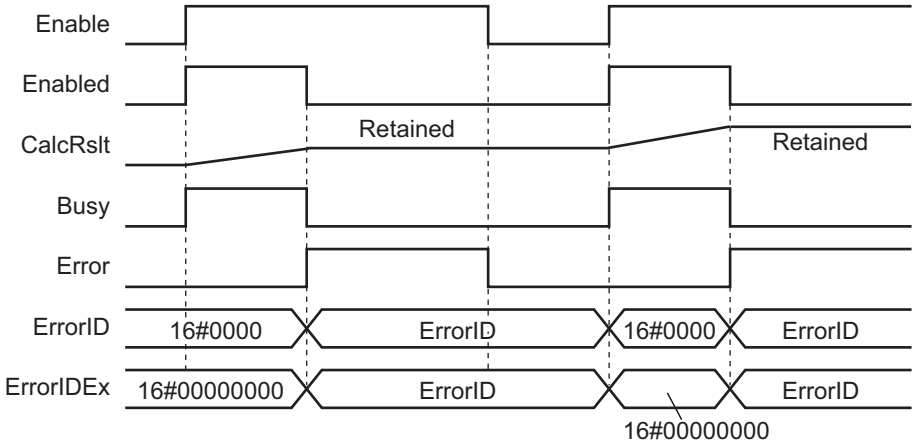
Timing Charts

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

● Normal End



Errors


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

Precautions

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

Nesting

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

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

Instruction Options

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

Re-execution of Function Blocks

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

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

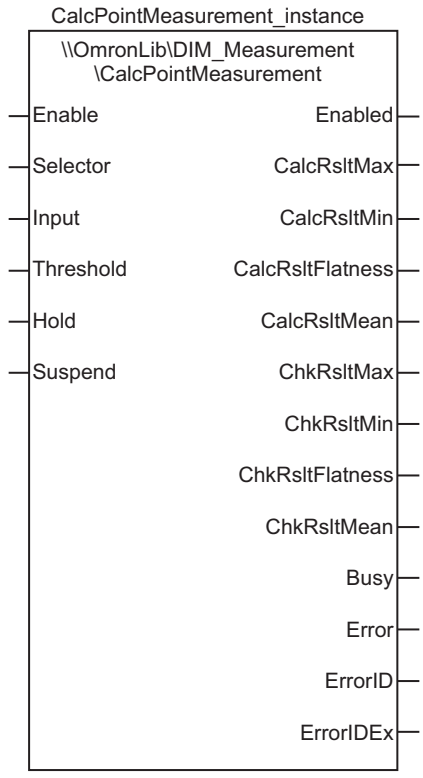
For details on re-execution, refer to the motion control user's manual.

Individual Specifications of Function Blocks

Function block name	Name	Page
CalcPointMeasurement	Point Measurement Calculation (Maximum Value, Minimum Value, Flatness and Mean Value)	P.32
PointMeasure_Deviation	Deviation Calculation	P.48
PointMeasure_Torsion	Torsion Calculation	P.63
PointMeasure_Curve	Curve Calculation	P.71
PointMeasure_Thickness	Thickness Calculation	P.79
ZW_ZeroResetControl	ZW Zero Reset Control	P.86
ZW_CmdControl	ZW Command Control	P.92
LineMeasure_Cartesian	Line Measurement with Cartesian Coordinate System (Surface Search/Tracer Control)	P.108
LineMeasure_Cartesian2	Line Measurement with Cartesian Coordinate System2 (Surface Search / Tracer Control)	P.151
LineMeasure_CreateShape2D_Master	Master 2D Shape Data Creation	P.176
LineMeasure_CreateShape2D_Master2	Master 2D Shape Data Creation 2	P.195
LineMeasure_CreateShape2D	2D Shape Data Creation	P.215
LineMeasure_CreateShape2D2	2D Shape Data Creation 2	P.226
Shape2D_Height	2D Shape Height Measurement	P.238
Shape2D_Height2	2D Shape Height Measurement 2	P.253
Shape2D_Edge	2D Shape Edge Position Measurement	P.267
Shape2D_Edge2	2D Shape Edge Position Measurement 2	P.277
Shape2D_InflectionPoint	2D Shape Inflection Point Measurement	P.288
Shape2D_InflectionPoint2	2D Shape Inflection Point Measurement 2	P.296
Shape2D_Angle	2D Shape Angle Measurement	P.304
Shape2D_Angle2	2D Shape Angle Measurement 2	P.312
Shape2D_Area	2D Shape Sectional Area Measurement	P.320
Shape2D_Area2	2D Shape Sectional Area Measurement 2	P.329
Shape2D_Compare	2D Shape Comparison Measurement	P.337
Shape2D_Compare2	2D Shape Comparison Measurement 2	P.346

CalcPointMeasurement

The CalcPointMeasurement function block utilizes the measurement data obtained from 1 to 16 measurement sensors, to perform the calculation of maximum value, minimum value, flatness, and mean value.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
CalcPoint Measurement	Point Measurement Calculation (Maximum Value, Minimum Value, Flatness and Mean Value)	FB		CalcPointMeasurement_Instance (Enable, Selector, Input, Threshold, Hold, Suspend, Enabled, CalcRsltMax, CalcRsltMin, CalcRsltFlatness, CalcRsltMean, ChkRsltMax, ChkRsltMin, ChkRsltFlatness, ChkRsltMean, Busy, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_PointMeasurement_V1_1.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00091
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Product name	Model numbers	Version
Device	Sensor Communications Unit	E3NW-ECT	Rev 1.03 or later
	Distributed Sensor Unit	E3NW-DS	-
	Contact-Type Smart Amplifier	E3NC-TA0	-
	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Enable	Execution	TRUE: Execute FALSE: Do not execute	BOOL	Depends on data type.	FALSE
Selector*1	Sensor Head Selection	TRUE: The sensor input value with the array element number set to TRUE is the target of calculation. FALSE: The sensor input value with the array element number set to FALSE is not the target of calculation. Specify one or more sensors.	ARRAY [0..15] OF BOOL	Depends on data type.	FALSE
Input	Input Value	The input value obtained from a measurement sensor. The measurement values of the sensors are input for each array element. For the unit, refer to the manual of the respective sensors.	ARRAY [0..15] OF DINT	Depends on data type.	0
Threshold*1	Threshold	Whether the threshold is exceeded is judged. Details on the data type: Refer to *Members of Structure sThresholdParams* on page 36.	OmronLib\DIM_Measurement\sThresholdParams	—	—
Hold*1	Output Value Hold	In the case of TRUE: • Holds the maximum value of *CalcRsltMax*. • Holds the minimum value of *CalcRsltMin*. • Outputs the difference between maximum and minimum values as flatness.	BOOL	Depends on data type.	FALSE
Suspend	Suspension	Set to TRUE when inputting measurement data on which you want to disable the calculation during execution.	BOOL	Depends on data type.	FALSE

*1. If the set values are used for calculation at the task period when FALSE changed to TRUE on *Enable* (Execution) of this function block, the values which changed while *Enable* was TRUE are not reflected in the calculation.

Output Variables

Variables	Name	Description	Data type	Valid range	Default
Enabled	Calculation in Progress	TRUE: Calculation in progress FALSE: Calculation stopped	BOOL	Depends on data type.	—
CalcRsltMax	Maximum Value Calculation Result	Stores the maximum value calculated, using the detection values of sensors specified in Sensor head selection.	DINT	Depends on data type.	—
CalcRsltMin	Minimum value Calculation Result	Stores the minimum value calculated, using the detection values of sensors specified in Sensor head selection.	DINT	Depends on data type.	—
CalcRsltFlatness	Flatness Calculation Result	Stores the flatness calculated, using the detection values of sensors specified in Sensor head selection.	DINT	Depends on data type.	—
CalcRsltMean	Mean Value Calculation Result	Stores the mean value calculated, using the detection values of sensors specified in Sensor head selection.	DINT	Depends on data type.	—
ChkRsltMax	Maximum Value Judgement Result	Turns to TRUE if the maximum value calculation result meets all the following criteria: • Not less than the threshold (Low). • Not more than the threshold (High).	BOOL	Depends on data type.	—
ChkRsltMin	Minimum Value Judgement Result	Turns to TRUE if the minimum value calculation result meets all the following criteria: • Not less than the threshold (Low). • Not more than the threshold (High).	BOOL	Depends on data type.	—
ChkRsltFlatness	Flatness Judgement Result	Turns to TRUE if the flatness calculation result meets all the following criteria: • Not less than the threshold (Low). • Not more than the threshold (High).	BOOL	Depends on data type.	—
ChkRsltMean	Mean Value Judgement Result	Turns to TRUE if the mean value calculation result meets all the following criteria: • Not less than the threshold (Low). • Not more than the threshold (High).	BOOL	Depends on data type.	—

Variables	Name	Description	Data type	Valid range	Default
Busy	Execution in progress	Turns to TRUE while the processing is in execution; turns to FALSE while the processing is not in execution.	BOOL	Depends on data type.	—
Error	Error	Outputs TRUE if an error occurs.	BOOL	Depends on data type.	—
ErrorID	Error Code	Outputs the error code if an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the extended error code if an error occurs.	DWORD	*1	—

*1. Refer to the *Troubleshooting* on page 42.

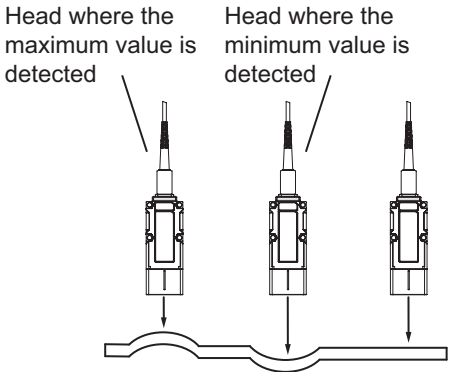
Members of Structure sThresholdParams

Member	Name	Data type	Valid range	Description
Max_High	Maximum Value Threshold (High)	DINT	Depends on data type.	The judgement output turns to TRUE when the calculation results are in between the threshold (High) and the threshold (Low), both of which are specified here. Specify the thresholds always as: Threshold (High) ≥ Threshold (Low).
Max_Low	Maximum Value Threshold (Low)	DINT	Depends on data type.	
Min_High	Minimum Value Threshold (High)	DINT	Depends on data type.	
Min_Low	Minimum Value Threshold (Low)	DINT	Depends on data type.	
Flatness_High	Flatness Threshold (High)	DINT	Depends on data type.	
Flatness_Low	Flatness Threshold (Low)	DINT	Depends on data type.	
Mean_High	Mean Value Threshold (High)	DINT	Depends on data type.	
Mean_Low	Mean Value Threshold (Low)	DINT	Depends on data type.	

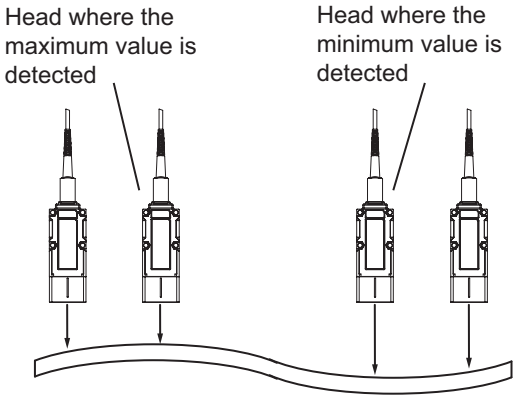
Function

The CalcPointMeasurement function block utilizes the measurement data obtained from any one of the following 1 to 16 measurement sensors, to perform the calculation of maximum value, minimum value, flatness, and mean value.

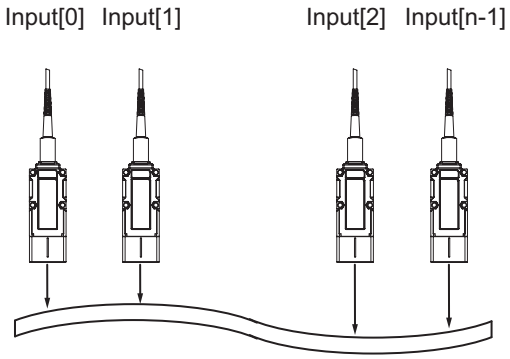
- Sensor amplifier connected to the Sensor Communications Unit (E3NW-ECT)
- Confocal Fiber Type Displacement Sensor (ZW-8000/7000/5000)



Flatness = (Maximum value – Minimum value)



Mean value = (Measurement value of X1 + Measurement value of X2 +... Measurement value of Xn)/ n



(a) Set up the measurement values of sensors in each array element of *Input* (input value) to take in the measurement values of 16 sensors into this function block. The following is an example where device variables are set so as to store the measurement values of sensors in the array elements of Input.

```
Input[0]  := E001_Measurement_Value;
Input[1]  := E002_Measurement_Value;
Input[2]  := E003_Measurement_Value;
•
•
Input[15] := E0016_Measurement_Value;
```

(b) When *Enable* (Execution) is set to TRUE:

- From the inputs with the array element number set to TRUE in *Selector* (Sensor Head Selection), the maximum value is set for *CalcRsltMax*.
- From the inputs with the array element number set to TRUE in *Selector* (Sensor Head Selection), the minimum value is set for *CalcRsltMin*.
- From the inputs with the array element number set to TRUE in *Selector* (Sensor Head Selection), the maximum and minimum values are extracted and their difference (i.e. the minimum value substracted from the maximum value) is set for *CalcRsltFlatness* as the flatness.
- From the inputs with the array element number set to TRUE in *Selector*, their mean value is set for *CalcRsltMean*.

While *Enable* is TRUE, the calculation continues. As *Enable* turns to FALSE, the calculation is terminated, but *CalcRsltMax*, *CalcRsltMin*, *CalcRsltFlatness*, and *CalcRsltMean* hold their values, which will be reset next time *Enable* changes to TRUE.

(c) When the value of *CalcRsltMax* is not less than the threshold (Low) and not more than the threshold (High), *ChkRsltMax* turns to TRUE. Even if *Enable* turns to FALSE, the value of *ChkRsltMax* is held and will be reset next time *Enable* changes to TRUE. *CalcRsltMin*, *CalcRsltFlatness*, and *CalcRsltMean* also show the similar behavior.

(d) When *Hold* (Output Value Hold) is set to TRUE:

- The maximum value of *CalcRsltMax* is held.
- The minimum value of *CalcRsltMin* is held.
- The difference between the above-mentioned maximum and minimum values is output to *CalcRsltFlatness* as the flatness.

(e) Setting *Suspend* (Suspension) to TRUE allows the calculation to be suspended while the execution is in progress. This function is useful to remove the calculation results in the task periods where process data communications fail and the measurement data is not updated correctly. During suspension, *Enabled* (Calculation in Progress) turns to FALSE, where the number of data points used for mean value calculation is not updated. As you set *Suspend* to FALSE, calculation resumes.

(f) If an error occurs, *Error* turns to TRUE and the processing of function blocks is interrupted. In addition, the error code is output to *ErrorID* and *ErrorIDEx*.

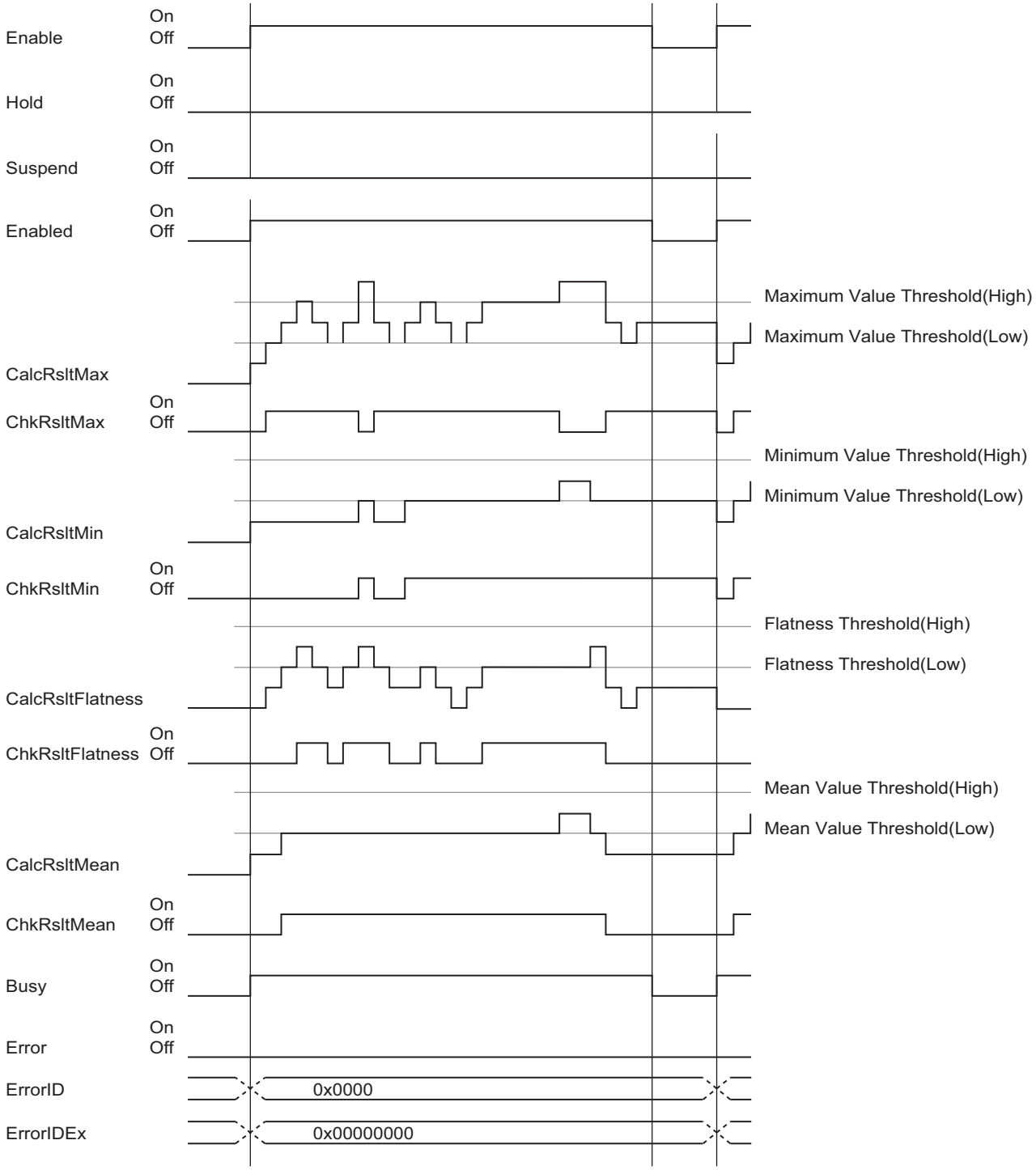
For details on the error codes, refer to *Troubleshooting* on page 42.

Timing Charts

The timing charts are shown below.

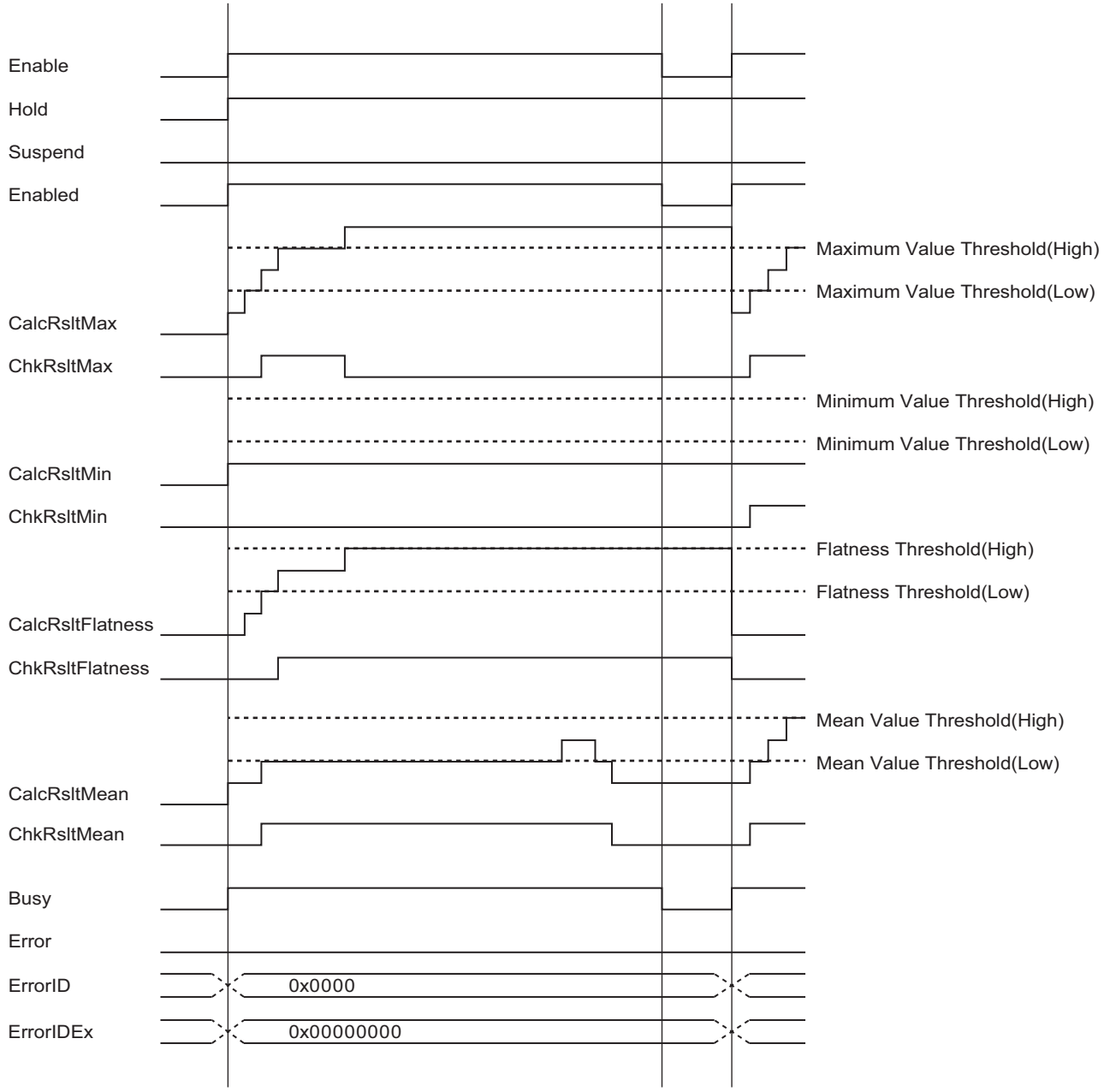
● Timing Chart in a Normal State (HOLD = FALSE)

At the same time when Enable (Execution) changes to TRUE, *Busy* (Execution in Progress) turns to TRUE. If the calculation results *CalcRsltMax* to *CalcRsltMean* do not exceed the thresholds, the judgement results from *ChkRsltMax* to *ChkRsltMean* turn to TRUE. As you set *Enable* to FALSE, *Busy* and *Enabled* turn to FALSE. The judgement results from *ChkRsltMax* to *ChkRsltMean* hold their values, which will be reset next time *Enable* changes to TRUE.



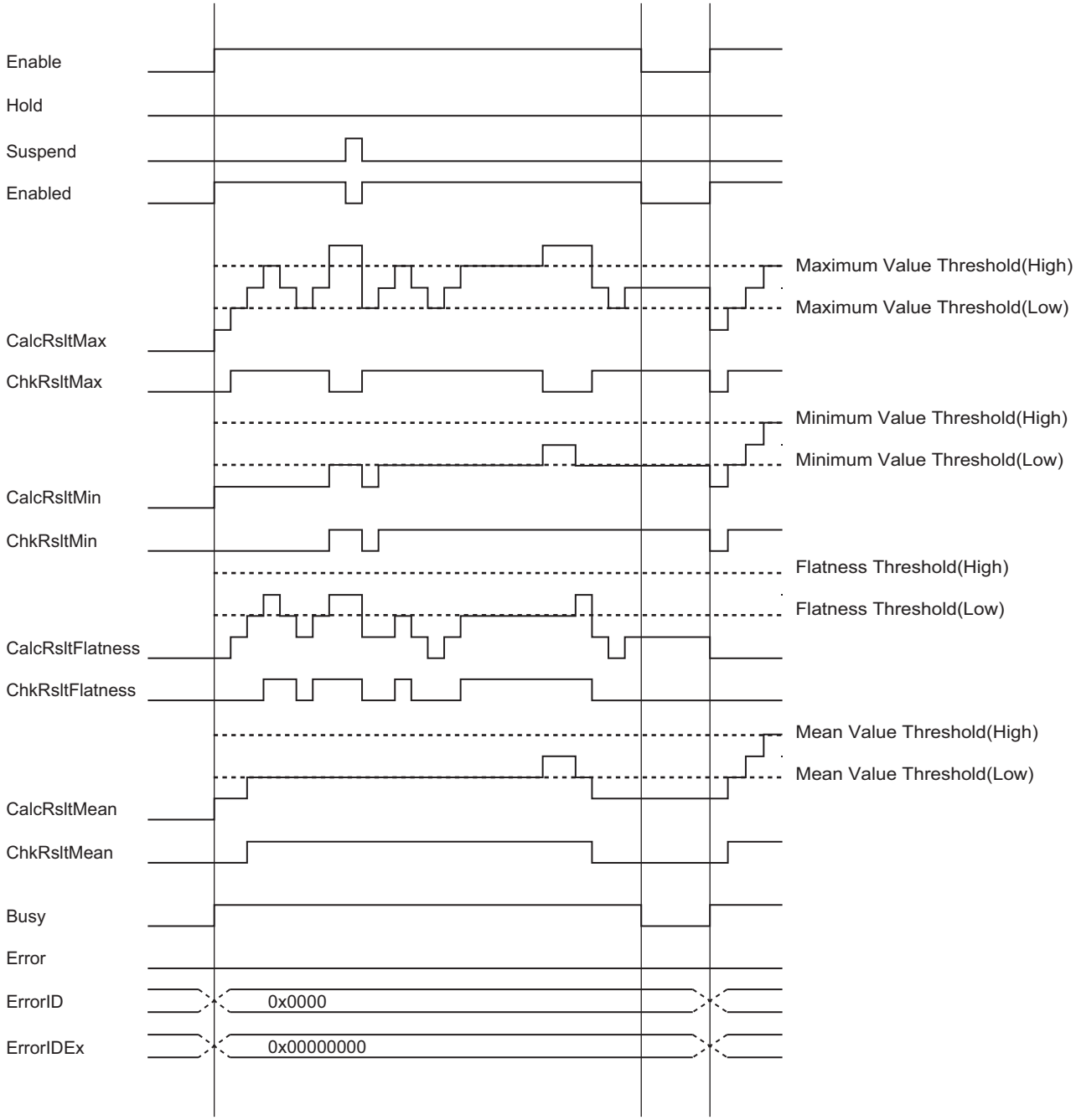
● Timing Chart in a Normal State (HOLD = TRUE)

Execution with *HOLD* set to TRUE results in the maximum value of *CalcRsltMax* and the minimum value of *CalcRsltMin* being held while *Enable* is TRUE. The flatness is output as: Maximum value of *CalcRsltMax* – Minimum value of *CalcRsltMin*.



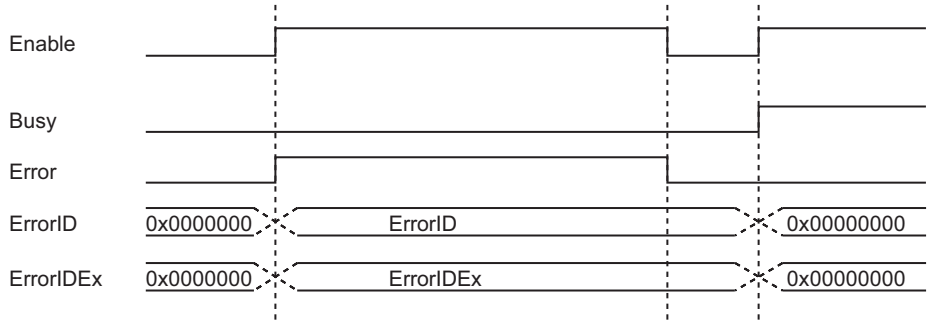
● Timing Chart in a Normal State (*Suspend* = TRUE)

With *Suspend* set to TRUE, the calculation processing is suspended while *Enable* is TRUE. While Suspend is TRUE, *Enabled* turns to FALSE, where *CalcRsltMax* to *CalcRsltMean* and *ChkRsltMax* to *ChkRsltMean* hold the old values before *Suspend* turned to TRUE.



● Timing Chart When an Error Occurs

If an error occurs while this function block is in execution, *Error* (Error) turns to TRUE and the values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code). As you set *Enable* to FALSE, Error turns to FALSE as well, but *ErrorID* and *ErrorIDEx* hold their values until the next execution.



Precautions for Use

- For input values to this function block, consistently use either sensors connected to the E3NW or sensors of the ZW-8000/7000/5000. The combined use results in a failure to perform the correct operation because the unit is different from each other.
- Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.
- For the ZW-8000/7000/5000 series, confirm that the following pre-conditions are met before executing this function block.
  a) The ZW_CmdControl function block is unexecuted.
  b) *TASKSTAT_TX* which is the status of the target task for measurement slave is TRUE.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3C9E	0x1	The upper threshold is smaller than the lower threshold.	Specify the thresholds so that the upper threshold will be larger than the lower threshold.
	0x2	The number of selected sensor heads is less than the minimum requirement.	For head selection, specify one or more heads undertaking the calculation.

Sample Programming

Program Description

For one second after the measurement start command, this program uses the measurement values of 16 types of sensors connected via EtherCAT as inputs to perform calculation by CalcPointMeasurement function block.

Preconditions

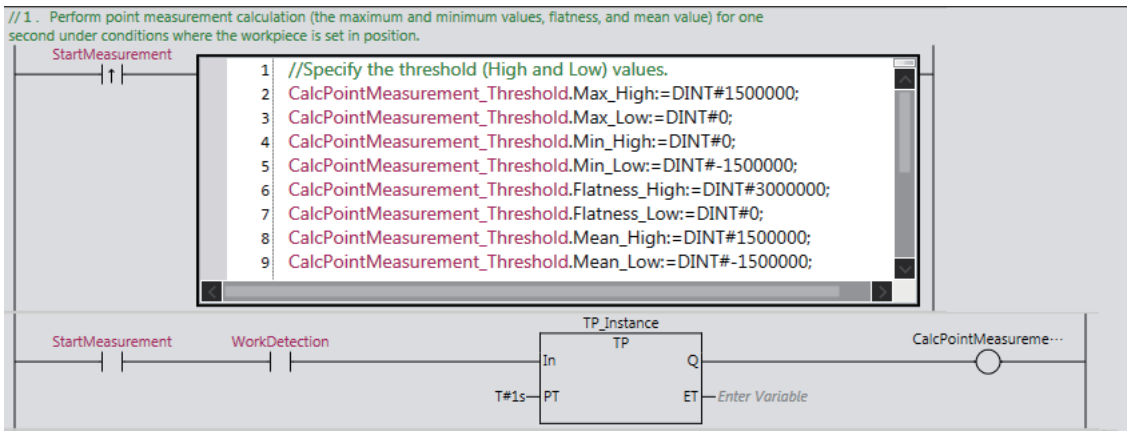
- Create device variables for the measurement values of the displacement sensor with which you want to perform measurement and use external references to the variables in the program. Refer to the *Sysmac Studio Version 1 Operation Manual* (Cat. No. W504) for details on how to create device variables.
- If E9NC-TA sensors are used to input values, change the PDO mapping settings for the E3NW-ECT so that the sensor detection amount is 4 bytes.
  Example: If E9NC-TA sensors are connected to an E3NW-ECT with unit number 01
  →Change the setting of 273rd transmit PDO Mapping (No_01 Detection Level IN 1) from *Input* to *Not selected*.
  →Change the setting of 423rd transmit PDO Mapping (No_01 Detection Level (4 bytes) from *Not selected* to *Input*.
- If ZW-8000/7000/5000 sensors are used to input values, set the measurement item for each sensor to the same item.
- Set each sensor to zero beforehand.

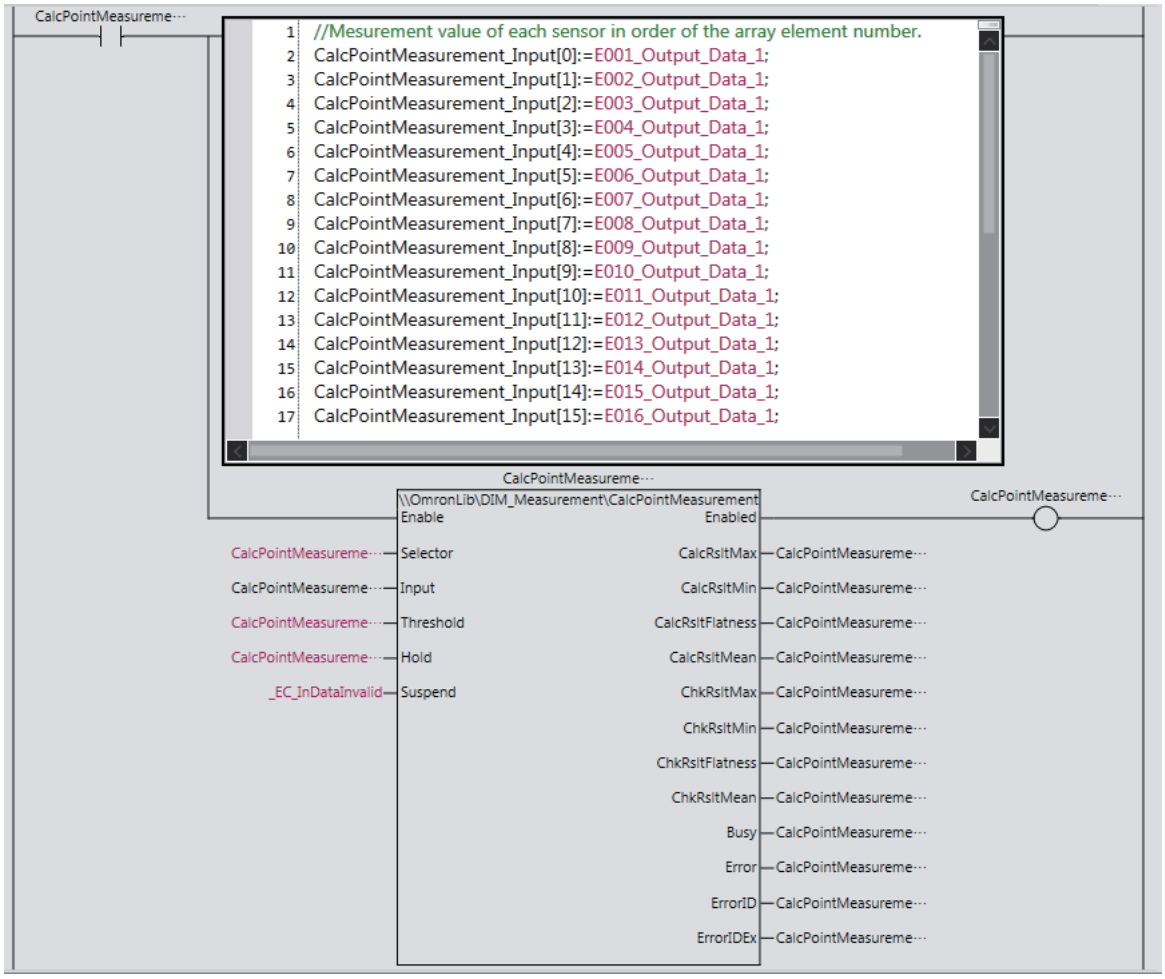
Main Variables

Name	Meaning	Data type	Default	Description
StartMeasurement	Start Point Measurement Calculation	BOOL	FALSE	Starts point measurement calculation when the variable changes to TRUE.
WorkDetection	Workpiece in Position Status	BOOL	FALSE	TRUE: The workpiece for measurement is set in position. FALSE: The workpiece for measurement is not set in position.
CalcPointMeasurement_ Selector[]	Sensor Head Selection	ARRAY[0..15] OF BOOL	FALSE	Selects the input value of the sensor with the array element number that changed this variable to TRUE. Assign this variable to the *Selector* input variable of CalcPointMeasurement.
CalcPointMeasurement_ input[]	Measurement Sensor Input Value	ARRAY[0..15] OF DINT	---	Assigns the OUT1 data in the output area I/O port of the sensor heads 1 to 16 in order of the array element number. Assign this variable to the *Input* input variable of CalcPointMeasurement.

Name	Meaning	Data type	Default	Description
CalcPointMeasurement_Threshold	Threshold Setting	OmronLib\DIM_Measurement\sThresholdParams	---	Sets the threshold range. Assign this variable to the *Threshold* input variable of CalcPointMeasurement.
CalcPointMeasurement_Hold	Output Value Hold	BOOL	FALSE	TRUE: • Holds the maximum value in *CalcRsltMax*. • Holds the minimum value in *CalcRsltMin*. Assign this variable to the *Hold* input variable of CalcPointMeasurement.
_EC_InDataInvalid	Input Data Disabled	BOOL	---	A system-defined variable for EtherCAT communications. TRUE when the process data communications executed in the primary periodic task is not normal and the input value is not valid. Assign this variable to the *Suspend* input variable of CalcPointMeasurement.
CalcPointMeasurement_CalcRsltMax	Maximum Value Calculation Result	DINT	---	Assign this variable to the *CalcRsltMax* output variable of CalcPointMeasurement.
CalcPointMeasurement_CalcRsltMin	Minimum Value Calculation Result	DINT	---	Assign this variable to the *CalcRsltMin* output variable of CalcPointMeasurement.
CalcPointMeasurement_CalcRsltFlatness	Flatness Calculation Result	DINT	---	Assign this variable to the *CalcRsltFlatness* output variable of CalcPointMeasurement.
CalcPointMeasurement_CalcRsltMean	Mean Value Calculation Result	DINT	---	Assign this variable to the *CalcRsltMean* output variable of CalcPointMeasurement.
CalcPointMeasurement_ChkRsltMax	Maximum Value Judgment Result	BOOL	---	Assign this variable to the *ChkRsltMax* output variable of CalcPointMeasurement.
CalcPointMeasurement_ChkRsltMin	Minimum Value Judgment Result	BOOL	---	Assign this variable to the *ChkRsltMin* output variable of CalcPointMeasurement.
CalcPointMeasurement_ChkRsltFlatness	Flatness Judgment Result	BOOL	---	Assign this variable to the *ChkRsltFlatness* output variable of CalcPointMeasurement.
CalcPointMeasurement_ChkRsltMean	Mean Value Judgment Result	BOOL	---	Assign this variable to the *ChkRsltMean* output variable of CalcPointMeasurement.

Ladder Diagram





● Code of Inline ST (Zeroth Line of Ladder Diagram):

```
// Specify the threshold (High and Low) values.
CalcPointMeasurement_Threshold.Max_High:=DINT#1500000;
CalcPointMeasurement_Threshold.Max_Low:=DINT#0;
CalcPointMeasurement_Threshold.Min_High:=DINT#0;
CalcPointMeasurement_Threshold.Min_Low:=DINT#-1500000;
CalcPointMeasurement_Threshold.Flatness_High:=DINT#3000000;
CalcPointMeasurement_Threshold.Flatness_Low:=DINT#0;
CalcPointMeasurement_Threshold.Mean_High:=DINT#1500000;
CalcPointMeasurement_Threshold.Mean_Low:=DINT#-1500000;
```

● Code of Inline ST (Second Line of Ladder Diagram):

```
// Input the measurement value of each sensor in order of the array element number.
CalcPointMeasurement_Input[0]:=E001_Output_Data_1;
CalcPointMeasurement_Input[1]:=E002_Output_Data_1;
CalcPointMeasurement_Input[2]:=E003_Output_Data_1;
CalcPointMeasurement_Input[3]:=E004_Output_Data_1;
CalcPointMeasurement_Input[4]:=E005_Output_Data_1;
CalcPointMeasurement_Input[5]:=E006_Output_Data_1;
CalcPointMeasurement_Input[6]:=E007_Output_Data_1;
CalcPointMeasurement_Input[7]:=E008_Output_Data_1;
CalcPointMeasurement_Input[8]:=E009_Output_Data_1;
CalcPointMeasurement_Input[9]:=E010_Output_Data_1;
CalcPointMeasurement_Input[10]:=E011_Output_Data_1;
CalcPointMeasurement_Input[11]:=E012_Output_Data_1;
CalcPointMeasurement_Input[12]:=E013_Output_Data_1;
CalcPointMeasurement_Input[13]:=E014_Output_Data_1;
CalcPointMeasurement_Input[14]:=E015_Output_Data_1;
CalcPointMeasurement_Input[15]:=E016_Output_Data_1;
```

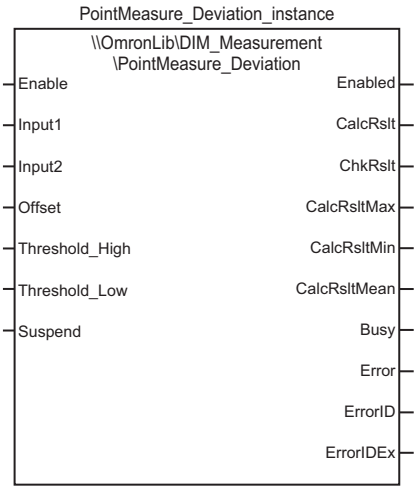
ST

```
// 1. Perform point measurement calculation (the maximum and minimum values, flatness, and mean value) for one second under condition where the workpiece is set in position.
R_TRIG_instance(Clk:=StartMeasurement, Q=>Start_Result);
IF (Start_Result=TRUE) THEN
   // Specify the threshold (High and Low) values.
   CalcPointMeasurement_Threshold.Max_High:=DINT#1500000;
   CalcPointMeasurement_Threshold.Max_Low:=DINT#0;
   CalcPointMeasurement_Threshold.Min_High:=DINT#0;
   CalcPointMeasurement_Threshold.Min_Low:=DINT#-1500000;
   CalcPointMeasurement_Threshold.Flatness_High:=DINT#3000000;
   CalcPointMeasurement_Threshold.Flatness_Low:=DINT#0;
   CalcPointMeasurement_Threshold.Mean_High:=DINT#1500000;
   CalcPointMeasurement_Threshold.Mean_Low:=DINT#-1500000;
END_IF;

IF (StartMeasurement=TRUE) AND (WorkDetection=TRUE) THEN
   TP_Instance(In:=TRUE, PT:=T#1s, Q=>CalcPointMeasurement_Enable);
ELSE
   TP_Instance(In:=FALSE, Q=>CalcPointMeasurement_Enable);
END_IF;

IF (CalcPointMeasurement_Enable=TRUE) THEN
   // Input the measurement value of each sensor in order of the array element number.
   CalcPointMeasurement_Input[0]:=E001_Output_Data_1;
   CalcPointMeasurement_Input[1]:=E002_Output_Data_1;
   CalcPointMeasurement_Input[2]:=E003_Output_Data_1;
   CalcPointMeasurement_Input[3]:=E004_Output_Data_1;
   CalcPointMeasurement_Input[4]:=E005_Output_Data_1;
   CalcPointMeasurement_Input[5]:=E006_Output_Data_1;
```

```
   CalcPointMeasurement_Input[6]:=E007_Output_Data_1;
   CalcPointMeasurement_Input[7]:=E008_Output_Data_1;
   CalcPointMeasurement_Input[8]:=E009_Output_Data_1;
   CalcPointMeasurement_Input[9]:=E010_Output_Data_1;
   CalcPointMeasurement_Input[10]:=E011_Output_Data_1;
   CalcPointMeasurement_Input[11]:=E012_Output_Data_1;
   CalcPointMeasurement_Input[12]:=E013_Output_Data_1;
   CalcPointMeasurement_Input[13]:=E014_Output_Data_1;
   CalcPointMeasurement_Input[14]:=E015_Output_Data_1;
   CalcPointMeasurement_Input[15]:=E016_Output_Data_1;
END_IF;

// Point measurement calculation function block
CalcPointMeasurement_Instance
(
Enable:=CalcPointMeasurement_Enable,
Selector:=CalcPointMeasurement_Selector,
Input:=CalcPointMeasurement_input,
Threshold:=CalcPointMeasurement_Threshold,
Hold:=CalcPointMeasurement_Hold,
Suspend:=_EC_InDataInvalid,
Enabled=>CalcPointMeasurement_Enabled,
CalcRsltMax=>CalcPointMeasurement_CalcRsltMax,
CalcRsltMin=>CalcPointMeasurement_CalcRsltMin,
CalcRsltFlatness=>CalcPointMeasurement_CalcRsltFlatness,
CalcRsltMean=>CalcPointMeasurement_CalcRsltMean,
ChkRsltMax=>CalcPointMeasurement_ChkRsltMax,
ChkRsltMin=>CalcPointMeasurement_ChkRsltMin,
ChkRsltFlatness=>CalcPointMeasurement_ChkRsltFlatness,
ChkRsltMean=>CalcPointMeasurement_ChkRsltMean,
Busy=>CalcPointMeasurement_Busy,
Error=>CalcPointMeasurement_Error,
ErrorID=>CalcPointMeasurement_ErrorID,
ErrorIDEx=>CalcPointMeasurement_ErrorIDEx
);
```

PointMeasure_Deviation

The PointMeasure_Deviation function block outputs the difference of measurement values (*Input1* and *Input2*) obtained from two measurement sensors on the calculation result.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
PointMeasure_ Deviation	Deviation Calculation	FB		PointMeasure_Deviation_Instance (Enable, Input1, Input2, Offset, Threshold_High, Threshold_Low, Suspend, Enabled, CalcRslt, ChkRslt, CalcRsltMax, CalcRsltMin, CalcRsltMean, Busy, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_PointMeasurement_V1_1.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00092
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Product name	Model numbers	Version
Device	Sensor Communications Unit	E3NW-ECT	Rev 1.03 or later
	Distributed Sensor Unit	E3NW-DS	-
	Contact-Type Smart Amplifier	E3NC-TA0	-
	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Enable	Execution	TRUE: Execute FALSE: Do not execute	BOOL	Depends on data type.	FALSE
Input1	Sensor Head 1 Input Value	The input value of the sensor head 1 is set.	DINT	Depends on data type.	0
Input2	Sensor Head 2 Input Value	The input value of the sensor head 2 is set.	DINT	Depends on data type.	0
Offset	Offset	The set value is added upon the calculation of *CalcRslt* (Calculation Result).	DINT	Depends on data type.	0
Threshold_High*1	Threshold (High)	Outputs the judgement of whether the threshold is exceeded. Specify the thresholds always as: Threshold (High) ≥ Threshold (Low).	DINT	Depends on data type.	0
Threshold_Low*1	Threshold (Low)		DINT		0
Suspend	Suspension	Set to TRUE when inputting measurement data on which you want to disable the calculation during execution.	BOOL	Depends on data type.	FALSE

*1. If the set values are used for calculation at the task period when FALSE changed to TRUE on *Enable* (Execution) of this function block, the values which changed while *Enable* was TRUE are not reflected in the calculation.

Output Variables

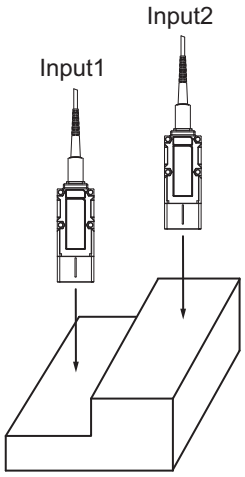
Variable	Name	Description	Data type	Valid range	Default
Enabled	Calculation in Progress	TRUE: Calculation in progress FALSE: Calculation stopped	BOOL	Depends on data type.	—
CalcRslt	Calculation Result	Stores the difference of sensor detection values to the calculation results.	DINT	Depends on data type.	—
ChkRslt	Judgement Output	Turns to TRUE if the calculation results meet all the following criteria: • Not less than the threshold (Low). • Not more than the threshold (High).	BOOL	Depends on data type.	—
CalcRsltMax	Calculation Result Maximum Value	Outputs the maximum value of *CalcRslt* under calculation.	DINT	Depends on data type.	—
CalcRsltMin	Calculation Result Minimum Value	Outputs the minimum value of *CalcRslt* under calculation.	DINT	Depends on data type.	—
CalcRsltMean	Calculation Result Mean Value	Outputs the mean value of *CalcRslt* under calculation.	DINT	Depends on data type.	—
Busy	Execution in Progress	Turns to TRUE while the processing is in execution; turns to FALSE while the processing is not in execution.	BOOL	Depends on data type.	—
Error	Error	Outputs TRUE if an error occurs.	BOOL	Depends on data type.	—
ErrorID	Error Code	Outputs the error code if an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the extended error code if an error occurs.	DWORD	*1	—

*1. Refer to the *Troubleshooting* on page 54.

Function

The PointMeasure_Deviation function block outputs the difference of measurement values (*Input1* and *Input2*) obtained from two measurement sensors, either type of the following sensors, on the calculation result.

- Sensor amplifier connected to the Sensor Communications Unit (E3NW-ECT)
- Confocal Fiber Displacement Sensor (ZW-8000/7000/5000)



(a) As you set *Enable* (Execution) to TRUE, the deviation is calculated from *Input1* and *Input2* to be set for *CalcRslt* (Calculation Result). The arithmetic expression is as follows:

(Input2 - Input1) + Offset

When the value of *CalcRslt* is not less than the threshold (Low) and not more than the threshold (High), *ChkRslt* (Judgement Output) turns to TRUE. While *Enable* is TRUE, the calculation continues for each task period. As *Enable* turns to FALSE, the calculation is terminated, but *CalcRslt* and *ChkRslt* hold their values, which will be reset next time *Enable* changes to TRUE.

(b) Under calculation, *CalcRsltMax* (Calculation Result Maximum Value), *CalcRsltMin* (Calculation Result Minimum Value), and *CalcRsltMean* (Calculation Result Mean Value) are output. While *Enable* is TRUE, the calculation continues. As *Enable* turns to FALSE, the calculation is terminated, but *CalcRsltMax*, *CalcRsltMin*, and *CalcRsltMean* hold their values, which will be reset next time *Enable* changes to TRUE.

(c) The mean value is calculated from values that are input while *Enable* is TRUE. However, if the maximum number of data points (number of calls of a function block while *Enable* is TRUE) is 65535 and then this maximum number is exceeded, the mean value is not updated.

(d) Setting *Suspend* (Suspension) to TRUE allows the calculation to be suspended while the execution is in progress. This function is useful to remove the calculation results in the task periods where process data communications fail and the measurement data is not updated correctly. During suspension, *Enabled* (Calculation in Progress) turns to FALSE, where the number of data points used for mean value calculation is not updated. As you set *Suspend* to FALSE, calculation resumes.

(e) If an error occurs, *Error* turns to TRUE, where the processing of function blocks is interrupted. In addition, the error code is output to *ErrorID* and *ErrorIDEx*.

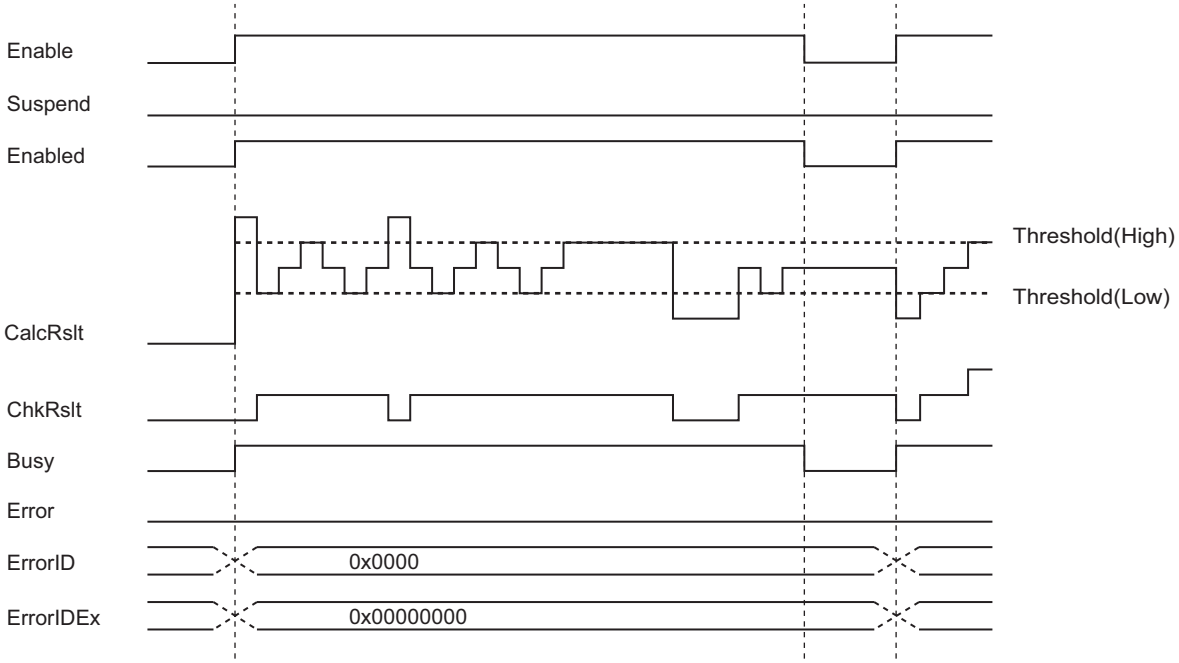
For details on the error codes, refer to *Troubleshooting* on page 54.

Timing Charts

The timing charts are shown below.

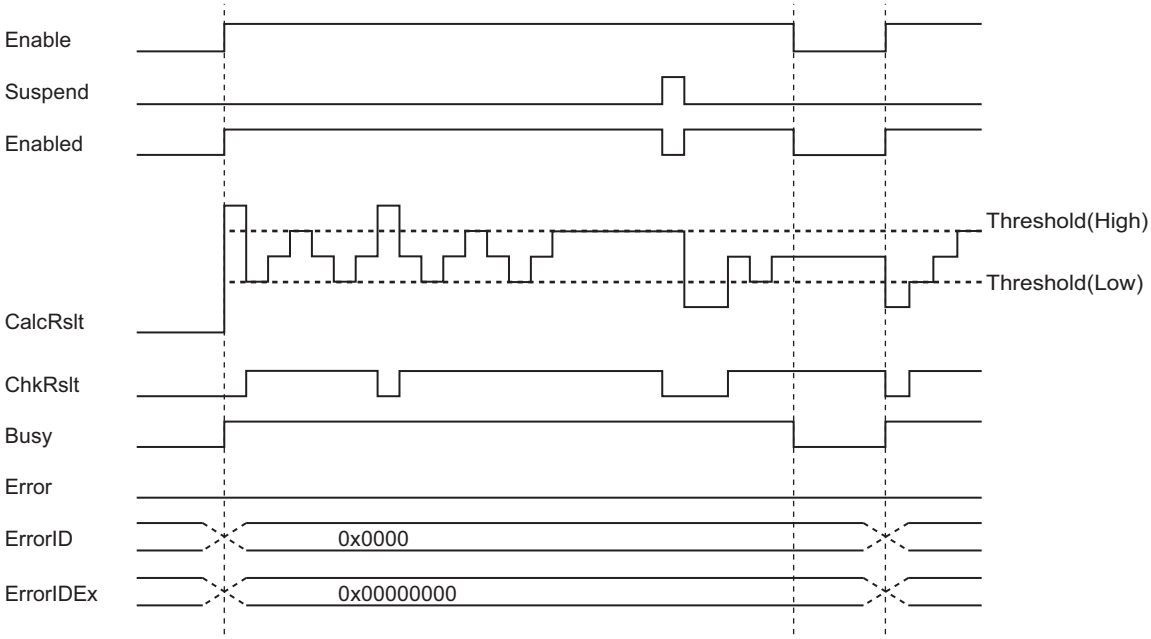
● In a Normal State

At the same time when Enable (Execution) changes to TRUE, *Busy* (Execution in Progress) and *Enabled* (Calculation result valid) turn to TRUE. If the calculation result exceeds the threshold, *ChkRslt* (Judgement Output) turns to FALSE. As you set *Enable* to FALSE, *Enabled* and *Busy* turn to FALSE, but *ChkRslt* (Judgment Output) holds its value, which will be reset next time *Enable* changes to TRUE.



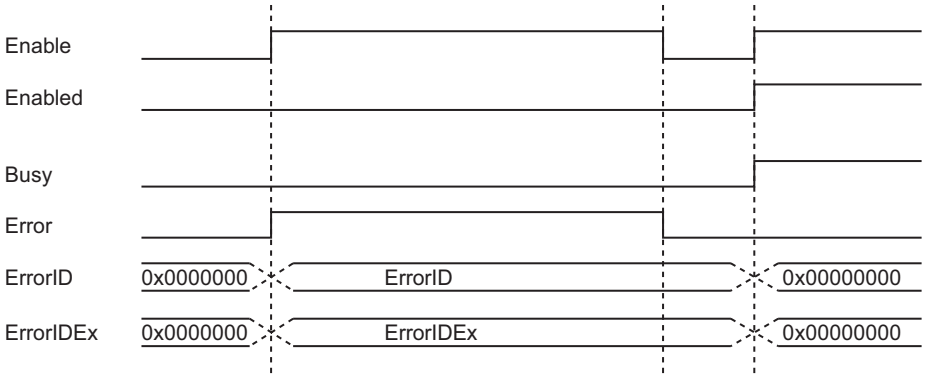
● In a Normal State (*Suspend* = TRUE)

With *Suspend* set to TRUE, the calculation processing is suspended while *Enable* (Execution) is TRUE. While *Suspend* is TRUE, *Enabled* turns to FALSE.



● When an Error Occurs

If an error occurs while this function block is in execution, *Error* turns to TRUE and the error code is output to *ErrorID*" and *ErrorIDEx*. As you set *Enable* to FALSE, *Error* turns to FALSE as well, but *ErrorID* and *ErrorIDEx* hold their values until the next execution.



Precautions for Use

- For input to this function block, consistently use either sensors connected to the E3NW or sensors of the ZW-8000/7000/5000. The combined use results in a failure to perform the correct operation because the unit is different from each other.
- Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.
- For the ZW-8000/7000/5000 series, confirm that the following pre-conditions are met before executing this function block.
  a) The ZW_CmdControl function block is unexecuted.
  b) *TASKSTAT_TX* which is the status of the target task for measurement slave is TRUE.

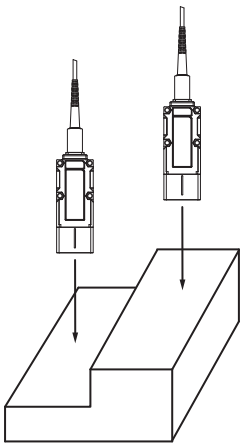
Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3C9F	0x1	The threshold (High) is smaller than the threshold (Low).	Specify the thresholds so that the threshold (High) will be larger than the threshold (Low).

Sample Programming

Program Description

This program uses two ZW-8000/7000/5000 displacement sensors to perform deviation calculation. The table below shows the processing flow.



No.	Outline	Description
1.	Bank change and zero reset	Under condition where the master workpiece for measurement with a deviation of 1 mm is set, change the ZW-8000/7000/5000 to the specified bank number and executes a zero reset.
2.	Deviation measurement	Under condition where the workpiece for measurement is set, measure its deviation for one second.

Preconditions

- Set the measurement item for each sensor to height.
- Select a combination of the workpiece and sensors that provides measurement values in the measurement range of the sensors.

Main Variables

Name	Meaning	Data type	Default	Description
Ready	Start Bank Change	BOOL	FALSE	Starts bank change when the variable changes to TRUE.
E001_FLG	Command Done in Response area I/O port on sensor head 1	BOOL	---	Command Done in the response area I/O port of the displacement sensor on the sensor head 1. Assign this variable to the *PDOFlg* input variable of ZW_CmdControl.

Name	Meaning	Data type	Default	Description
E001_READY	Signal Input Ready Status in Response area I/O port on sensor head 1	BOOL	---	Signal Input Ready Status in the response area I/O port of the displacement sensor on the sensor head 1. Assign this variable to the *PDOReady* input variable of ZW_CmdControl.
E001_Response	Command Echo Back in Response area I/O port on sensor head 1	BOOL	---	Command Echo Back in the response area I/O port of the displacement sensor on the sensor head 1. Assign this variable to the *PDOResponse* input variable of ZW_CmdControl.
E001_Response_Code	Response Code in Response area I/O port on sensor head 1	DWORD	---	Response Code in the response area I/O port of the displacement sensor on the sensor head 1. Assign this variable to the *PDOResponseCode* input variable of ZW_CmdControl.
E001_Response_Data1	Response Data in Response area I/O port on sensor head 1	DINT	---	Response Data in the response area I/O port of the displacement sensor on the sensor head 1. Assign this variable to the *PDOResponseData* input variable of ZW_CmdControl.
E001_EXE	Execute Command in Command area I/O port on sensor head 1	BOOL	FALSE	Execute Command in the command area I/O port of the displacement sensor on the sensor head 1. Assign this variable to the *PDOExecute* output variable of ZW_CmdControl.
E001_Command	Command Code in Command area I/O port on sensor head 1	DWORD	16#0	Command Code in the command area I/O port of the displacement sensor on the sensor head 1. Assign this variable to the *PDOCmdCode* output variable of ZW_CmdControl.
E001_Command_Parameter1	Command Parameter 1 in Command area I/O port on sensor head 1	UINT	UINT#0	Command Parameter 1 in the command area I/O port of the displacement sensor on the sensor head 1. Assign this variable to the *PDOCmdParam1* output variable of ZW_CmdControl.
E001_Command_Parameter2	Command Parameter 2 in Command area I/O port on sensor head 1	UINT	UINT#0	Command Parameter 2 in the command area I/O port of the displacement sensor on the sensor head 1. Assign this variable to the *PDOCmdParam2* output variable of ZW_CmdControl.

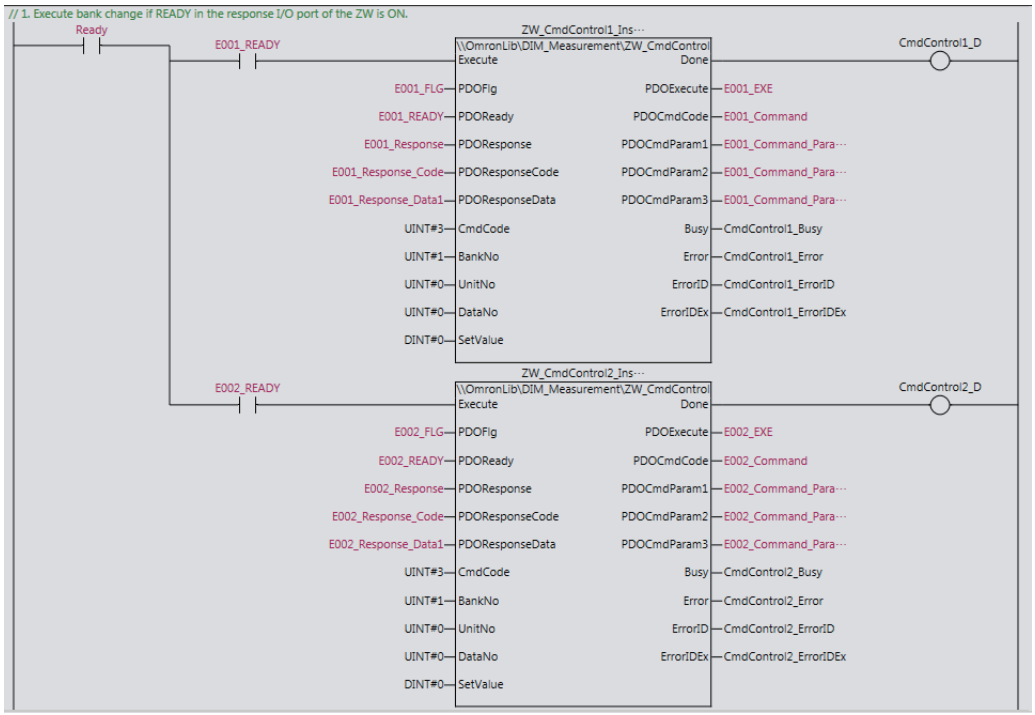
Name	Meaning	Data type	Default	Description
E001_Command_Parameter3	Command Parameter 3 in Command area I/O port on sensor head 1	DINT	DINT#0	Command Parameter 3 in the command area I/O port of the displacement sensor on the sensor head 1. Assign this variable to the *PDOCmdParam3* output variable of ZW_CmdControl.
E001_TASKSTAT_T1	TASKSTAT_T1 Bit in Response area I/O port on sensor head 1	BOOL	---	TASKSTAT_T1 Bit in the response area I/O port of the displacement sensor on the sensor head 1.
E001_ZEROSTAT_T1	Task 1 Zero Reset Status in Response area I/O port on sensor head 1	BOOL	---	Task 1 Zero Reset Status in the response area I/O port of the displacement sensor on the sensor head 1. Assign this variable to the *PDOZeroStat* input variable of ZW_ZeroResetControl.
E001_ZERO_T1	Execute Zero Reset for Task 1 in Command area I/O port on sensor head 1	BOOL	FALSE	Execute Zero Reset for Task 1 in the command area I/O port of the displacement sensor on the sensor head 1. Assign this variable to the *PDOZero* output variable of ZW_ZeroResetControl.
E001_ZEROCLR_T1	Clear Zero Reset for Task 1 in Command area I/O port on sensor head 1	BOOL	FALSE	Clear Zero Reset for Task 1 in the command area I/O port of the displacement sensor on the sensor head 1. Assign this variable to the *PDOZeroClr* output variable of ZW_ZeroResetControl.
E002_FLG	Command Done in Response area I/O port on sensor head 2	BOOL	---	Command Done in the response area I/O port of the displacement sensor on the sensor head 2. Assign this variable to the *PDOFlg* input variable of ZW_CmdControl.
E002_READY	Signal Input Ready Status in Response area I/O port on sensor head 2	BOOL	---	Signal Input Ready Status in the response area I/O port of the displacement sensor on the sensor head 2. Assign this variable to the *PDOReady* input variable of ZW_CmdControl.
E002_Response	Command Echo Back in Response area I/O port on sensor head 2	BOOL	---	Command Echo Back in the response area I/O port of the displacement sensor on the sensor head 2. Assign this variable to the *PDOResponse* input variable of ZW_CmdControl.
E002_Response_Code	Response Code in Response area I/O port on sensor head 2	DWORD	---	Response Code in the response area I/O port of the displacement sensor on the sensor head 2. Assign this variable to the *PDOResponseCode* input variable of ZW_CmdControl.

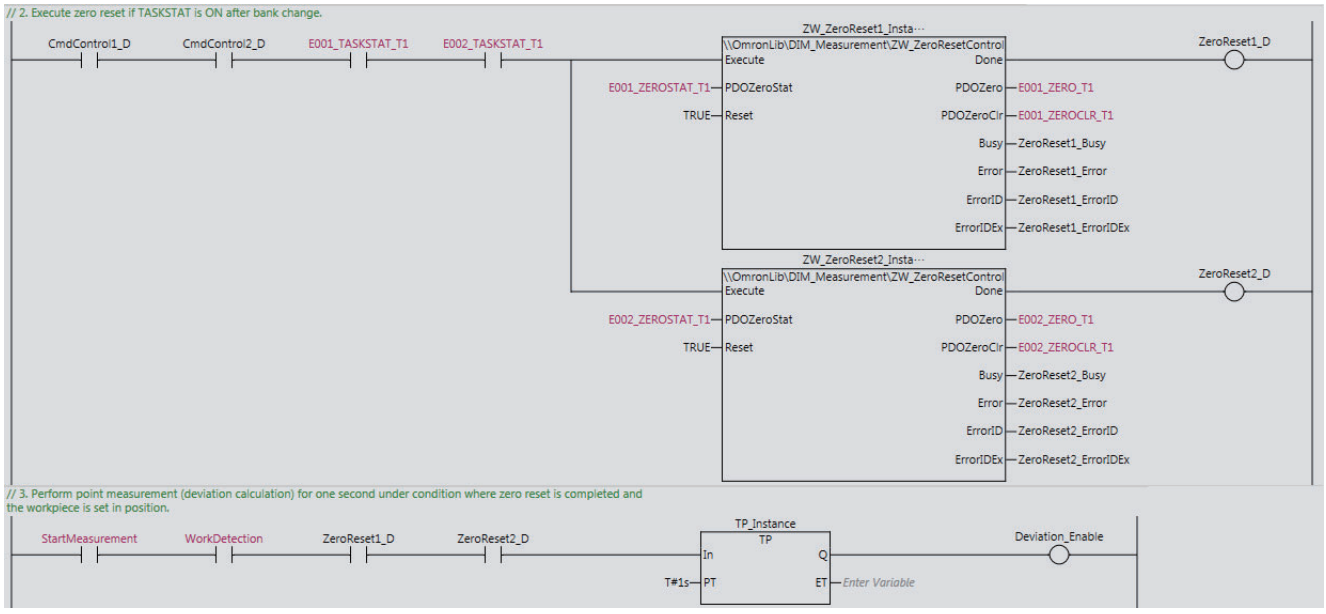
Name	Meaning	Data type	Default	Description
E002_Response_Data1	Response Data in Response area I/O port on sensor head 2	DINT	---	Response Data in the response area I/O port of the displacement sensor on the sensor head 2. Assign this variable to the *PDOResponseData* input variable of ZW_CmdControl.
E002_EXE	Execute Command in Command area I/O port on sensor head 2	BOOL	FALSE	Execute Command in the command area I/O port of the displacement sensor on the sensor head 2. Assign this variable to the *PDOExecute* output variable of ZW_CmdControl.
E002_Command	Command Code in Command area I/O port on sensor head 2	DWORD	16#0	Command Code in the command area I/O port of the displacement sensor on the sensor head 2. Assign this variable to the *PDOCmdCode* output variable of ZW_CmdControl.
E002_Command_Parameter1	Command Parameter 1 in Command area I/O port on sensor head 2	UINT	UINT#0	Command Parameter 1 in the command area I/O port of the displacement sensor on the sensor head 2. Assign this variable to the *PDOCmdParam1* output variable of ZW_CmdControl.
E002_Command_Parameter2	Command Parameter 2 in Command area I/O port on sensor head 2	UINT	UINT#0	Command Parameter 2 in the command area I/O port of the displacement sensor on the sensor head 2. Assign this variable to the *PDOCmdParam2* output variable of ZW_CmdControl.
E002_Command_Parameter3	Command Parameter 3 in Command area I/O port on sensor head 2	DINT	DINT#0	Command Parameter 3 in the command area I/O port of the displacement sensor on the sensor head 2. Assign this variable to the *PDOCmdParam3* output variable of ZW_CmdControl.
E002_TASKSTAT_T1	TASKSTAT_T1 Bit in Response area I/O port on sensor head 2	BOOL	---	TASKSTAT_T1 Bit in the response area I/O port of the displacement sensor on the sensor head 2.
E002_ZEROSTAT_T1	Task 1 Zero Reset Status in Response area I/O port on sensor head 2	BOOL	---	Task 1 Zero Reset Status in the response area I/O port of the displacement sensor on the sensor head 2. Assign this variable to the *PDOZeroStat* input variable of ZW_ZeroResetControl.

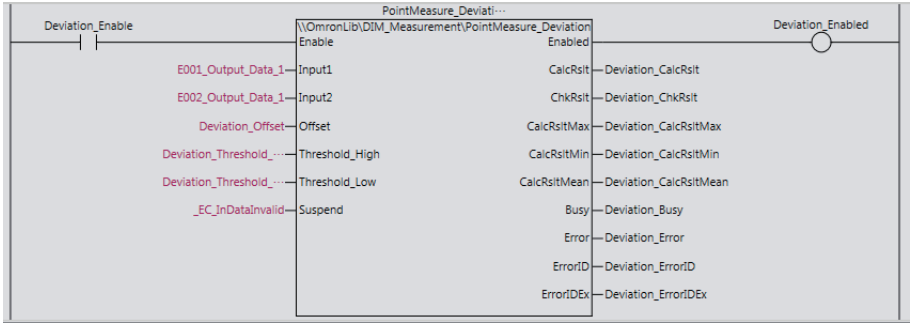
Name	Meaning	Data type	Default	Description
E002_ZERO_T1	Execute Zero Reset for Task 1 in Command area I/O port on sensor head 2	BOOL	FALSE	Execute Zero Reset for Task 1 in the command area I/O port of the displacement sensor on the sensor head 2. Assign this variable to the *PDOZero* output variable of ZW_ZeroResetControl.
E002_ZEROCLR_T1	Clear Zero Reset for Task 1 in Command area I/O port on sensor head 2	BOOL	FALSE	Clear Zero Reset for Task 1 in the command area I/O port of the displacement sensor on the sensor head 2. Assign this variable to the *PDOZeroClr* output variable of ZW_ZeroResetControl.
StartMeasurement	Start Deviation Calculation	BOOL	FALSE	Starts deviation calculation when the variable changes to TRUE.
WorkDetection	Workpiece in Position Status	BOOL	FALSE	TRUE: The workpiece for measurement is set in position. FALSE: The workpiece for measurement is not set in position.
E001_Output_Data_1	OUT1 Data in Output area I/O port on sensor head 1	DINT	---	OUT1 Data in the output area I/O port of the sensor head 1. Assign this variable to the *Input1* input variable of PointMeasure_Deviation.
E002_Output_Data_1	OUT1 Data in Output area I/O port on sensor head 2	DINT	---	OUT1 Data in the output area I/O port of the sensor head 2. Assign this variable to the *Input2* input variable of PointMeasure_Deviation.
Deviation_Offset	Set Offset	DINT	DINT#0	Sets the input value for *Offset*. Assign this variable to the *Offset* input variable of PointMeasure_Deviation.
Deviation_Threshold_High	Set Threshold (High)	DINT	DINT#0	Sets the threshold (High). Assign this variable to the *Threshold_High* input variable of PointMeasure_Deviation.
Deviation_Threshold_Low	Set Threshold (Low)	DINT	DINT#0	Sets the threshold (Low). Assign this variable to the *Threshold_Low* input variable of PointMeasure_Deviation.
_EC_InDataInvalid	Input Data Disabled	BOOL	---	A system-defined variable for EtherCAT communications. TRUE when the process data communications executed in the primary periodic task is not normal and the input value is not valid. Assign this variable to the *Suspend* input variable of PointMeasure_Deviation.
Deviation_CalcRslt	Calculation Result	DINT	---	Assign this variable to the *CalcRslt* output variable of PointMeasure_Deviation.

Name	Meaning	Data type	Default	Description
Deviation_ChkRslt	Judgment Output	BOOL	---	Assign this variable to the *ChkRslt* output variable of PointMeasure_Deviation.
Deviation_CalcRsltMax	Calculation Result Maximum Value	DINT	---	Assign this variable to the *CalcRsltMax* output variable of PointMeasure_Deviation.
Deviation_CalcRsltMin	Calculation Result Minimum Value	DINT	---	Assign this variable to the *CalcRsltMin* output variable of PointMeasure_Deviation.
Deviation_CalcRsltMean	Calculation Result Mean Value	DINT	---	Assign this variable to the *CalcRsltMean* output variable of PointMeasure_Deviation.

Ladder Diagram







ST

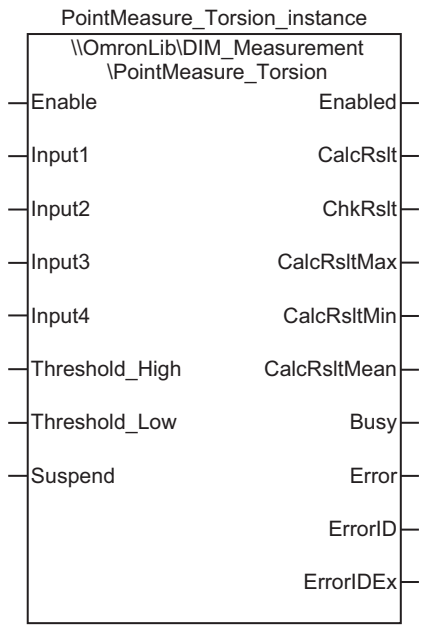
```
// 1. Execute bank change if READY in the response I/O port of the ZW is ON.
IF (Ready=TRUE) THEN
   IF(E001_READY=TRUE) THEN
      CmdControl1_Execute:=TRUE;
   ELSE
      CmdControl1_Execute:=FALSE;
   END_IF;
   IF(E002_READY=TRUE) THEN
      CmdControl2_Execute:=TRUE;
   ELSE
      CmdControl2_Execute:=FALSE;
   END_IF;
ELSE
   CmdControl1_Execute:=FALSE;
   CmdControl2_Execute:=FALSE;
END_IF;

// ZW Command Control function block for slave 1
ZW_CmdControl1_Instance
(
Execute:=CmdControl1_Execute,
PDOFlg:=E001_FLG,
PDOReady:=E001_READY,
PDOResponse:=E001_Response,
PDOResponseCode:=E001_Response_Code,
PDOResponseData:=E001_Response_Data1,
CmdCode:=3,
BankNo:=UINT#1,
UnitNo:=UINT#0,
DataNo:=UINT#0,
SetValue:=DINT#0,
Done=>CmdControl1_D,
PDOExecute=>E001_EXE,
PDOCmdCode=>E001_Command,
PDOCmdParam1=>E001_Command_Parameter1,
PDOCmdParam2=>E001_Command_Parameter2,
PDOCmdParam3=>E001_Command_Parameter3,
Busy=>CmdControl1_Busy,
Error=>CmdControl1_Error,
ErrorID=>CmdControl1_ErrorID,
ErrorIDEx=>CmdControl1_ErrorIDEx
);

// ZW Command Control function block for slave 2
ZW_CmdControl2_Instance
(
Execute:=CmdControl2_Execute,
PDOFlg:=E002_FLG,
PDOReady:=E002_READY,
PDOResponse:=E002_Response,
```

```
PDOResponseCode:=E002_Response_Code,
PDOResponseData:=E002_Response_Data1,
CmdCode:=3,
BankNo:=UINT#1,
UnitNo:=UINT#0,
DataNo:=UINT#0,
SetValue:=DINT#0,
Done=>CmdControl2_D,
PDOExecute=>E002_EXE,
PDOCmdCode=>E002_Command,
PDOCmdParam1=>E002_Command_Parameter1,
PDOCmdParam2=>E002_Command_Parameter2,
PDOCmdParam3=>E002_Command_Parameter3,
Busy=>CmdControl2_Busy,
Error=>CmdControl2_Error,
ErrorID=>CmdControl2_ErrorID,
ErrorIDEx=>CmdControl2_ErrorIDEx
);

// 2. Execute zero reset if TASKSTAT is ON after bank change.
IF (CmdControl1_D=TRUE) AND
   (CmdControl2_D=TRUE) AND
   (E001_TASKSTAT_T1=TRUE) AND
   (E002_TASKSTAT_T1=TRUE) THEN
   ZeroReset1_Execute:=TRUE;
   ZeroReset2_Execute:=TRUE;
ELSE
   ZeroReset1_Execute:=FALSE;
   ZeroReset2_Execute:=FALSE;
END_IF;

// Zero Reset Control function block for slave 1
ZW_ZeroReset1_Instance
(
Execute:=ZeroReset1_Execute,
PDOZeroStat:=E001_ZEROSTAT_T1,
Reset:=TRUE,
Done=>ZeroReset1_D,
PDOZero=>E001_ZERO_T1,
PDOZeroClr=>E001_ZEROCLR_T1,
Busy=>ZeroReset1_Busy,
Error=>ZeroReset1_Error,
ErrorID=>ZeroReset1_ErrorID,
ErrorIDEx=>ZeroReset1_ErrorIDEx
);

// Zero Reset Control function block for slave 2
ZW_ZeroReset2_Instance
(
Execute:=ZeroReset2_Execute,
PDOZeroStat:=E002_ZEROSTAT_T1,
Reset:=TRUE,
Done=>ZeroReset2_D,
PDOZero=>E002_ZERO_T1,
PDOZeroClr=>E002_ZEROCLR_T1,
Busy=>ZeroReset2_Busy,
Error=>ZeroReset2_Error,
ErrorID=>ZeroReset2_ErrorID,
ErrorIDEx=>ZeroReset2_ErrorIDEx
);

// 3. Perform point measurement (deviation calculation) for one second under condi-
tion where zero reset is completed and the workpiece is set in position.
IF (StartMeasurement=TRUE) AND
    (WorkDetection=TRUE) AND
```

```
   (ZeroReset1_D=TRUE) AND
   (ZeroReset2_D=TRUE) THEN
   TP_Instance(In:=TRUE, PT:=T#1s, Q=>Deviation_Enable);
ELSE
   TP_Instance(In:=FALSE, Q=>Deviation_Enable);
END_IF;

// Deviation Calculation function block
PointMeasure_Deviation_Instance
(
Enable:=Deviation_Enable,
Input1:=E001_Output_Data_1,
Input2:=E002_Output_Data_1,
Offset:=Deviation_Offset,
Threshold_High:=Deviation_Threshold_High,
Threshold_Low:=Deviation_Threshold_Low,
Suspend:=_EC_InDataInvalid,
Enabled=>Deviation_Enabled,
CalcRslt=>Deviation_CalcRslt,
ChkRslt=>Deviation_ChkRslt,
CalcRsltMax=>Deviation_CalcRsltMax,
CalcRsltMin=>Deviation_CalcRsltMin,
CalcRsltMean=>Deviation_CalcRsltMean,
Busy=>Deviation_Busy,
Error=>Deviation_Error,
ErrorID=>Deviation_ErrorID,
ErrorIDEx=>Deviation_ErrorIDEx
);
```

PointMeasure_Torsion

The PointMeasure_Torsion function block calculates the degree of torsion from the measurement values (*Input1, Input2, Input3*, and *Input4*) obtained from four measurement sensors outputting it on the calculation result.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
PointMeasure_ Torsion	Torsion Calculation	FB		PointMeasure_Torsion_Instance (Enable, Input1, Input2, Input3, Input4, Threshold_High, Threshold_Low, Suspend, Enabled, CalcRslt, ChkRslt, CalcRsltMax, CalcRsltMin, CalcRsltMean, Busy, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_PointMeasurement_V1_1.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00093
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Product name	Model numbers	Version
Device	Sensor Communications Unit	E3NW-ECT	Rev 1.03 or later
	Distributed Sensor Unit	E3NW-DS	-
	Contact-Type Smart Amplifier	E3NC-TA0	-
	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Enable	Execution	TRUE: Execute FALSE: Do not execute	BOOL	Depends on data type.	FALSE
Input1	Sensor Head 1 Input Value	The input value of the sensor head 1 is set.	DINT	Depends on data type.	0
Input2	Sensor Head 2 Input Value	The input value of the sensor head 2 is set.	DINT	Depends on data type.	0
Input3	Sensor Head 3 Input Value	The input value of the sensor head 3 is set.	DINT	Depends on data type.	0
Input4	Sensor Head 4 Input Value	The input value of the sensor head 4 is set.	DINT	Depends on data type.	0
Threshold_High*1	Threshold (High)	Outputs the judgement of whether the threshold is exceeded. Specify the thresholds always as: Threshold (High) ≥ Threshold (Low).	DINT	Depends on data type.	0
Threshold_Low*1	Threshold (Low)		DINT		0
Suspend	Suspension	Set to TRUE when inputting measurement data on which you want to disable the calculation during execution.	BOOL	Depends on data type.	FALSE

*1. If the set values are used for calculation at the task period when FALSE changed to TRUE on *Enable* (Execution) of this function block, the values which changed while *Enable* was TRUE are not reflected in the calculation.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Enabled	Calculation in Progress	TRUE: Calculation in progress FALSE: Calculation stopped	BOOL	Depends on data type.	—
CalcRslt	Calculation Result	Stores the result of torsion calculation based on the measurement values of sensors.	DINT	Depends on data type.	—
ChkRslt	Judgement Output	Turns to TRUE if the calculation results meet all the following criteria: • Not less than the threshold (Low). • Not more than the threshold (High).	BOOL	Depends on data type.	—
CalcRsltMax	Calculation Result Maximum Value	Outputs the maximum value of *CalcRslt* under calculation.	DINT	Depends on data type.	—
CalcRsltMin	Calculation Result Minimum Value	Outputs the minimum value of *CalcRslt* under calculation.	DINT	Depends on data type.	—
CalcRslt-Mean	Calculation Result Mean Value	Outputs the mean value of *CalcRslt* under calculation.	DINT	Depends on data type.	—
Busy	Execution in Progress	Turns to TRUE while the processing is in execution; turns to FALSE while the processing is not in execution.	BOOL	Depends on data type.	—
Error	Error	Outputs TRUE if an error occurs.	BOOL	Depends on data type.	—
ErrorID	Error Code	Outputs the error code if an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the extended error code if an error occurs.	DWORD	*1	—

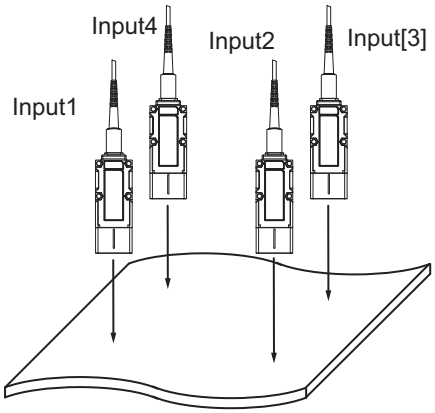
*1. Refer to the *Troubleshooting* on page 67.

Function

The PointMeasure_Torsion function block calculates the degree of torsion from the measurement values (*Input1, Input2, Input3*, and *Input4*) obtained from four measurement sensors, any type of the following sensors, outputting it on the calculation result.

- Sensor amplifier connected to the Sensor Communications Unit (E3NW-ECT)
- Confocal Fiber Displacement Sensor (ZW-8000/7000/5000)

Calculation result = (*Input1* - *Input2*) - (*Input4* - *Input3*)



(a) As you set *Enable* (Execution) to TRUE, the torsion (arithmetic expression: (Input1 - Input2) - (Input4 - Input3)) is calculated from *Input1, Input2, Input3*, and *Input4* to be set for *CalcRslt* (Calculation Result). When the value of *CalcRslt* is not less than the threshold (Low) and not more than the threshold (High), *ChkRslt* (Judgement Output) turns to TRUE. While *Enable* is TRUE, the calculation continues for each task period. As *Enable* turns to FALSE, the calculation is terminated, but *CalcRslt* and *ChkRslt* hold their values, which will be reset next time *Enable* changes to TRUE.

(b) Under calculation, *CalcRsltMax* (Calculation Result Maximum Value), *CalcRsltMin* (Calculation Result Minimum Value), and *CalcRsltMean* (Calculation Result Mean Value) are output. While *Enable* is TRUE, the calculation continues. As *Enable* turns to FALSE, the calculation is terminated, but *CalcRsltMax*, *CalcRsltMin*, and *CalcRsltMean* hold their values, which will be reset next time *Enable* changes to TRUE.

(c) The mean value is calculated from values that are input while *Enable* is TRUE. However, if the maximum number of data points (number of calls of a function block while *Enable* is TRUE) is 65535 and then this maximum number is exceeded, the mean value is not updated.

(d) Setting *Suspend* (Suspension) to TRUE allows the calculation to be suspended while the execution is in progress. This function is useful to remove the calculation results in the task periods where process data communications fail and the measurement data is not updated correctly. During suspension, *Enabled* (Calculation in Progress) turns to FALSE, where the number of data points used for mean value calculation is not updated. As you set *Suspend* to FALSE, calculation resumes.

(e) If an error occurs, *Error* turns to TRUE, where the processing of function blocks is interrupted. In addition, the error code is output to *ErrorID* and *ErrorIDEx*.

For details on the error codes, refer to *Troubleshooting* on page 67.

Timing Charts

Refer to the *Timing Charts* on page 52 for PointMeasure_Deviation.

Precautions for Use

- For input to this function block, consistently use either sensors connected to the E3NW or sensors of the ZW-8000/7000/5000. The combined use results in a failure to perform the correct operation because the unit is different from each other.
- Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.
- For the ZW-8000/7000/5000 series, confirm that the following pre-conditions are met before executing this function block.
  a) The ZW_CmdControl function block is unexecuted.
  b) *TASKSTAT_TX* which is the status of the target task for measurement slave is TRUE.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3CA0	0x1	The threshold (High) is smaller than the threshold (Low).	Specify the thresholds so that the threshold (High) will be larger than the threshold (Low).

Sample Programming

Program Description

For one second after the measurement start command, this program uses the measurement values of four types of sensors connected via EtherCAT as inputs to perform torsion calculation by PointMeasure_Torsion.

Preconditions

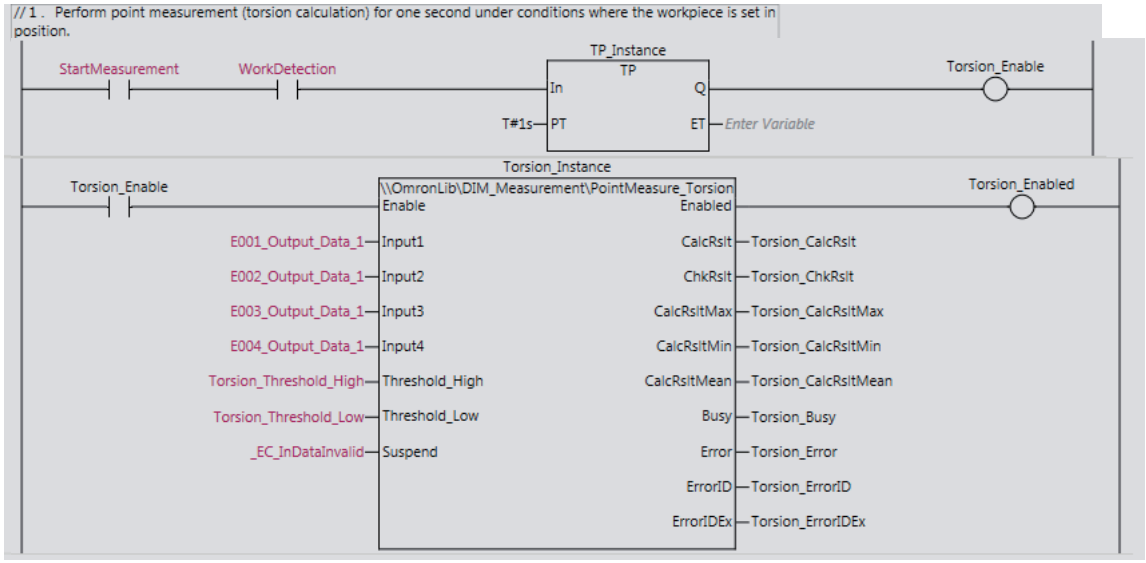
- Create device variables for the measurement values of the displacement sensor with which you want to perform measurement and use external references to the variables in the program. Refer to the *Sysmac Studio Version 1 Operation Manual* (Cat. No. W504) for details on how to create device variables.
- If E9NC-TA sensors are used to input values, change the PDO mapping settings for the E3NW-ECT so that the sensor detection amount is 4 bytes.
  Example: If E9NC-TA sensors are connected to an E3NW-ECT with unit number 01
  →Change the setting of 273rd transmit PDO Mapping (No_01 Detection Level IN 1) from *Input* to *Not selected*.
  →Change the setting of 423rd transmit PDO Mapping (No_01 Detection Level (4 bytes) from *Not selected* to *Input*.
- If ZW-8000/7000/5000 sensors are used to input values, set the measurement item for each sensor to height.
- Set each sensor to zero beforehand.
- Select a combination of the workpiece and sensors that provides measurement values in the measurement range of the sensors.

Main Variables

Name	Meaning	Data type	Default	Description
StartMeasurement	Start Torsion Calculation	BOOL	FALSE	Starts torsion calculation when the variable changes to TRUE.
WorkDetection	Workpiece in Position Status	BOOL	FALSE	TRUE: The workpiece for measurement is set in position. FALSE: The workpiece for measurement is not set in position.
E001_Output_Data_1	OUT1 Data in Output area I/O port on sensor head 1	DINT	---	OUT1 Data in the output area I/O port of the sensor head 1. Assign this variable to the *Input1* input variable of PointMeasure_Torsion.
E002_Output_Data_1	OUT1 Data in Output area I/O port on sensor head 2	DINT	---	OUT1 Data in the output area I/O port of the sensor head 2. Assign this variable to the *Input2* input variable of PointMeasure_Torsion.
E003_Output_Data_1	OUT1 Data in Output area I/O port on sensor head 3	DINT	---	OUT1 Data in the output area I/O port of the sensor head 3. Assign this variable to the *Input3* input variable of PointMeasure_Torsion.
E004_Output_Data_1	OUT1 Data in Output area I/O port on sensor head 4	DINT	---	OUT1 Data in the output area I/O port of the sensor head 4. Assign this variable to the *Input4* input variable of PointMeasure_Torsion.
Torsion_Threshold_High	Set Threshold (High)	DINT	DINT#0	Sets the threshold (High). Assign this variable to the *Threshold_High* input variable of PointMeasure_Torsion.
Torsion_Threshold_Low	Set Threshold (Low)	DINT	DINT#0	Sets the threshold (Low). Assign this variable to the *Threshold_Low* input variable of PointMeasure_Torsion.
_EC_InDataInvalid	Input Data Disabled	BOOL	---	A system-defined variable for EtherCAT communications. TRUE when the process data communications executed in the primary periodic task is not normal and the input value is not valid. Assign this variable to the *Suspend* input variable of PointMeasure_Torsion.
Torsion _CalcRslt	Calculation Result	DINT	---	Assign this variable to the *CalcRslt* output variable of PointMeasure_Torsion.
Torsion _ChkRslt	Judgment Output	BOOL	---	Assign this variable to the *ChkRslt* output variable of PointMeasure_Torsion.
Torsion _CalcRsltMax	Calculation Result Maximum Value	DINT	---	Assign this variable to the *CalcRsltMax* output variable of PointMeasure_Torsion.

Name	Meaning	Data type	Default	Description
Torsion _CalcRsltMin	Calculation Result Minimum Value	DINT	---	Assign this variable to the *CalcRsltMin* output variable of PointMeasure_Torsion.
Torsion _CalcRsltMean	Calculation Result Mean Value	DINT	---	Assign this variable to the *CalcRsltMean* output variable of PointMeasure_Torsion.

Ladder Diagram

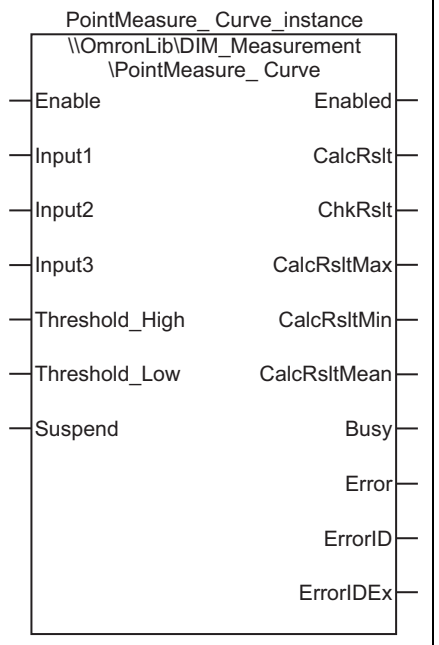


ST

```
// 1. Perform point measurement (torsion calculation) for one second under condition
where the workpiece is set in position.
IF (StartMeasurement=TRUE) AND (WorkDetection=TRUE) THEN
   TP_Instance(In:=TRUE, PT:=T#1s, Q=>Torsion_Enable);
ELSE
   TP_Instance(In:=FALSE, Q=>Torsion_Enable);
END_IF;

// Torsion Calculation function block
PointMeasure_Torsion_Instance
(
Enable:=Torsion_Enable,
Input1:=E001_Output_Data_1,
Input2:=E002_Output_Data_1,
Input3:=E003_Output_Data_1,
Input4:=E004_Output_Data_1,
Threshold_High:=Torsion_Threshold_High,
Threshold_Low:=Torsion_Threshold_Low,
Suspend:=_EC_InDataInvalid,
Enabled=>Torsion_Enabled,
CalcRslt=>Torsion_CalcRslt,
ChkRslt=>Torsion_ChkRslt,
CalcRsltMax=>Torsion_CalcRsltMax,
CalcRsltMin=>Torsion_CalcRsltMin,
CalcRsltMean=>Torsion_CalcRsltMean,
Busy=>Torsion_Busy,
Error=>Torsion_Error,
ErrorID=>Torsion_ErrorID,
ErrorIDEx=>Torsion_ErrorIDEx
);
```

PointMeasure_Curve

The PointMeasure_Curve function block calculates the degree of curve from the measurement values (*Input1, Input2*, and *Input3*) obtained from three measurement sensors outputting it on the calculation result.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
PointMea-sure_Curve	Curve Calculation	FB		PointMeasure_ Curve_Instance (Enable, Input1, Input2, Input3, Threshold_High, Threshold_Low, Suspend, Enabled, CalcRslt, ChkRslt, CalcRsltMax, CalcRsltMin, CalcRsltMean, Busy, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_PointMeasurement_V1_1.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00094
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Product name	Model numbers	Version
Device	Sensor Communications Unit	E3NW-ECT	Rev 1.03 or later
	Distributed Sensor Unit	E3NW-DS	-
	Contact-Type Smart Amplifier	E3NC-TA0	-
	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Enable	Execution	TRUE: Execute FALSE: Do not execute	BOOL	Depends on data type.	FALSE
Input1	Sensor Head 1 Input Value	The input value of the sensor head 1 is set.	DINT	Depends on data type.	0
Input2	Sensor Head 2 Input Value	The input value of the sensor head 2 is set.	DINT	Depends on data type.	0
Input3	Sensor Head 3 Input Value	The input value of the sensor head 3 is set.	DINT	Depends on data type.	0
Threshold_High*1	Threshold (High)	Judges the value from the calculation result is in or out of the threshold range.	DINT	Depends on data type.	0
Threshold_Low*1	Threshold (Low)		DINT		0
Suspend	Suspension	Set to TRUE when inputting measurement data on which you want to disable the calculation during execution.	BOOL	Depends on data type.	FALSE

*1. If the set values are used for calculation at the task period when FALSE changed to TRUE on *Enable* (Execution) of this function block, the values which changed while *Enable* was TRUE are not reflected in the calculation.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Enabled	Calculation in Progress	TRUE: Calculation in progress FALSE: Calculation stopped	BOOL	Depends on data type.	—
CalcRslt	Calculation Result	Stores the result of curve calculation based on the measurement values of sensors.	DINT	Depends on data type.	—
ChkRslt	Judgement Output	The judgement output turns to TRUE when the calculation results are not less than the threshold (Low) and not more than the threshold (High)	BOOL	Depends on data type.	—
CalcRsltMax	Calculation Result Maximum Value	Outputs the maximum value of *CalcRslt* under calculation.	DINT	Depends on data type.	—
CalcRsltMin	Calculation Result Minimum Value	Outputs the minimum value of *CalcRslt* under calculation.	DINT	Depends on data type.	—
CalcRslt-Mean	Calculation Result Mean Value	Outputs the mean value of *CalcRslt* under calculation.	DINT	Depends on data type.	—
Busy	Execution in Progress	Turns to TRUE while the processing is in execution; turns to FALSE while the processing is not in execution.	BOOL	Depends on data type.	—
Error	Error	Outputs TRUE if an error occurs.	BOOL	Depends on data type.	—
ErrorID	Error Code	Outputs the error code if an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the extended error code if an error occurs.	DWORD	*1	—

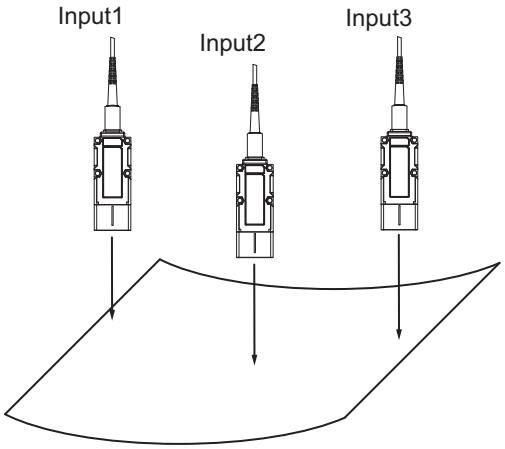
*1. Refer to the *Troubleshooting* on page 75.

Function

The PointMeasure_Curve function block calculates the degree of curve from the measurement values (*Input1, Input2*, and *Input3*) obtained from three measurement sensors, any type of the following sensors, outputting it on the calculation result.

- Sensor amplifier connected to the Sensor Communications Unit (E3NW-ECT)
- Confocal Fiber Displacement Sensor (ZW-8000/7000/5000)

Calculation result = (Input1 + Input3) / 2 – Input2



(a) As you set *Enable* (Execution) to TRUE, the degree of curve is calculated from *Input1, Input2*, and *Input3* to be set for *CalcRslt* (Calculation Result). The arithmetic expression for the degree of curve is as follows:
(Input1 + Input3) / 2 - Input2
When the value of *CalcRslt* is not less than the threshold (Low) and not more than the threshold (High), *ChkRslt* (Judgement Output) turns to TRUE. While *Enable* is TRUE, the calculation continues for each task period. As *Enable* turns to FALSE, the calculation is terminated, but *CalcRslt* and *ChkRslt* hold their values, which will be reset next time *Enable* changes to TRUE.

(b) Under calculation, *CalcRsltMax* (Calculation Result Maximum Value), *CalcRsltMin* (Calculation Result Minimum Value), and *CalcRsltMean* (Calculation Result Mean Value) are output. While *Enable* is TRUE, the calculation continues. As *Enable* turns to FALSE, the calculation is terminated, but *CalcRsltMax*, *CalcRsltMin*, and *CalcRsltMean* hold their values, which will be reset next time *Enable* changes to TRUE.

(c) The mean value is calculated from values that are input while *Enable* is TRUE. However, that the maximum number of data points (number of calls of a function block while *Enable* is TRUE) is 65535 and then this maximum number is exceeded, the mean value is not updated.

(d) Setting *Suspend* (Suspension) to TRUE allows the calculation to be suspended while the execution is in progress. This function is useful to remove the calculation results in the task periods where process data communications fail and the measurement data is not updated correctly. During suspension, *Enabled* (Calculation in Progress) turns to FALSE, where the number of data points used for mean value calculation is not updated. As you set *Suspend* to FALSE, calculation resumes.

(e) If an error occurs, *Error* turns to TRUE, where the processing of function blocks is interrupted. In addition, the error code is output to *ErrorID* and *ErrorIDEx*.

For details on the error codes, refer to *Troubleshooting* on page 75.

Timing Charts

Refer to the *Timing Charts* on page 52 for PointMeasure_Deviation.

Precautions for Use

- For input to this function block, consistently use either sensors connected to the E3NW or sensors of the ZW-8000/7000/5000. The combined use results in a failure to perform the correct operation because the unit is different from each other.
- Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.
- For the ZW-8000/7000/5000 series, confirm that the following pre-conditions are met before executing this function block.
  a) The ZW_CmdControl function block is unexecuted.
  b) *TASKSTAT_TX* which is the status of the target task for measurement slave is TRUE.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3CA1	0x1	The threshold (High) is smaller than the threshold (Low).	Specify the thresholds so that the threshold (High) will be larger than the threshold (Low).

Sample Programming

Program Description

For one second after the measurement start command, this program uses the measurement values of three types of sensors connected via EtherCAT as inputs to perform curve calculation by PointMeasure_Curve.

Preconditions

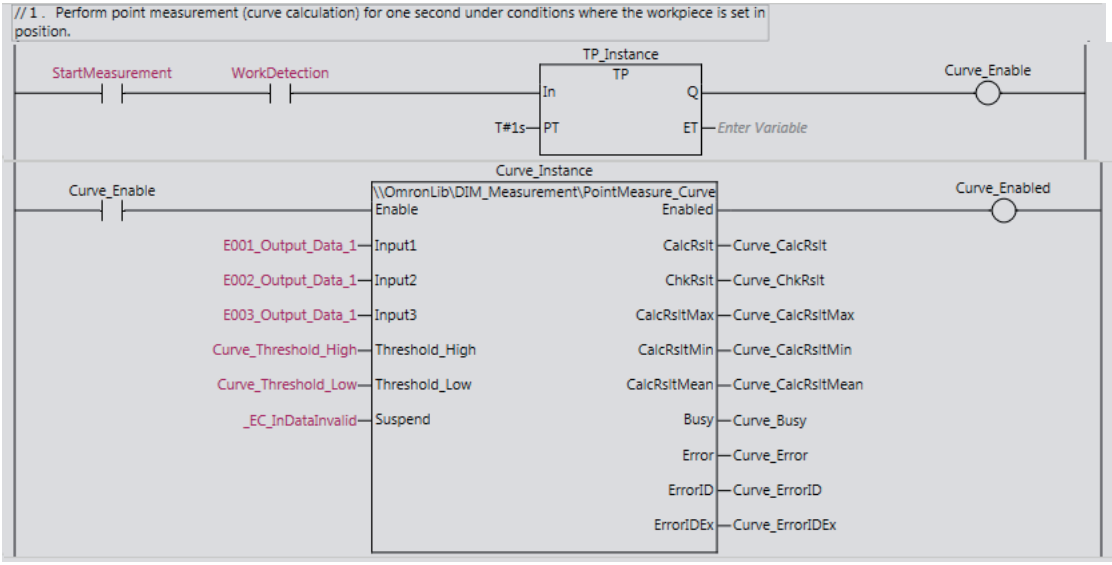
Refer to *Preconditions* on page 67 for the PointMeasure_Torsion function block.

Main Variables

Name	Meaning	Data type	Default	Description
StartMeasurement	Start Curve Calculation	BOOL	FALSE	Starts curve calculation when the variable changes to TRUE.
WorkDetection	Workpiece in Position Status	BOOL	FALSE	TRUE: The workpiece for measurement is set in position. FALSE: The workpiece for measurement is not set in position.
E001_Output_Data_1	OUT1 Data in Output area I/O port on sensor head 1	DINT	---	OUT1 Data in the output area I/O port of the sensor head 1. Assign this variable to the *Input1* input variable of PointMeasure_Curve.
E002_Output_Data_1	OUT1 Data in Output area I/O port on sensor head 2	DINT	---	OUT1 Data in the output area I/O port of the sensor head 2. Assign this variable to the *Input2* input variable of PointMeasure_Curve.
E003_Output_Data_1	OUT1 Data in Output area I/O port on sensor head 3	DINT	---	OUT1 Data in the output area I/O port of the sensor head 3. Assign this variable to the *Input3* input variable of PointMeasure_Curve.
Curve_Threshold_High	Set Threshold (High)	DINT	DINT#0	Sets the threshold (High). Assign this variable to the *Threshold_High* input variable of PointMeasure_Curve.
Curve_Threshold_Low	Set Threshold (Low)	DINT	DINT#0	Sets the threshold (Low). Assign this variable to the *Threshold_Low* input variable of PointMeasure_Curve.
_EC_InDataInvalid	Input Data Disabled	BOOL	---	A system-defined variable for EtherCAT communications. TRUE when the process data communications executed in the primary periodic task is not normal and the input value is not valid. Assign this variable to the *Suspend* input variable of PointMeasure_Curve.
Curve _CalcRslt	Calculation Result	DINT	---	Assign this variable to the *CalcRslt* output variable of PointMeasure_Curve.

Name	Meaning	Data type	Default	Description
Curve_ChkRslt	Judgment Output	BOOL	---	Assign this variable to the *ChkRslt* output variable of PointMeasure_Curve.
Curve _CalcRsltMax	Calculation Result Maximum Value	DINT	---	Assign this variable to the *CalcRsltMax* output variable of PointMeasure_Curve.
Curve _CalcRsltMin	Calculation Result Minimum Value	DINT	---	Assign this variable to the *CalcRsltMin* output variable of PointMeasure_Curve.
Curve _CalcRsltMean	Calculation Result Mean Value	DINT	---	Assign this variable to the *CalcRsltMean* output variable of PointMeasure_Curve.

Ladder Diagram

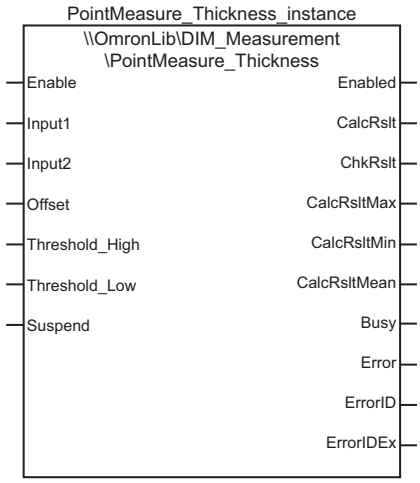


ST

```
// 1. Perform point measurement (curve calculation) for one second under condition
where the workpiece is set in position.
IF (StartMeasurement=TRUE) AND (WorkDetection=TRUE) THEN
   TP_Instance(In:=TRUE, PT:=T#1s, Q=>Curve_Enable);
ELSE
   TP_Instance(In:=FALSE, Q=>Curve_Enable);
END_IF;

// Curve Calculation function block
PointMeasure_Curve_Instance
(
Enable:=Curve_Enable,
Input1:=E001_Output_Data_1,
Input2:=E002_Output_Data_1,
Input3:=E003_Output_Data_1,
Threshold_High:=Curve_Threshold_High,
Threshold_Low:=Curve_Threshold_Low,
Suspend:=_EC_InDataInvalid,
Enabled=>Curve_Enabled,
CalcRslt=>Curve_CalcRslt,
ChkRslt=>Curve_ChkRslt,
CalcRsltMax=>Curve_CalcRsltMax,
CalcRsltMin=>Curve_CalcRsltMin,
CalcRsltMean=>Curve_CalcRsltMean,
Busy=>Curve_Busy,
Error=>Curve_Error,
ErrorID=>Curve_ErrorID,
ErrorIDEx=>Curve_ErrorIDEx
);
```

PointMeasure_Thickness

The PointMeasure_Thickness function block calculates the thickness from the measurement values (*Input1* and *Input2*) obtained from two measurement sensors outputting it on the calculation result.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
PointMeasure_ Thickness	Thickness Calculation	FB		PointMeasure_Thickness _Instance (Enable, Input1, Input2, Offset, Threshold_High, Threshold_Low, Suspend, Enabled, CalcRslt, ChkRslt, CalcRsltMax, CalcRsltMin, CalcRsltMean, Busy, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_PointMeasurement_V1_1.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00095
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Product name	Model numbers	Version
Device	Sensor Communications Unit	E3NW-ECT	Rev 1.03 or later
	Distributed Sensor Unit	E3NW-DS	-
	Contact-Type Smart Amplifier	E3NC-TA0	-
	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Enable	Execution	TRUE: Execute FALSE: Do not execute	BOOL	Depends on data type.	FALSE
Input1	Sensor Head 1 Input Value	The input value of the sensor head 1 is set.	DINT	Depends on data type.	0
Input2	Sensor Head 2 Iinput Value	The input value of the sensor head 2 is set.	DINT	Depends on data type.	0
Offset	Offset	The set value is added upon the calculation of *CalcRslt* (Calculation Result).	DINT	Depends on data type.	0
Threshold_High*1	Threshold (High)	Judges the value from the calculation result is in or out of the threshold range.	DINT	Depends on data type.	0
Threshold_Low*1	Threshold (Low)		DINT		0
Suspend	Suspension	Set to TRUE when inputting measurement data on which you want to disable the calculation during execution.	BOOL	Depends on data type.	FALSE

*1. If the set values are used for calculation at the task period when FALSE changed to TRUE on *Enable* (Execution) of this function block, the values which changed while *Enable* was TRUE are not reflected in the calculation.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Enabled	Calculation in Progress	TRUE: Calculation in progress FALSE: Calculation stopped	BOOL	Depends on data type.	—
CalcRslt	Calculation Result	Stores the result of thickness calculation based on the measurement values of sensors.	DINT	Depends on data type.	—
ChkRslt	Judgement Output	The judgement output turns to TRUE when the calculation results are not less than the threshold (Low) and not more than the threshold (High).	BOOL	Depends on data type.	—
CalcRsltMax	Calculation Result Maximum Value	Outputs the maximum value of *CalcRslt* under calculation.	DINT	Depends on data type.	—
CalcRsltMin	Calculation Result Minimum Value	Outputs the minimum value of *CalcRslt* under calculation.	DINT	Depends on data type.	—
CalcRslt-Mean	Calculation Result Mean Value	Outputs the mean value of *CalcRslt* under calculation.	DINT	Depends on data type.	—
Busy	Execution in Progress	Turns to TRUE while the processing is in execution; turns to FALSE while the processing is not in execution.	BOOL	Depends on data type.	—
Error	Error	Outputs TRUE if an error occurs.	BOOL	Depends on data type.	—
ErrorID	Error Code	Outputs the error code if an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the extended error code if an error occurs.	DWORD	*1	—

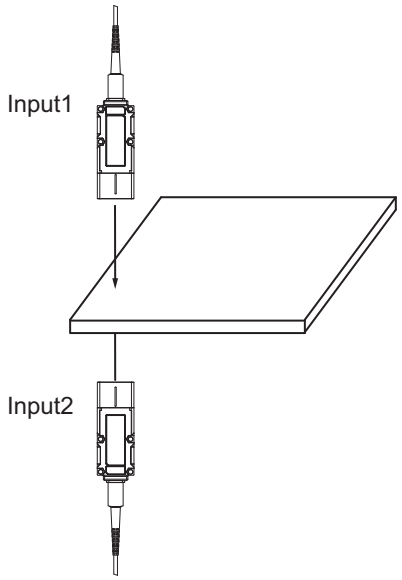
*1. Refer to the *Troubleshooting* on page 83.

Function

The PointMeasure_Thickness function block calculates the thickness from the measurement values (*Input1* and *Input2*) obtained from two measurement sensors, either type of the following sensors, outputting it on the calculation result.

- Sensor amplifier connected to the Sensor Communications Unit (E3NW-ECT)
- Confocal Fiber Displacement Sensor (ZW-8000/7000/5000)

Calculation result = (Input1 + Input2) + Offset



(a) As you set *Enable* (Execution) to TRUE, the thickness is calculated from *Input1* and *Input2* to be set for *CalcRslt* (Calculation Result). The arithmetic expression for the thickness is as follows:

Input1 + Input2 + Offset

When the value of *CalcRslt* is not less than the threshold (Low) and not more than the threshold (High), *ChkRslt* (Judgement Output) turns to TRUE. While *Enable* is TRUE, the calculation continues for each task period. As *Enable* turns to FALSE, the calculation is terminated, but *CalcRslt* and *ChkRslt* hold their values, which will be reset next time *Enable* changes to TRUE.

(b) Under calculation, *CalcRsltMax* (maximum valur of CalcRslt), *CalcRsltMin* (Calculation Result Minimum Value), and *CalcRsltMean* (Calculation Result Mean Value) are output. While *Enable* is TRUE, the calculation continues. As *Enable* turns to FALSE, the calculation is terminated, but *CalcRsltMax*, *CalcRsltMin*, and *CalcRsltMean* hold their values, which will be reset next time *Enable* changes to TRUE.

(c) The mean value is calculated from values that are input while *Enable* is TRUE. However, that the maximum number of data points (number of calls of a function block while *Enable* is TRUE) is 65535 and then this maximum number is exceeded, the mean value is not updated.

(d) Setting *Suspend* (Suspension) to TRUE allows the calculation to be suspended while the execution is in progress. This function is useful to remove the calculation results in the task periods where process data communications fail and the measurement data is not updated correctly. During suspension, *Enabled* (Calculation in Progress) turns to FALSE, where the number of data points used for mean value calculation is not updated. As you set *Suspend* to FALSE, calculation resumes.

(e) If an error occurs, *Error* turns to TRUE, where the processing of function blocks is interrupted. In addition, the error code is output to *ErrorID* and *ErrorIDEx*.

For details on the error codes, refer to *Troubleshooting* on page 83.

Timing Charts

Refer to the *Timing Charts* on page 52 for PointMeasure_Deviation.

Precautions for Use

- For input to this function block, consistently use either sensors connected to the E3NW or sensors of the ZW-8000/7000/5000. The combined use results in a failure to perform the correct operation because the unit is different from each other.
- Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.
- For the ZW-8000/7000/5000 series, confirm that the following pre-conditions are met before executing this function block.
  a) The ZW_CmdControl function block is unexecuted.
  b) *TASKSTAT_TX* which is the status of the target task for measurement slave is TRUE.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3CA2	0x1	The threshold (High) is smaller than the threshold (Low).	Specify the thresholds so that the threshold (High) will be larger than the threshold (Low).

Sample Programming

Program Description

For one second after the measurement start command, this program uses the measurement values of two types of sensors connected via EtherCAT as inputs to perform thickness calculation by PointMeasure_Thickness.

Preconditions

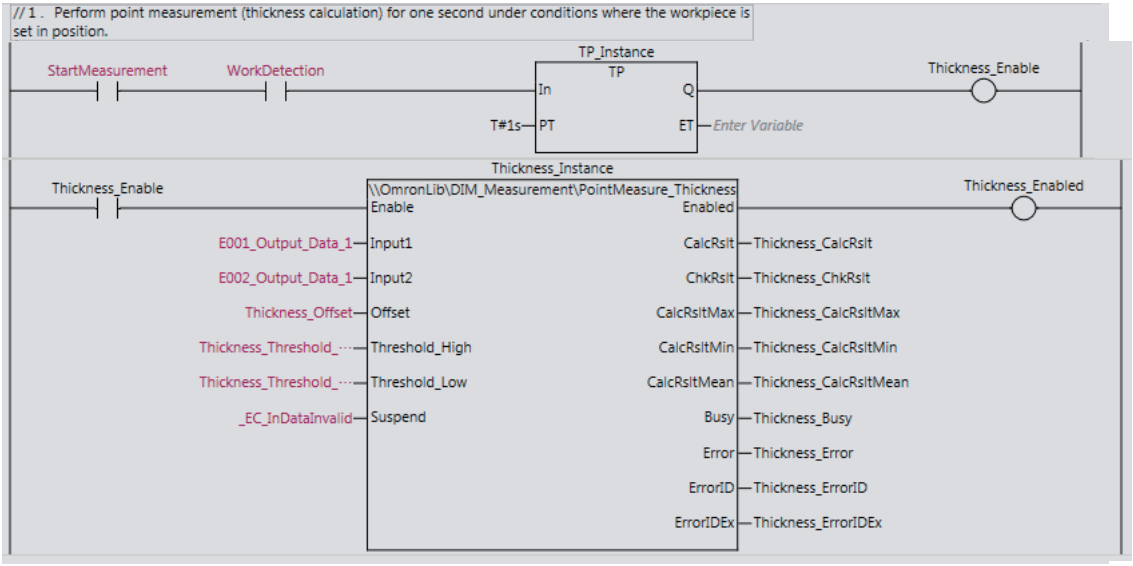
- Set the measurement item for each sensor to height.
- Select a combination of the workpiece and sensors that provides measurement values in the measurement range of the sensors.
- Perform the zero reset of each sensor by setting the master workpiece for measurement or the like mentioned in the example for PointMeasure_Deviation.

Main Variables

Name	Meaning	Data type	Default	Description
StartMeasurement	Start Thickness Calculation	BOOL	FALSE	Starts thickness calculation when the variable changes to TRUE.
WorkDetection	Workpiece in Position Status	BOOL	FALSE	TRUE: The workpiece for measurement is set in position. FALSE: The workpiece for measurement is not set in position.
E001_Output_Data_1	OUT1 Data in Output area I/O port on sensor head 1	DINT	---	OUT1 Data in the output area I/O port of the sensor head 1. Assign this variable to the *Input1* input variable of PointMeasure_Thickness.

Name	Meaning	Data type	Default	Description
E002_Output_Data_1	OUT1 Data in Output area I/O port on sensor head 2	DINT	---	OUT1 Data in the output area I/O port of the sensor head 2. Assign this variable to the *Input2* input variable of PointMeasure_Thickness.
Thickness _Offset	Set Offset	DINT	DINT#0	Sets the input value for *Offset*. Assign this variable to the *Offset* input variable of Thickness_Deviation.
Thickness_Threshold_High	Set Threshold (High)	DINT	DINT#0	Sets the threshold (High). Assign this variable to the *Threshold_High* input variable of PointMeasure_Thickness.
Thickness_Threshold_Low	Set Threshold (Low)	DINT	DINT#0	Sets the threshold (Low). Assign this variable to the *Threshold_Low* input variable of PointMeasure_Thickness.
_EC_InDataInvalid	Input Data Disabled	BOOL	---	A system-defined variable for EtherCAT communications. TRUE when the process data communications executed in the primary periodic task is not normal and the input value is not valid. Assign this variable to the *Suspend* input variable of PointMeasure_Thickness.
Thickness_CalcRslt	Calculation Result	DINT	---	Assign this variable to the *CalcRslt* output variable of PointMeasure_Thickness.
Curve_ChkRslt	Judgment Output	BOOL	---	Assign this variable to the *ChkRslt* output variable of PointMeasure_Thickness.
Thickness _CalcRsltMax	Calculation Result Maximum Value	DINT	---	Assign this variable to the *CalcRsltMax* output variable of PointMeasure_Thickness.
Thickness_CalcRsltMin	Calculation Result Minimum Value	DINT	---	Assign this variable to the *CalcRsltMin* output variable of PointMeasure_Thickness.
Thickness_CalcRsltMean	Calculation Result Mean Value	DINT	---	Assign this variable to the *CalcRsltMean* output variable of PointMeasure_Thickness.

Ladder Diagram

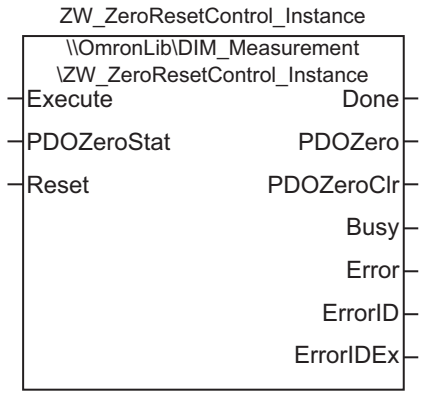


ST

```
// 1. Perform point measurement (thickness calculation) for one second under condi-
tion where the workpiece is set in position.
IF (StartMeasurement=TRUE) AND (WorkDetection=TRUE) THEN
   TP_Instance(In:=TRUE, PT:=T#1s, Q=>Thickness_Enable);
ELSE
   TP_Instance(In:=FALSE, Q=>Thickness_Enable);
END_IF;

// Thickness Calculation function block
PointMeasure_Thickness_Instance
(
Enable:=Thickness_Enable,
Input1:=E001_Output_Data_1,
Input2:=E002_Output_Data_1,
Offset:=Thickness_Offset,
Threshold_High:=Thickness_Threshold_High,
Threshold_Low:=Thickness_Threshold_Low,
Suspend:=_EC_InDataInvalid,
Enabled=>Thickness_Enabled,
CalcRslt=>Thickness_CalcRslt,
ChkRslt=>Thickness_ChkRslt,
CalcRsltMax=>Thickness_CalcRsltMax,
CalcRsltMin=>Thickness_CalcRsltMin,
CalcRsltMean=>Thickness_CalcRsltMean,
Busy=>Thickness_Busy,
Error=>Thickness_Error,
ErrorID=>Thickness_ErrorID,
ErrorIDEx=>Thickness_ErrorIDEx
);
```

ZW_ZeroResetControl

Through EtherCAT communications, the ZW_ZeroResetControl function block performs the Zero Reset and Clear Zero Reset functions for the specified task of the ZW-8000/7000/5000 series Confocal Fiber Displacement Sensor. The Zero Reset function allows registering the measured value as the reference value at any timing while the sensor is in the Run mode.

Function block name	Name	FB/ FUN	Graphics	ST expression
ZW_ZeroReset-Control	ZW Zero Reset Control	FB		ZW_ZeroRestControl_Instance (Execute, PDOZeroStat, Reset, Done, PDOZero, PDOZeroClr, Busy, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_Measurement_V1_1.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00096
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Name	Model numbers	Version
Device	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Hereinafter, the combination of the confocal fiber displacement sensor controllers and confocal fiber displacement sensor heads are called “displacement sensor”.

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	The function block is executed when this variable changes to TRUE. Re-executing the function block is prohibited.*1	BOOL	Depends on data type.	FALSE
PDOZeroStat	Zero Reset Status (Response area I/O Port)	TRUE when a Zero Reset is performed for the specified task of the displacement sensor and the sensor is in the Zero Reset state. FALSE when a Zero Reset is not performed for the specified task of the displacement sensor.	BOOL	Depends on data type.	FALSE
Reset*2	Execute Zero Reset	TRUE: The function block executes the Zero Reset. FALSE: The function block executes the Clear Zero Reset.	BOOL	Depends on data type.	FALSE

*1. Re-executing the function block means the following; when the *Execute* input variable changes to TRUE during the function block execution, the processing during execution is aborted, and the processing sequence is redone from the start.

*2. The set value of a task period when *Execute* changes to TRUE is used for operation.
The value is not refreshed even if it is changed while *Execute* is TRUE.

Output Variables

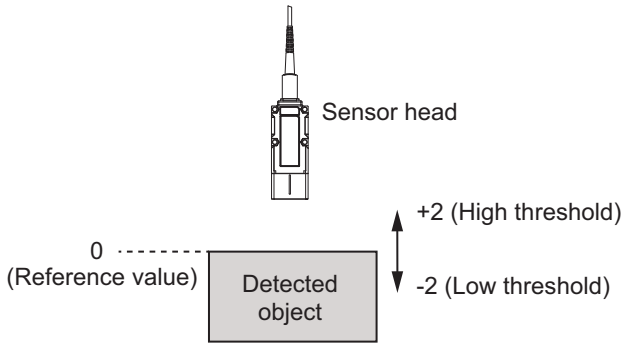
Variable	Name	Description	Data type	Valid range	Default
Done	Done	Changes to TRUE when the function block is completed.	BOOL	Depends on data type.	FALSE
PDOZero	Execute Zero Reset (Control area I/O Port)	Changes to TRUE when a Zero Reset command is sent for the specified task of the displacement sensor.	BOOL	Depends on data type.	FALSE
PDOZeroClr	Clear Zero Reset (Control area I/O Port)	Changes to TRUE when a Clear Zero Reset command is sent for the specified task of the displacement sensor.	BOOL	Depends on data type.	FALSE
Busy	Busy	TRUE while the function block is executed.	BOOL	Depends on data type.	FALSE
Error	Error	TRUE while there is an error.	BOOL	Depends on data type.	FALSE
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	16#0
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	-

*1. Refer to the *Troubleshooting* on page 91.

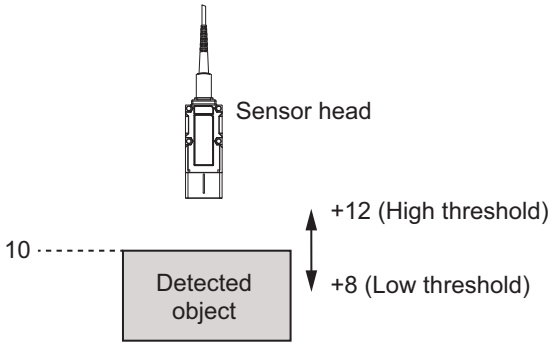
Function

Through EtherCAT communications, the ZW_ZeroResetControl function block performs the Zero Reset and Clear Zero Reset functions for the specified task of the ZW-8000/7000/5000 series Confocal Fiber Displacement Sensor. The Zero Reset function allows registering the measured value as the reference value at any timing while the sensor is in the Run mode. The Zero Reset is executed prior to measuring in the following examples.

Example1: The height of a detected object is registered as the reference value.
The measured values are expressed as deviation (tolerance).



Example 2: The measured values are expressed as the heights of detected objects. (10 is set as the offset value.)



The Clear Zero Reset clears the reference value set by the Zero Reset. The Clear Zero Reset is used for checking and adjusting the absolute distance between the sensor head and a workpiece.

For details of the Zero Reset and the Clear Zero Reset, refer to the *ZW-8000/7000/5000 Confocal Fiber Type Displacement Sensor User's Manual* (Cat. No. Z362).

To specify a slave and task for which the Zero Reset/the Clear Zero Reset is executed, you must set the slave's device variables (the command area I/O port and response area I/O port) for the function block's input and output variables.

(a) This function block performs the Zero Reset or Clear Zero Reset function for task 1, 2, 3 or 4 of the specified slave (displacement sensor) on the EtherCAT network. The following table shows allocation of the area I/O ports to the corresponding input/output variables. The function block sends a command to the displacement sensor by writing a value to the command area I/O port.

Input variable	PDOZeroStat
The corresponding response area I/O port	ZEROSTAT1_TX*1

*1. X is 1 to 4. Select the task number you carry out.

Output variable	PDOZero	PDOZeroClr
The corresponding the control area I/O port	ZERO1_TX*1	ZEROCLR1_TX*1

*1. X is 1 to 4. Select the task number you carry out.

(b) When the *Reset* input variable is TRUE, a Zero Reset is performed.

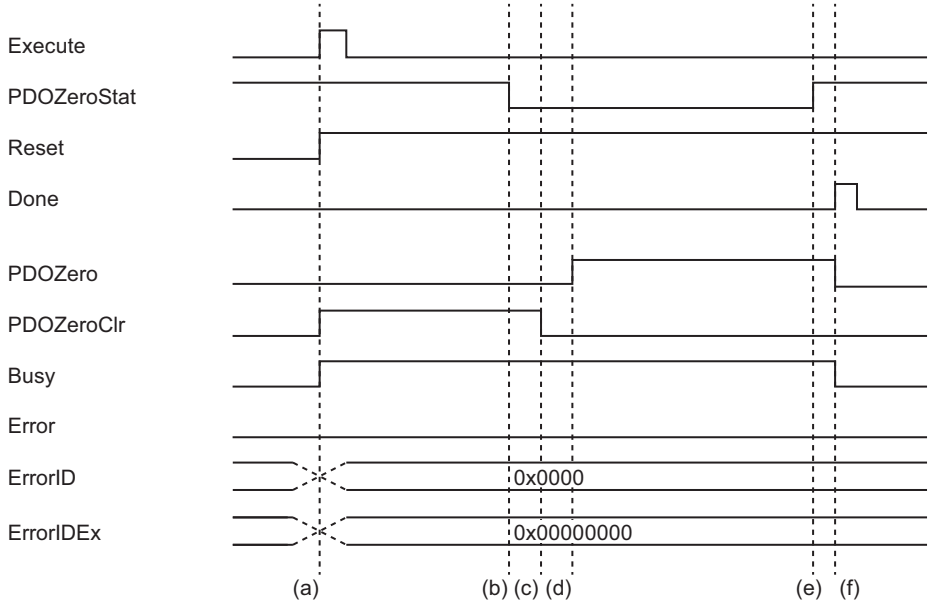
Timing Charts

The timing charts are shown below.

● Normal End (The Zero Reset Is Executed.)

(a) When *Execute* changes to TRUE, concurrently the function block changes *Busy* to TRUE.
When *PDOZeroStat* is FALSE, go to (d).
When *PDOZeroStat* is TRUE, the function block changes *PDOZeroClr* to TRUE. Go to (b).
(b) *PDOZeroStat* changes to FALSE.
(c) The function block turns *PDOZeroClr* to FALSE.
(d) The function block turns *PDOZero* to TRUE.
(e) *PDOZeroStat* changes to TRUE.
(f) The function block turns *Busy* and *PDOZero* to FALSE. *Done* changes to TRUE.

Even after the function block has been completed, *Done* is retained while *Execute* is being TRUE.

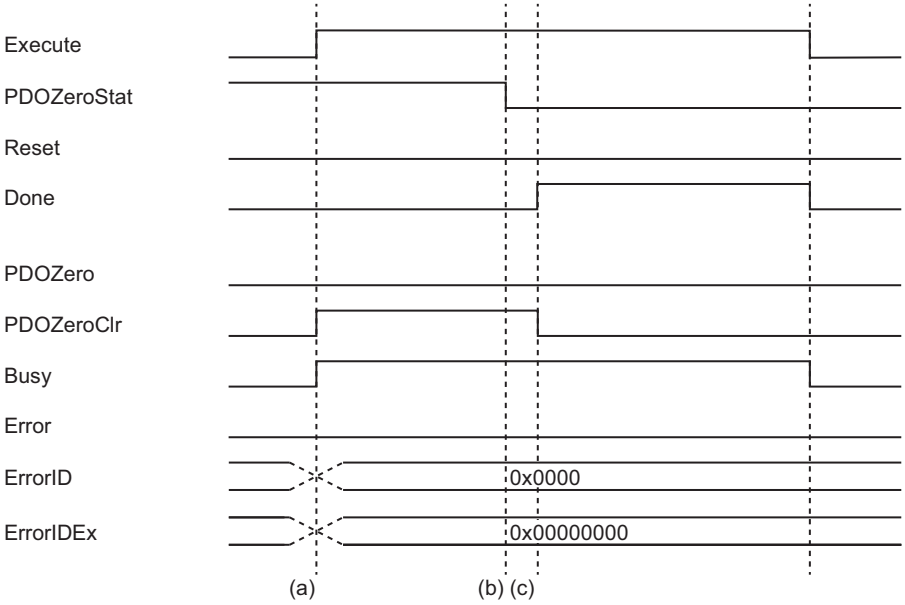


● Normal End (The Zero Reset Is Executed.)

(a) The function block is executed when *Execute* changes to TRUE.
When *PDOZeroStat* is TRUE, *Busy* and *PDOZeroClr* change to TRUE. Go to (b).
When *PDOZeroStat* is FALSE, go to (c).

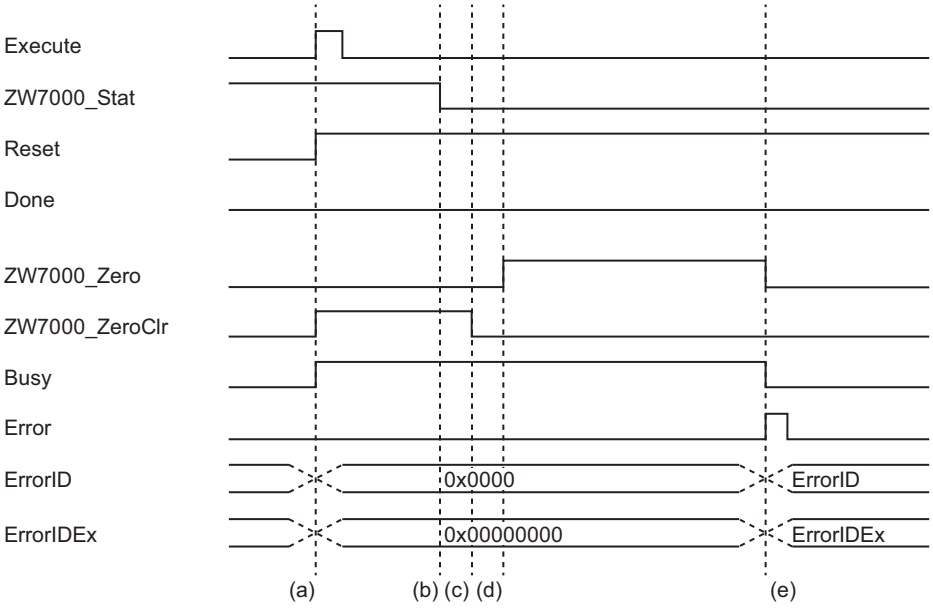
(b) *PDOZeroStat* changes to FALSE.

(c) The function block turns *Busy* and *PDOZeroClr* to FALSE. *Done* changes to TRUE.



● Error End (The Zero Reset Is Executed.)

Although the following (a) to (d) are the same with Normal End of the Zero Reset, the following case is an example that *PDOZeroStat* does not change to TRUE, and the timeout error occurs at (e). Once the timeout error occurs, *Busy* changes to FALSE. *Error* changes to TRUE. The values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code). Even after the command has been done, *Error* is retained while *Execute* is being TRUE. When *Execute* changes to FALSE, *Error* changes to FALSE accordingly. However, *ErrorID* and *ErrorIDEx* retain the values until the next function block is executed.



Precautions for Correct Use

- Always set the device variables of the same slave and same task in the input and output variables for the area I/O ports. If the device variables of different slaves and tasks are set, control may not be performed properly.
- During the function block execution, do not allow a device outside the function block to write to the control area I/O port that is set for the output variables. Otherwise, operation may not be performed properly.
- Before executing this function block, thoroughly read the manuals of the devices that are used, and ensure safety.
- Confirm that the following pre-conditions are met before executing this function block.
  a) The ZW_CmdControl function block is unexecuted.
  b) *TASKSTAT_TX* which is the status of the target task for measurement slave is TRUE.

Troubleshooting

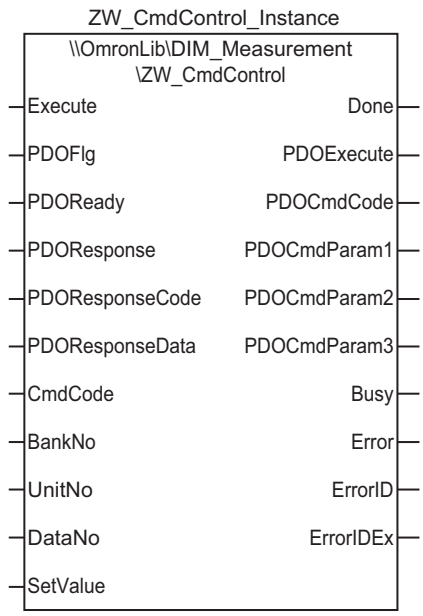
Error code	Expansion error code	Description	Corrective action
3CA3	0x1	Timeout error	Check whether there is an EtherCAT communication error or displacement sensor error. If so, correct the error and then execute the function block.

Sample Programming

Refer to *Sample Programming* on page 54 for the PointMeasureDeviation function block.

ZW_CmdControl

The ZW_CmdControl function block controls the commands for the ZW-8000/7000/5000 series Confocal Fiber Displacement Sensor that communicates with the NJ/NX/NY-Series Controller via EtherCAT. Use this function block to set or adjust the measurement conditions before using the displacement sensor, or to save the set and adjusted data in the displacement sensor.

Function block name	Name	FB/ FUN	Graphics	ST expression
ZW_CmdControl	ZW Command Control	FB		ZW_CmdControl_Instance (Execute PDOFlg, PDOReady, PDOResponse, PDOResponseCode, PDOResponseData, CmdCode, BankNo, UnitNo, DataNo, SetValue, Done, PDOExecute, PDOCmdCode, PDOCmdParam1, PDOCmdParam2, PDOCmdParam3, Busy, Error, ErrorID ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_Measurement_V1_1.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00097
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Name	Model numbers	Version
Device	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Hereinafter, the combination of the confocal fiber displacement sensor controllers (ZW-8000/7000/5000) and confocal fiber displacement sensor heads (ZW-□□□□) are called “displacement sensor”.

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	The function block is executed when this variable changes to TRUE. Re-executing the function block is prohibited.*1	BOOL	Depends on data type.	FALSE
PDOFlg	Control Command Completed (Responce Area I/O Port)	Changes to TRUE when control command execution in the displacement sensor is completed.	BOOL	Depends on data type.	FALSE
PDOReady	Ready (Responce Area I/O Port)	Changes to TRUE when the displacement sensor is ready to execute the control command or measurement synchronization.	BOOL	Depends on data type.	FALSE
PDOResponse	Command Code (Responce Area I/O Port)	The code of the executed command is returned.	DWORD	Depends on data type.	16#0
PDOResponse-Code	Response Code (Responce Area I/O Port)	Stores the response code of the executed command.	DWORD	Depends on data type.	16#0
PDOResponse-Data	Response Data (Responce Area I/O Port)	Stores the response data of the executed command.	DINT	Depends on data type.	0
CmdCode*2	Select Command	Select the command to execute. 0: Save data 1: Calibrate sensor head 2: Restart 3: Set current bank 4: Acquire processing unit data 5: Set processing unit data 6: Acquire system data 7: Set system data	UINT	0 to 7	0
BankNo*2	Bank Number	Specify the bank number to set the current bank.	UINT	Refer to (a) of the *Function*.	1
UnitNo*2	Unit Number	Specify the unit number to acquire/set processing unit data.	UINT	Refer to (a) of the *Function*.	0
DataNo*2	Data Number	Specify the data number to acquire/set processing unit data or system data.	UINT	Refer to (a) of the *Function*.	0

Variable	Name	Description	Data type	Valid range	Default
SetValue*2	Setup Data	Specify the data that is set when the Set processing unit data or Set system data command is executed.	DINT	Refer to (a) of the *Function*.	0

*1. Re-executing the function block means the following; when the *Execute* input variable changes to TRUE during the function block execution, the processing during execution is aborted, and the processing sequence is re-done from the start.

*2. The set value of a task period when *Execute* changes to TRUE is used for operation.
The value is not refreshed even if it is changed while *Execute* is TRUE.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	Changes to TRUE when function block execution is completed.	BOOL	Depends on data type.	FALSE
PDOExecute	Execute Command (Command Area I/O Port)	The value set for EXE (control command execution) of the command area I/O port of the displacement sensor is output.	BOOL	Depends on data type.	FALSE
PDOCmdCode	Command Code (Command Area I/O Port)	The value set for Command (command code) of the command area I/O port of the displacement sensor is output.	DWORD	Depends on data type.	16#0
PDOCmdParam1	Command Parameter 1 (Command Area I/O Port)	The value set for Command Parameter 1 of the command area I/O port of the displacement sensor is output.	UINT	Depends on data type.	UINT#0
PDOCmdParam2	Command Parameter 2 (Command Area I/O Port)	The value set for Command Parameter 2 of the command area I/O port of the displacement sensor is output.	UINT	Depends on data type.	UINT#0
PDOCmdParam3	Command Parameter 3 (Command Area I/O Port)	The value set for Command Parameter 3 of the command area I/O port of the displacement sensor is output.	DINT	Depends on data type.	DINT#0
Busy	Busy	Changes to TRUE while the function block is being executed.	BOOL	Depends on data type.	FALSE
Error	Error	TRUE while there is an error.	BOOL	Depends on data type.	FALSE
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	16#0
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	WORD	*1	16#0

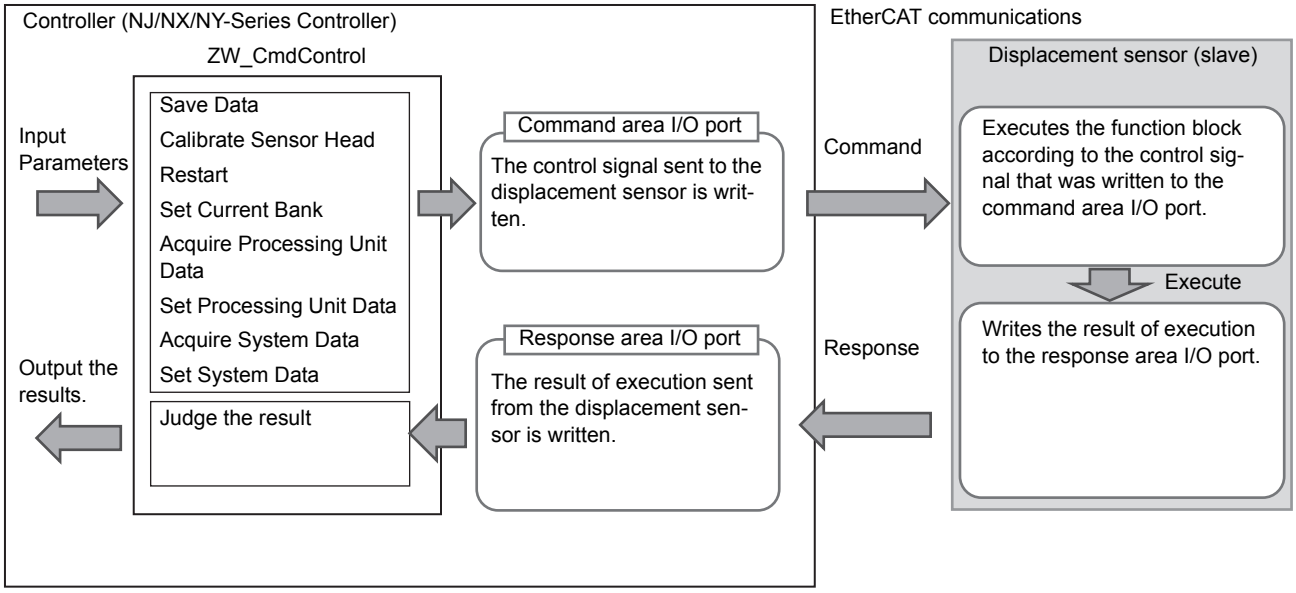
*1. Refer to the *Troubleshooting* on page 101.

Function

The ZW_CmdControl function block controls the commands for the ZW-8000/7000/5000 series Confocal Fiber Displacement Sensor that communicates with the NJ/NX/NY-Series Controller via EtherCAT. Use this function block to set or adjust the measurement conditions before using the displacement sensor, or to save the set and adjusted data in the displacement sensor.

This function block controls commands (listed in the *ZW-8000/7000/5000 Confocal Fiber Type Displacement Sensor User's Manual* (Cat. No. Z362)) to the specified slave on the EtherCAT network. To specify the slave to control, you must set the slave's device variables (the command area I/O port and response area I/O port) for the function block's input/output variables.

When a command is set in the input parameter and the function block is executed, a control signal is output. The function block sends a command to the displacement sensor by writing the control signal to the command area I/O port. The function block also determines the result of command execution of the slave from the value written to the response area I/O port. The function block outputs the judgment result.



(a) This function block outputs the signals of commands (listed in the *ZW-8000/7000/5000 Confocal Fiber Type Displacement Sensor User's Manual* (Cat. No. Z362)) that can be used on the EtherCAT network.
As shown in the following table, when the function block is executed with the values set in the *CmdCode, BankNo, UnitNo, DataNo,* and *SetValue* input variables, it outputs the values that are written to the command area I/O port of the displacement sensor. A command is sent to the displacement sensor when these values are written to the command area I/O port. For proper operation of the function block, you must allocate the command area I/O ports to the corresponding output variables.

Command	Input variable					The command area I/O ports (lower) corresponding to the output variables (upper)				
	Cmd-Code	BankNo	UnitNo	DataNo	Set-Value	PDOExecute	PDOCmd	PDOCmdParam1	PDOCmdParam2	PDOCmdParam3
						EXE	Command	Command Parameter1	Command Parameter1	Command Parameter1
Save Data	0		-	-	—	The function block is executed when FALSE changes to TRUE.	0x00103011	—	—	—
Calibrate Sensor Head	1		-	-	—	↑	0x0010E000	—	—	—
Restart	2		-	-	—	↑	0x0010F010	—	—	—
Set Current Bank	3	Bank number	-	-	—	↑	0x00308000	Bank number -1	—	—
Acquire Processing Unit Data	4	-	Unit number	Data number	—	↑	0x00401000	Unit number	Data number	—
Set Processing Unit Data	5	-	Unit number	Data number	Setup data	↑	0x00501000	Unit number	Data number	Setup data
Acquire System Data	6	-	-	Data number	—	↑	0x00404000	Data number	-	-
Set System Data	7	-	-	Data number	Setup data	↑	0x00504000	Data number	-	Setup data

Refer to the items in the Processing Item Data List and the System Data List of the user's manual on the individual displacement sensor, for the values, meanings and setting ranges of the bank number, unit number, data number, and setup data in the above table.

(b) After sending a command to the displacement sensor, the function block determines the result of command execution. The function block outputs the judgment result on whether command execution ended normally in the displacement sensor, and outputs the error code and expansion error code if an error end occurred. For proper operation of the function block, you must allocate the response area I/O ports in the following table to the corresponding output variables.

Response Area I/O Port	FLG	READY	Response	Response Code	Response Data
Corresponding input variables	PDOFlg	PDOReady	PDOResponse	PDOResponseCode	PDOResponseData

Timing Charts

The timing charts are shown below.

● Timing Chart in a Normal State (Except When the Select Command Is Restarted.)

(a) When *Execute* changes to TRUE, concurrently *Busy* changes to TRUE. The updated values with the input variables will be output to *PDOCmd*, *PDOCmdParam1*, *PDOCmdParam2*, and *PDOCmd-Param3*. *PDOExecute* turns TRUE from FALSE. The command is sent to the displacement sensor.

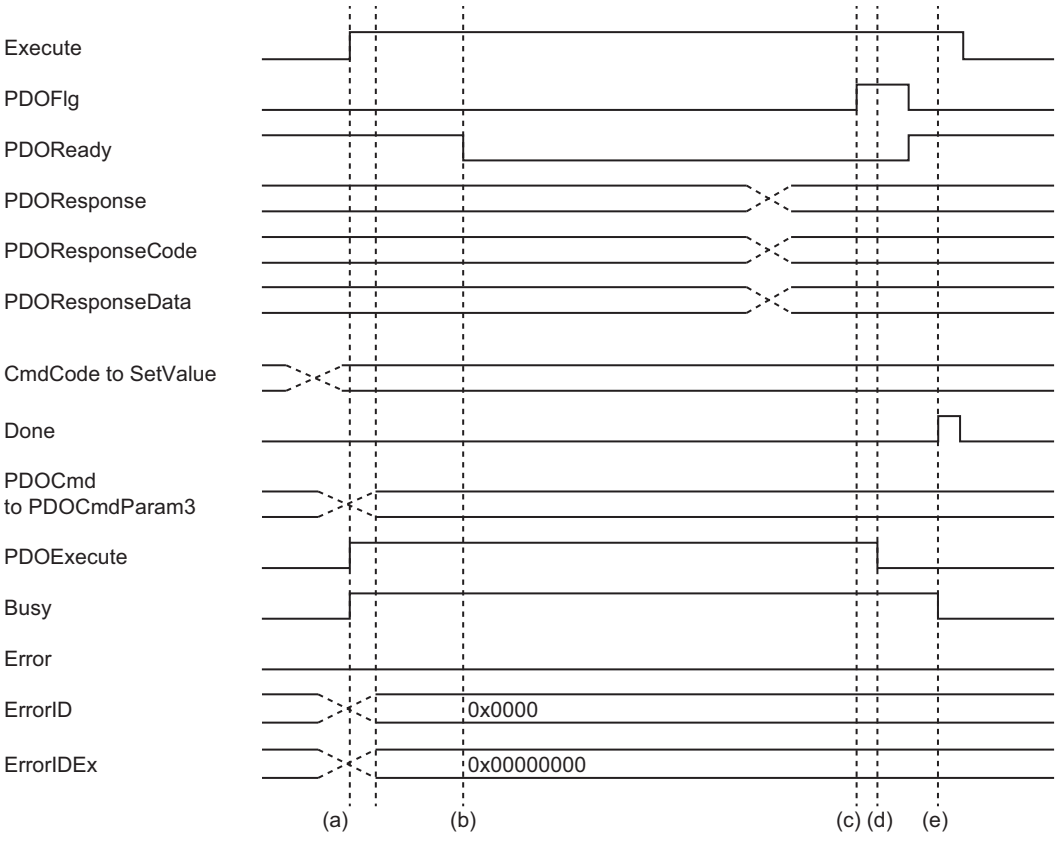
(b) Once the displacement sensor receives the command, *PDOReady* changes to FALSE.

(c) Once the displacement sensor has completed the processing, *PDOFlg* changes to TRUE. Based on the values of *PDOResponse*, *PDOResponseCode*, and *PDOResponseData* at that time, the execution result with the displacement sensor is judged. Normal end is confirmed.

(d) *PDOExecute* changes to FALSE.

(e) After *PDOReady* changes to TRUE and *PDOFlg* changes to FALSE, *Done* changes to TRUE. *Busy* changes to FALSE.

Even after the function block has been done, *Done* is retained while *Execute* is being TRUE.



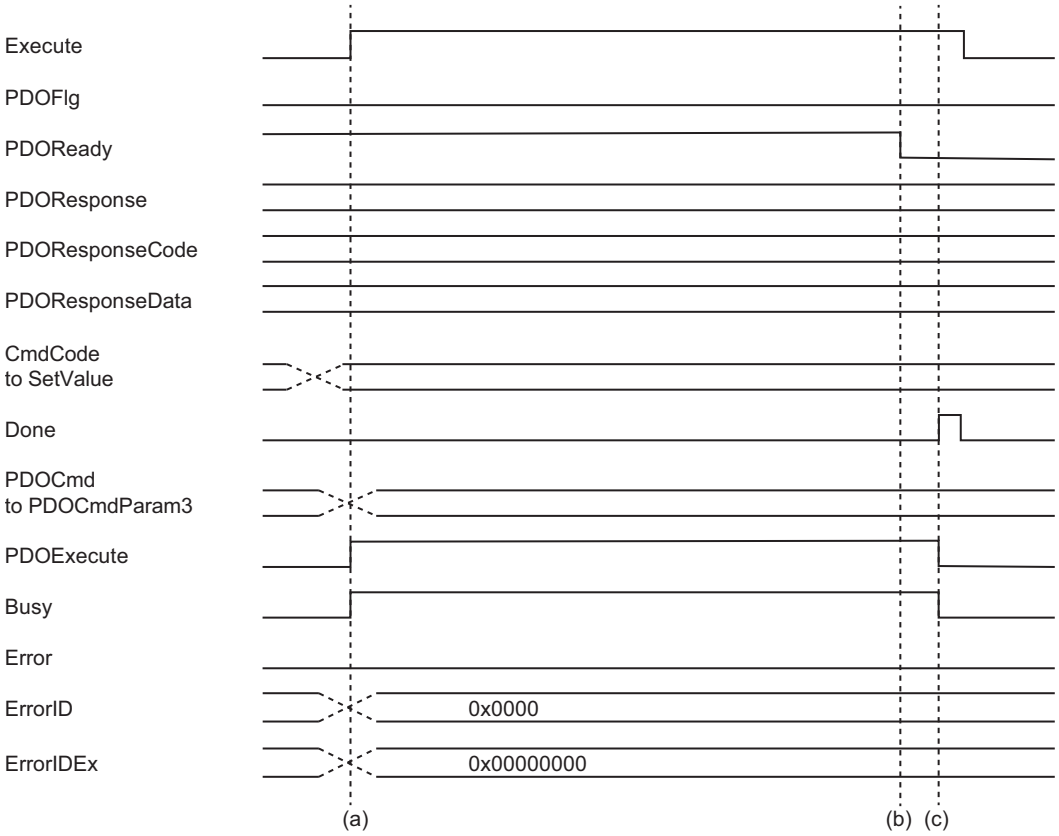
● Timing Chart in a Normal State (When the Select Command Is Restarted.)

(a) When *Execute* changes to TRUE, concurrently *Busy* changes to TRUE. The updated values with input variables are output to *PDOCmd*, *PDOCmdParam1*, *PDOCmdParam2*, and *PDOCmdParam3*.

PDOExecute turns TRUE from FALSE. The command is sent to the displacement sensor.

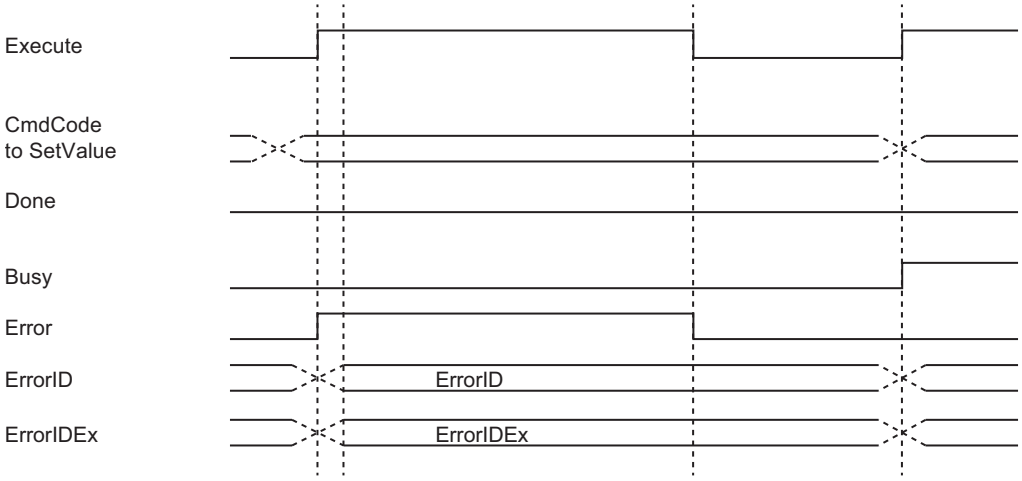
(b) Once the displacement sensor receives the command, *PDOReady* changes to FALSE.

(c) *PDOExecute* changes to FALSE, *Done* changes to TRUE, and *Busy* changes to FALSE.



● Timing Chart When an Error Occurs (Startup Error)

If an out-of-range value is set to *CmdCode* when the function block is executed, a startup error occurs. *Error* changes to TRUE. The values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code). When *Execute* changes to FALSE, *Error* changes to FALSE accordingly. However, *ErrorID* and *ErrorIDEx* retain the values until the next function block is executed.



● Timing Chart When an Error Occurs (When an Error Is Judged Based on the Response of the Response Area I/O.)

Except when an out-of-range value is set to *CmdCode*, timing chart at error is as follows:

(a) The same as normal end.

(b) The same as normal end.

(c) The same as normal end.

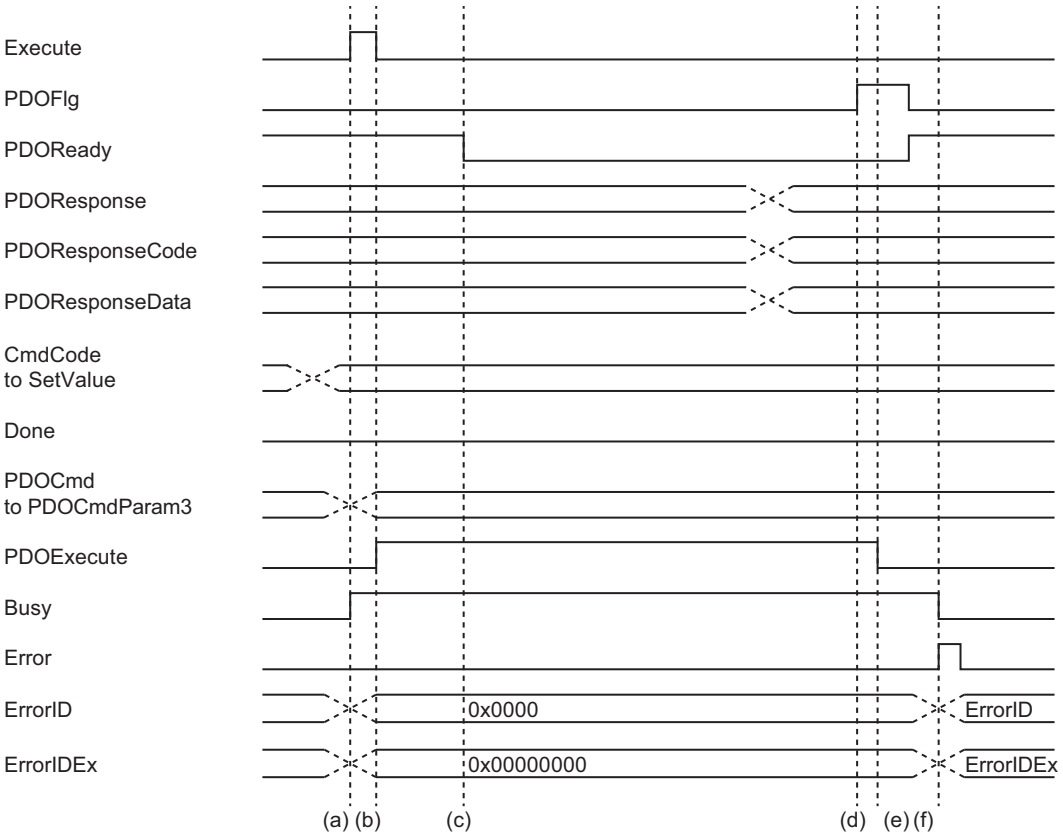
(d) When the displacement sensor has completed the processing, *PDOFlg* changes to TRUE. Based on the values of *PDOResponse*, *PDOResponseCode*, and *PDOResponseData* at that time, the execution result in ZW-8000/7000/5000 is judged and then the error end is confirmed.

(e) *PDOExecute* changes to FALSE.

(f) After *PDOReady* changes to TRUE and *PDOFlg* changes to FALSE, *Busy* changes to FALSE. *Error* changes to TRUE. The values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code).

If error occurs when this function block is executed, *Error* changes to TRUE. The values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code). Even after this function block has been completed, *Error* retains output while *Execute* is being TRUE.

When *Execute* changes to FALSE, *Error* changes to FALSE accordingly. However, *ErrorID* and *ErrorIDEx* retain the values until the next command is executed.



Precautions for Correct Use

- Always set the device variables of the same slave and same task in the input and output variables for the area I/O ports. If the device variables of different slaves and tasks are set, control may not be performed properly.
- During the function block execution, do not allow a device outside the function block to write to the control area I/O port that is set for the output variables. Otherwise, control may not be performed properly.
- Before executing this function block, thoroughly read the manuals of the devices that are used, and ensure safety.

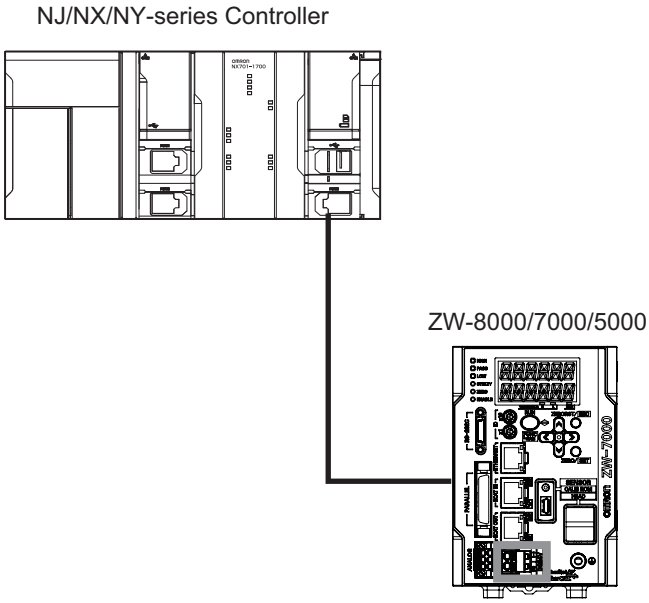
Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3CA4	0x1	The value of CmdCode is out of range.	Set the correct value in *CmdCode*.
	0x2	BankNo, UnitNo, DataNo, or SetValue is out of range.	Check the set value of *BankNo, UnitNo, DataNo*, and *SetValue*, and set the correct value.
	0x3	Processing error	Check if there is an EtherCAT communication error or displacement sensor error. If there is an error, correct it and then execute the function block.
	0x4	Mode error	After setting the Run mode, execute the command.
	0x5	Timeout error	Check if there is an EtherCAT communication error or displacement sensor error. If there is an error, correct it and then execute the function block.
	0x6	Error response is received.	Check if there is an EtherCAT communication error or displacement sensor error. If there is an error, correct it and then execute the function block.
	0xB	The function block cannot operate because the command is not ready for execution.	Make sure that no command is being executed from the function block in another instance or for directly setting PDOs. Then, execute the function block again.

Sample Programming

Program Description

In the following system in which a ZW-8000/7000/5000 and the NJ/NX/NY-series Controller are connected via EtherCAT, this program uses ZW_CmdControl to calibrate the sensor head, and set and save parameters in task 2 of bank 2.



The table below shows the processing flow.

No.	Outline	Description
1.	Sensor head calibration	Calibrates the sensor head if not yet calibrated.
2.	Bank change	Changes the current bank to bank 2.
3.	Median filter mode change	Changes the median filter mode for task 2 to 9 times.
4.	Saving	Saves the parameters set for the displacement sensor controller.

Preconditions

- Set field bus parameters for the displacement sensor so that it can communicate via EtherCAT. For details, refer to the *ZW-8000/7000/5000 Confocal Fiber Type DisplacementSensor User's Manual for Communications Settings* (Cat. No. Z363).
- Create device variables for the command area I/O and response area I/O ports of the displacement sensor with which to communicate and use external references to the variables in the program. Refer to the *Sysmac Studio Version 1 Operation Manual* (Cat. No. W504) for details on how to create device variables.

Main Variables

Name	Meaning	Data type	Default	Description
E001_FLG	Command Done in Response area I/O port	BOOL	---	Command Done in the response area I/O port of the displacement sensor. Assign this variable to the *PDOFlg* input variable of ZW_CmdControl.
E001_READY	Signal Input Ready Status in Response area I/O port	BOOL	---	Signal Input Ready Status in the response area I/O port of the displacement sensor. Assign this variable to the *PDOReady* input variable of ZW_CmdControl.
E001_Response	Command Echo Back in Response area I/O port	BOOL	---	Command Echo Back in the response area I/O port of the displacement sensor. Assign this variable to the *PDOResponse* input variable of ZW_CmdControl.
E001_Response_Code	Response Code in Response area I/O port	DWORD	---	Response Code in the response area I/O port of the displacement sensor. Assign this variable to the *PDOResponseCode* input variable of ZW_CmdControl.
E001_Response_Data1	Response Data in Response area I/O port	DINT	---	Response Data in the response area I/O port of the displacement sensor. Assign this variable to the *PDOResponseData* input variable of ZW_CmdControl.
E001_EXE	Execute Command in Command area I/O port	BOOL	FALSE	Execute Command in the command area I/O port of the displacement sensor. Assign this variable to the *PDOExecute* output variable of ZW_CmdControl.
E001_Command	Command Code in Command area I/O port	DWORD	16#0	Command Code in the command area I/O port of the displacement sensor. Assign this variable to the *PDOCmdCode* output variable of ZW_CmdControl.
E001_Command_Parameter1	Command Parameter 1 in Command area I/O port	UINT	UINT#0	Command Parameter 1 in the command area I/O port of the displacement sensor. Assign this variable to the *PDOCmdParam1* output variable of ZW_CmdControl.
E001_Command_Parameter2	Command Parameter 2 in Command area I/O port	UINT	UINT#0	Command Parameter 2 in the command area I/O port of the displacement sensor. Assign this variable to the *PDOCmdParam2* output variable of ZW_CmdControl.
E001_Command_Parameter3	Command Parameter 3 in Command area I/O port	DINT	DINT#0	Command Parameter 3 in the command area I/O port of the displacement sensor. Assign this variable to the *PDOCmdParam3* output variable of ZW_CmdControl.

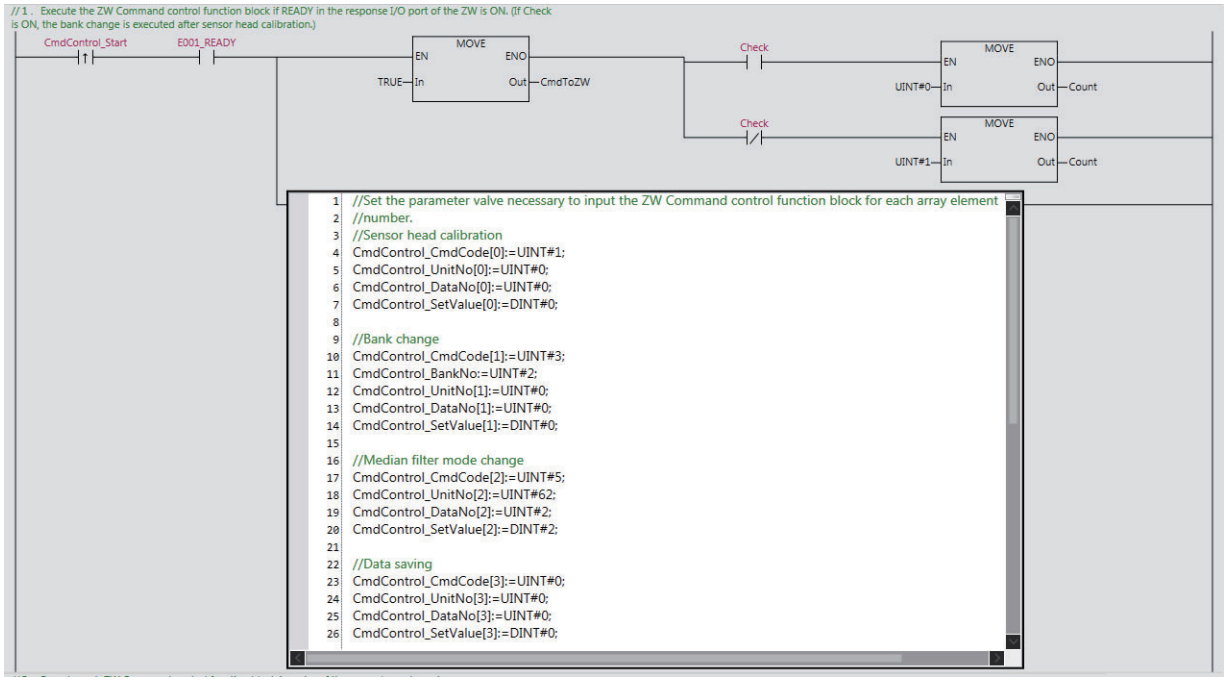
Name	Meaning	Data type	Default	Description
CmdControl_Start	Start ZW Command Control	BOOL	FALSE	Starts the ZW Command Control for the displacement sensor when the variable changes to TRUE.
Check	Calibrate Sensor	BOOL	FALSE	Calibrates the sensor after the start of the ZW Command Control when the variable is TRUE.
CmdControl_CmdCode[]	Command Code	ARRAY[0..3] OF DWORD	16#0	Sets the following command code. Value of the array element number 0 to 3 : Command code that matches the processing flow number 1 to 4
CmdControl_BankNo	Bank Number	UINT	UINT#0	Sets the bank number for which to set parameters.
CmdControl_UnitNo[]	Unit Number	ARRAY[0..3] OF UINT	16#0	Sets the following unit number. Value of the array element number 0, 1, and 3 : 0 Value of the array element number 2 : 42+20*(2-1)=62 *1
CmdControl_DataNo[]	Data Number	ARRAY[0..3] OF UINT	16#0	Sets the following data number. Value of the array element number 0, 1, and 3 : 0 Value of the array element number 2 : 2 *2
CmdControl_SetValue[]	Set Value	ARRAY[0..3] OF DINT	16#0	Sets the following data number. Value of the array element number 0, 1, and 3 : 0 Value of the array element number 2 : 2 *3

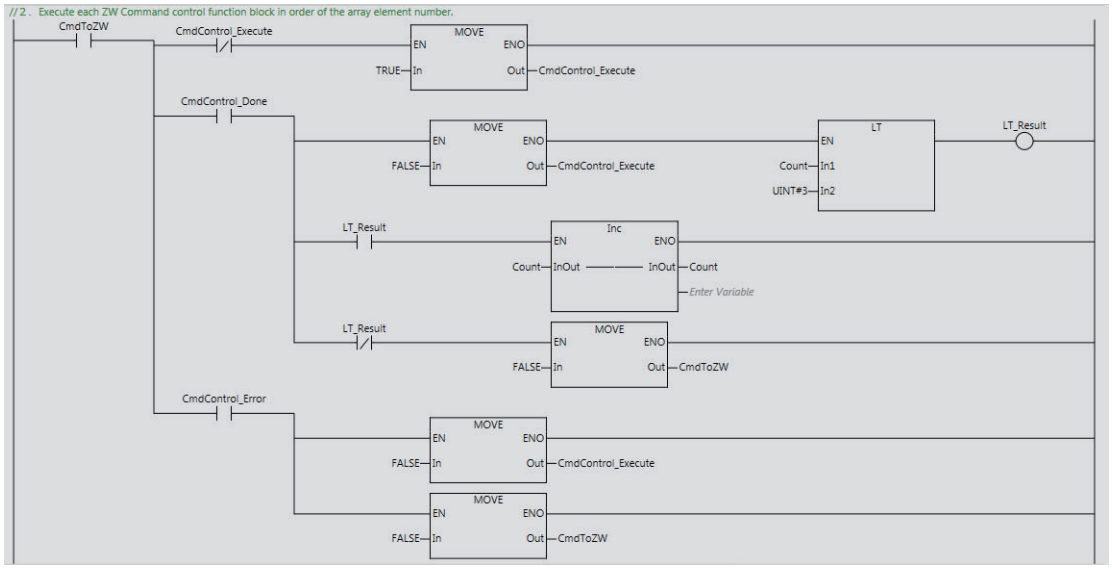
*1. An example for calculating the unit number for the median of task 2 using the ZW-8000/7000/5000. For details, refer to the displacement sensor user's manual.

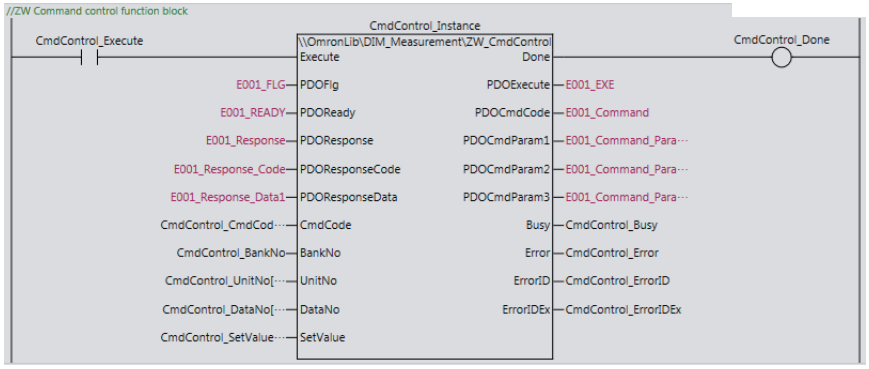
*2. The data number for the median filter of the ZW-8000/7000/5000. For details, refer to the displacement sensor user's manual.

*3. The value used when setting the median filter mode of the ZW-8000/7000/5000 to 9 times. For details, refer to the displacement sensor user's manual.

Ladder Diagram







● Code of Inline ST:

```
// Set the parameter value necessary to input the ZW Command Control function block
for each array element number.
// Sensor head calibration
CmdControl_CmdCode[0]:=UINT#1;
CmdControl_UnitNo[0]:=UINT#0;
CmdControl_DataNo[0]:=UINT#0;
CmdControl_SetValue[0]:=DINT#0;

// Bank change
CmdControl_CmdCode[1]:=UINT#3;
CmdControl_BankNo:=UINT#2;
CmdControl_UnitNo[1]:=UINT#0;
CmdControl_DataNo[1]:=UINT#0;
CmdControl_SetValue[1]:=DINT#0;

// Median filter mode change
CmdControl_CmdCode[2]:=UINT#5;
CmdControl_UnitNo[2]:=UINT#62;
CmdControl_DataNo[2]:=UINT#2;
CmdControl_SetValue[2]:=DINT#2;

// Data save processing
CmdControl_CmdCode[3]:=UINT#0;
CmdControl_UnitNo[3]:=UINT#0;
CmdControl_DataNo[3]:=UINT#0;
CmdControl_SetValue[3]:=DINT#0;
```

ST

```
// 1. Execute the ZW Command Control function block if READY in the response I/O port
of the ZW is ON. (If Check is ON, the bank change is executed after sensor head cal-
ibration.)
R_TRIG_instance(Clk:=CmdControl_Start, Q=>Start_Result);
IF (Start_Result=TRUE) AND (E001_READY=TRUE) THEN
   IF (Check=TRUE) THEN
      Count:=UINT#0;
   ELSE
      Count:=UINT#1;
   END_IF;
   CmdToZW:=TRUE;
   // Set the parameter value necessary to input the ZW Command Control function
block for each array element number.
   // Sensor head calibration
   CmdControl_CmdCode[0]:=UINT#1;
   CmdControl_UnitNo[0]:=UINT#0;
   CmdControl_DataNo[0]:=UINT#0;
   CmdControl_SetValue[0]:=DINT#0;

   // Bank change
   CmdControl_CmdCode[1]:=UINT#3;
   CmdControl_BankNo:=UINT#2;
   CmdControl_UnitNo[1]:=UINT#0;
   CmdControl_DataNo[1]:=UINT#0;
   CmdControl_SetValue[1]:=DINT#0;

   // Median filter mode change
   CmdControl_CmdCode[2]:=UINT#5;
   CmdControl_UnitNo[2]:=UINT#62;
   CmdControl_DataNo[2]:=UINT#2;
   CmdControl_SetValue[2]:=DINT#2;

   // Data save processing
   CmdControl_CmdCode[3]:=UINT#0;
```

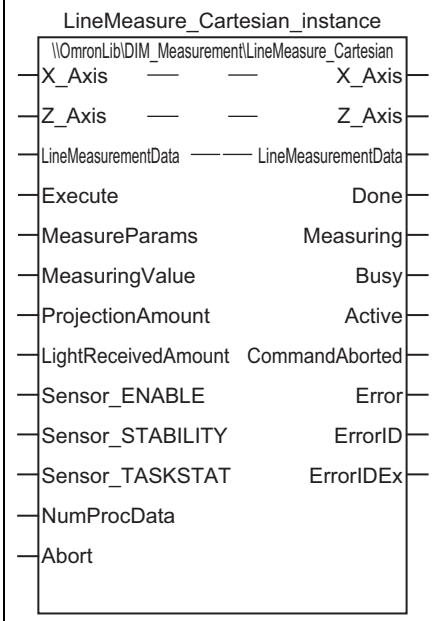
```
   CmdControl_UnitNo[3]:=UINT#0;
   CmdControl_DataNo[3]:=UINT#0;
   CmdControl_SetValue[3]:=DINT#0;
END_IF;

// 2. Execute each ZW Command Control function block in order of the array element
number.
IF (CmdToZW=TRUE) THEN
   IF NOT(CmdControl_Execute) THEN
      CmdControl_Execute:=TRUE;
   END_IF;
   IF (CmdControl_Done=TRUE) THEN
      CmdControl_Execute:=FALSE;
      IF (Count<UINT#3) THEN
         Inc(Count);
      ELSE
         CmdToZW:=FALSE;
      END_IF;
   ELSIF (CmdControl_Error=TRUE) THEN
      CmdControl_Execute:=FALSE;
      CmdToZW:=FALSE;
   END_IF;
END_IF;

// ZW Command Control function block
ZW_CmdControl_Instance
(Execute:=CmdControl_Execute,
PDOFlg:=E001_FLG,
PDOReady:=E001_READY,
PDOResponse:=E001_Response,
PDOResponseCode:=E001_Response_Code,
PDOResponseData:=E001_Response_Data1,
CmdCode:=CmdControl_CmdCode[Count],
BankNo:=CmdControl_BankNo,
UnitNo:=CmdControl_UnitNo[Count],
DataNo:=CmdControl_DataNo[Count],
SetValue:=CmdControl_SetValue[Count],
Done=>CmdControl_Done,
PDOExecute=>E001_EXE,
PDOCmdCode=>E001_Command,
PDOCmdParam1=>E001_Command_Parameter1,
PDOCmdParam2=>E001_Command_Parameter2,
PDOCmdParam3=>E001_Command_Parameter3,
Busy=>CmdControl_Busy,
Error=>CmdControl_Error,
ErrorID=>CmdControl_ErrorID,
ErrorIDEx=>CmdControl_ErrorIDEx
);
```

LineMeasure_Cartesian

The LineMeasure_Cartesian function block measures the height of measurement surfaces while moving a single axis with a displacement sensor in X and Z directions.

From the measurement result, this function block creates the line measurement data (sLineMeasurementData).

You can select from the following two methods to control the single axis during measurement: Surface Search and Tracer Control.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
LineMeasure_ Cartesian	Line Measurement with Cartesian Coordinate System (Surface Search/Tracer Control)	FB		LineMeasure_Cartesian_instance(X_Axis, Z_Axis, LineMeasurementData, Execute, MeasureParams, MeasuringValue, ProjectionAmount, LightReceivedAmount, Sensor_ENABLE, Sensor_STABILITY, Sensor_TASKSTAT, NumProcData, Abort, Done, Measuring, Busy, Active, CommandAborted, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V1_X.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00147
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Product name	Model numbers	Version
Device	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
Measure Params*1	Measurement parameter	Set the measurement parameter.	sMeasureP-arams	—	—
MeasuringValue	Displacement Sensor Measurement Value	Assign the measurement value output from the displacement sensor. The unit is [nm].	DINT	Depends on data type.	0
Projection Amount	Displacement Sensor Amount of Emitted Light	Assign the amount of emitted light output from the displacement sensor.	UINT	Depends on data type.	0
LightReceived Amount	Displacement Sensor Amount of Light Received	Assign the amount of emitted light output from the displacement sensor.	UINT	Depends on data type.	0
Sensor_ ENABLE	Displacement Sensor ENABLE Output	Assign the ENABLE output from the displacement sensor.	BOOL	TRUE or FALSE	FALSE
Sensor _STABILITY	Displacement Sensor STABILITY Output	Assign the STABILITY output from the displacement sensor.	BOOL	TRUE or FALSE	FALSE
Sensor _TASKSTAT	Displacement Sensor TASKSTAT Output	Assign the TASKSTAT output from the displacement sensor.	BOOL	TRUE or FALSE	FALSE
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a task period in the segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
Measuring	Measuring	TRUE when measurement is in progress. FALSE when data is outside the measurement range or when re-positioning is in progress.	BOOL	TRUE or FALSE	—
Busy	Executing	Changes to TRUE when the function block is acknowledged.	BOOL	TRUE or FALSE	—
Active	Controlling	TRUE when control is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 137.

In-Out Variables

Variable	Name	Description	Data type	Valid range
X_Axis	X Axis	Specify the X axis.*1 Specify the "Servo Axis".	_sAXIS_REF	—
Z_Axis	Z Axis	Specify the Z axis.*1 Specify the "Servo Axis".	_sAXIS_REF	—
LineMeasurement Data	Line measurement data	The line measurement data is recorded.	sLineMeasurementData	—

*1. Specify a user-defined Axis Variable that was created in the Axis Basic Settings of the Sysmac Studio (default: *MC_Axis****) or a system-defined axis variable name (*_MC_AX[*]*, *_MC1_AX[*]*, or *_MC2_AX[*])*.

Members of Structure sMeasureParams

Member	Name	Data type	Valid range	Description
CtrlType	Control Type	USINT	0, 1	0: Surface Search 1: Tracer Control
X_Prepare Position	X-axis Preparation Position	LREAL	Negative number, positive number, or 0	Set the measurement preparation position. The unit is [command unit].*1
X_TargetPosition	X-axis Target Position	LREAL	Negative number, positive number, or 0	Set the measurement target position. The unit is [command unit].*1
X_StartPosition	X-axis Measurement Start Position	LREAL	Negative number, positive number, or 0	Set the measurement start position. The unit is [command unit].*1
X_EndPosition	X-axis Measurement End Position	LREAL	Negative number, positive number, or 0	Set the measurement end position. The unit is [command unit].*1
X_Prepare Velocity	X-axis Preparation Velocity	LREAL	Positive number	Set the velocity for the X-axis movement to the preparation position. The unit is [command unit/s].*1
X_Measuring Velocity	X-axis Measuring Velocity	LREAL	Positive number	Set the velocity for the X-axis measuring movement. The unit is [command unit/s].*1
X_Acceleration	X-axis Acceleration	LREAL	Positive number or 0	Set the acceleration for the X-axis movement. The unit is [command unit/s^2].*1
X_Deceleration	X-axis Deceleration	LREAL	Positive number or 0	Set the deceleration for the X-axis movement. The unit is [command unit/s^2].*1
X_Jerk	X-axis Jerk	LREAL	Positive number or 0	Set the jerk for the X-axis movement. The unit is [command unit/s^3].*1
Z_Retract Positon	Z-axis Retraction Position	LREAL	Negative number, positive number, or 0	Set the position for the Z-axis retraction. The unit is [command unit].*1
Z_SearchEnd Position	Z-axis Search End Position	LREAL	Negative number, positive number, or 0	Set the position at which the displacement sensor ends a search for a position that allows measurement. The search range is a range between the retraction position and search end position. If measurement is not enabled within the search range, another search starts. The unit is [command unit].*1
Z_Velocity	Z-axis Velocity	LREAL	Positive number	Set the velocity for the Z-axis movement. The unit is [command unit/s^2].*1
Z_Acceleration	Z-axis Acceleration	LREAL	Positive number or 0	Set the acceleration for the Z-axis movement. The unit is [command unit/s^2].*1

Member	Name	Data type	Valid range	Description
Z_Deceleration	Z-axis Deceleration	LREAL	Positive number or 0	Set the deceleration for the Z-axis movement. The unit is [command unit/s^2].*1
Z_Jerk	Z-axis Jerk	LREAL	Positive number or 0	Set the jerk for the Z-axis movement. The unit is [command unit/s^3].*1
Resolution	Measurement Resolution	UINT	Positive number	Measurement is performed for each specified distance. The unit is [μm].
InPosition	Measurement In-position	UINT	Positive number or 0	Set the in-position width for the target position at which the measurement value is recorded. The unit is [μm].
UseMotionCmd	Motion Control Method for Tracer Control	USINT	0	Set how to control the motion of the single-axis during the Tracer Control. 0: Standard Tracer Control 1 or larger: (Reserved)
CopyErrTolerateNum	Tolerable Number of Measurement Errors in Tracer Control	USINT	0	Supported in a future upgrade

*1. Refer to *Unit Conversion Settings* in the motion control user's manual for information on command units.

Members of Structure sLineMeasurementData

Member	Name	Data type	Valid range	Description
DataCount	Number of Recorded Data	UINT	0 to 20,000	The number of measurement data points recorded in *XValue* and *ZValue*.
Resolution	Resolution	UINT	Positive number	The X-direction resolution that is used during measurement is stored. The unit is [μm].
XValue	X Measurement Data	ARRAY [0..19999] OF LREAL	Depends on data type.	The measurement positions are recorded. The unit is mm.
ZValue	Z Measurement Data	ARRAY [0..19999] OF LREAL	Depends on data type.	The heights measured at the measurement positions are recorded. The unit is mm.
Projection Amount	Amount of Light Emitted Data	ARRAY [0..19999] OF UINT	Depends on data type.	The amount of light emitted at the measurement positions are recorded.
LightReceived Amount	Amount of Light Received Data	ARRAY [0..19999] OF UINT	Depends on data type.	The amount of light received at the measurement positions are recorded.

Function

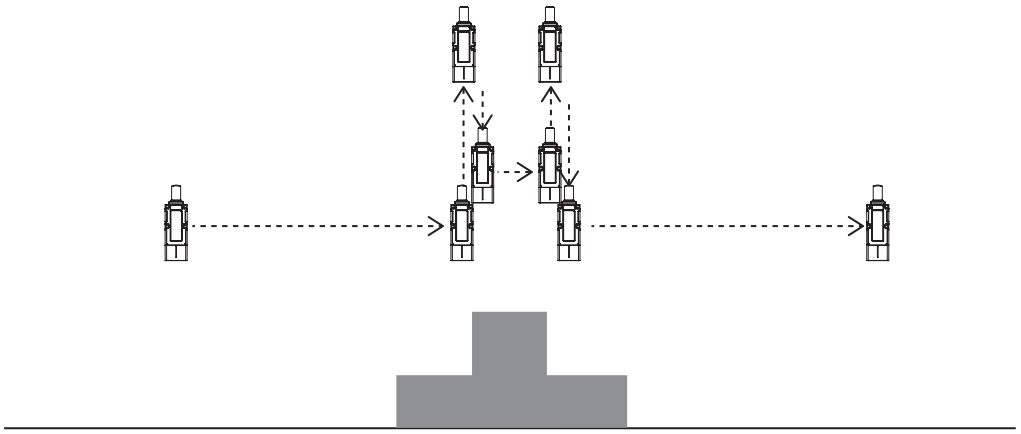
The LineMeasure_Cartesian function block measures the height of measurement surfaces while moving a single axis with a displacement sensor in X and Z directions.

From the measurement result, this function block creates the line measurement data (sLineMeasurementData).

- **Shape measurement using the Surface Search (searching a measurement surface)**

  The height of the displacement sensor is fixed within the measurement range. The sensor scans and measures the target surfaces that come within the measurement range.

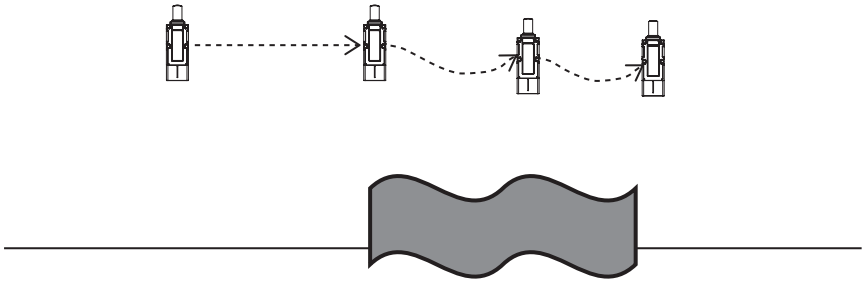
  When the height of the measurement surface changes and it goes outside the measurement range of the displacement sensor, the height of the displacement sensor is readjusted and the measurement is continued. This method is suitable for measuring shapes whose height varies greatly and sharply.



- **Shape measurement using the Tracer Control**

  During measurement, the height of the displacement sensor is adjusted continuously to keep a constant distance between the sensor and measurement surface.

  If the height of the measurement surface changes sharply and the measurement in the Tracer Control cannot be continued, the height is readjusted in the same way as the Surface Search so that the measurement in the Tracer Control can be resumed. This method is suitable for measuring shapes whose height varies gradually.



To obtain accurate line measurement data, always execute this function block in the primary periodic task.

(a) You can select from the following two methods to control the single axis during measurement: Surface Search and Tracer Control.

(b) The function block checks the input parameters when *Execute* changes to TRUE.
If there is no error in the input parameters, either the Surface Search or Tracer Control is started.
If there is an error in the input parameters, the measurement control is not performed, and the function block execution is aborted.
For details on the error codes, refer to *Troubleshooting* on page 137.

(c) For the unit of X-axis and Z-axis display, select “mm”, “μm”, “nm”, or “inch”.
An error end occurs if “pulse” or “degree” is selected.
The unit of axis display is not reflected in the measurement data.
The unit for the measurement data is always “mm”.

(d) An error occurs if the X-axis travel velocity (travel distance per period in the primary periodic task) exceeds the resolution.
Refer to *Surface Search* on page 126 for details.

(e) When *CtrlType* (Control Type) is 0, measurement by the Surface Search method is performed.
Refer to *Surface Search* on page 126 for function details on the Surface Search.

(f) When *CtrlType* (Control Type) is 1, measurement by the Tracer Control method is performed.
Refer to *Tracer Control* on page 128 for function details on the Tracer Control.

(g) Even when the measurement direction (travel direction along the X axis) is negative (*X_StartPosition* > *X_EndPosition*), the line measurement data is output in ascending order of X value.
Sort in ascending order is performed for each unit of the processed data points specified in *NumProcData*.
So, to prevent a Task Period Exceeded error, specify a value for this variable according to the time of the task period to which this function block is assigned.

(h) If *Abort* is changed to TRUE and motion control is in progress, the MC_Stop instruction is executed to stop axis operation.

(i) If an error occurs, *Error* changes to TRUE and the function block execution is aborted. In addition, the error codes are output to *ErrorID* and *ErrorIDEx*.
For details on the error codes, refer to *Troubleshooting* on page 137.

(j) If the function block execution is aborted or ended in an error, the line measurement data will not be created correctly.

(k) In the last stage of processing this FB, the line measurement data is sorted in ascending order. In the sort processing, the number of processed data points (NumProcData) per control period can be set.

Input Variables to Set Values Output from the Displacement Sensor

The following is a setting example of the input variables to which the values output from the displacement sensor are assigned.

Change the settings according to your setting conditions.

● *MeasuringValue*, *ProjectionAmount*, and *LightReceivedAmount*

On the ZW-8000/7000/5000 setting page of the Sysmac Studio, the following digital output is set for each bank.

▼ Digital Output

Item	Content	Initial value
OUT1	Task1	Task1
OUT2	Light power	Task2
OUT3	Peak amount of received light	Task3
OUT4	Task4	Task4
Clamp output	-2147.483648(0x80000000) mm	-2147.483648(0x80000000) mm

Set the Task 1 output to Height. The Task 1 output value is the value of *MeasuringValue*.

▼ Measurement item

Item	Content	Initial value
Measurement item	Height	Height
Measurement surface	Peak	Peak

When the digital output for the displacement sensor is set as above, assign the following device variables to *MeasuringValue*, *ProjectionAmount*, and *LightReceivedAmount*.

Output Data 1	Output Data 1	R	DINT	E001_Output_Data_1	→MeasuringValue
Output Data 2	Output Data 2	R	DINT	E001_Output_Data_2	→ProjectionAmount
Output Data 3	Output Data 3	R	DINT	E001_Output_Data_3	→LightReceivedAmount
Output Data 4	Output Data 4	R	DINT	E001_Output_Data_4	

● *Sensor_ENABLE, Sensor_STABILITY, Sensor_TASKSTAT*

Assign the following device variables to the *Sensor_ENABLE*, *Sensor_STABILITY*, and *Sensor_TASKSTAT* input variables. *MeasuringValue* (Displacement Sensor Measurement Value) is obtained from the Task 1 output, so *Sensor_TASKSTAT* is also obtained from the TASKSTAT_T1 bit.

▼ Sensor Head Status Signal2	Sensor Head Status Signal2	DWORD	E001_Sensor_Head_Status_Signal2	
HOLDSTAT	Hold status	BOOL	E001_HOLDSTAT	
RESETSTAT	Reset status	BOOL	E001_RESETSTAT	
LIGHT	Light status	BOOL	E001_LIGHT	
STABILITY	Stability status	BOOL	E001_STABILITY	→Sensor_ENABLE
ENABLE	Enable status	BOOL	E001_ENABLE	→Sensor_STABILITY
GATE	Data output completion	BOOL	E001_GATE	
OR	Overall judgement output	BOOL	E001_OR	
TASKSTAT_T1	TASKSTAT_T1 Bit	BOOL	E001_TASKSTAT_T1	→Sensor_TASKSTAT
TASKSTAT_T2	TASKSTAT_T2 Bit	BOOL	E001_TASKSTAT_T2	
TASKSTAT_T3	TASKSTAT_T3 Bit	BOOL	E001_TASKSTAT_T3	
TASKSTAT_T4	TASKSTAT_T4 Bit	BOOL	E001_TASKSTAT_T4	

Process Flow from Creating Line Measurement Data to Creating 2D Shape Data

This section describes the process flow from creating the line measurement data to creating the 2D shape data using the LineMeasure_Cartesian, LineMeasure_CreateShape2D, and LineMeasure_CreateShape2DMaster function blocks.

Measurement Environment

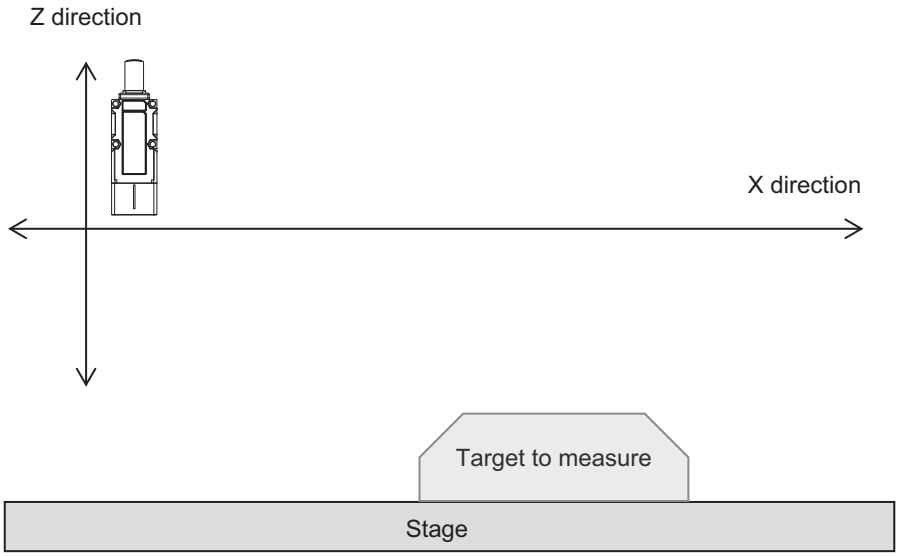
Place the target to measure (workpiece) on the stage.

Configure a servo axis that can move in X and Z directions relative to the stage. Attach a displacement sensor to the axis. The distance between the attached sensor and workpiece must vary in X and Z directions.

Attach a displacement sensor so that the sensor measurement value will be the opposite direction to the world coordinate system. In the following example, the sensor measurement value changes to the negative direction if the sensor head moves to the positive direction. Also, the sensor measurement value changes to the positive direction if the sensor head moves to the negative direction.

Then, make the settings to enable the displacement sensor to perform measurement.

Refer to the manuals for the displacement sensor for how to make settings for the displacement sensor.



Line Measurement Data and 2D Shape Data

The line measurement data is the information on positions of two axes (parallel to the sensor measurement direction and parallel to the scanning direction) that are measured during servo axis motion and information on the displacement sensor. The information is stored as a group of heights (Z direction) measured at scanning points (X direction). The line measurement data is used to create the 2D shape data, which are described in the following sections. Refer to *Line Measurement Data* on page 123.

2D shape data is the converted line measurement data after processing such as equal-interval arrangement of data points, filtering, and shape correction (slope, X and Z directions) are performed. 2D shape data is used for the feature amount calculations or graphic display on an OMRON NA-series programmable terminal. Refer to *2D Shape Data* on page 123.

Master Measurement and Target Measurement

- “Master measurement” refers to the measurement that is performed in the standard environment and with the standard samples.

  “Master 2D shape data” refers to the 2D shape data created from the line measurement data obtained through master measurement. The data such as the correction parameter used for creating the 2D shape data is also included in the master 2D shape data. Refer to *Master 2D Shape Data* on page 123.
- “Target measurement” refers to normal measurement.

  “Target 2D shape data” refers to the 2D shape data created from the line measurement data obtained through target measurement.

Target 2D Shape Data Creation Method

There are the following two methods to create the target 2D shape data:

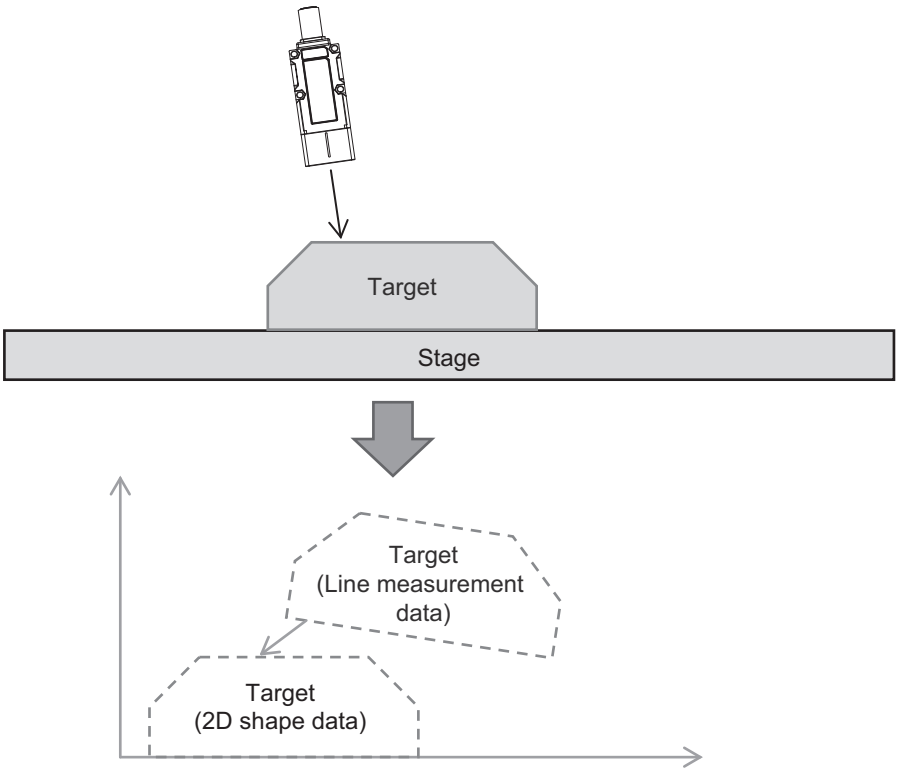
The process in which the 2D shape data is created through target measurement only. The process in which the 2D shape data is created through target measurement based on the result of master measurement.

The first is used to measure shapes to create the target 2D shape data with the feature amount calculation function block.

The second can be used not only to measure shapes but also to compare the data with the master 2D shape data.

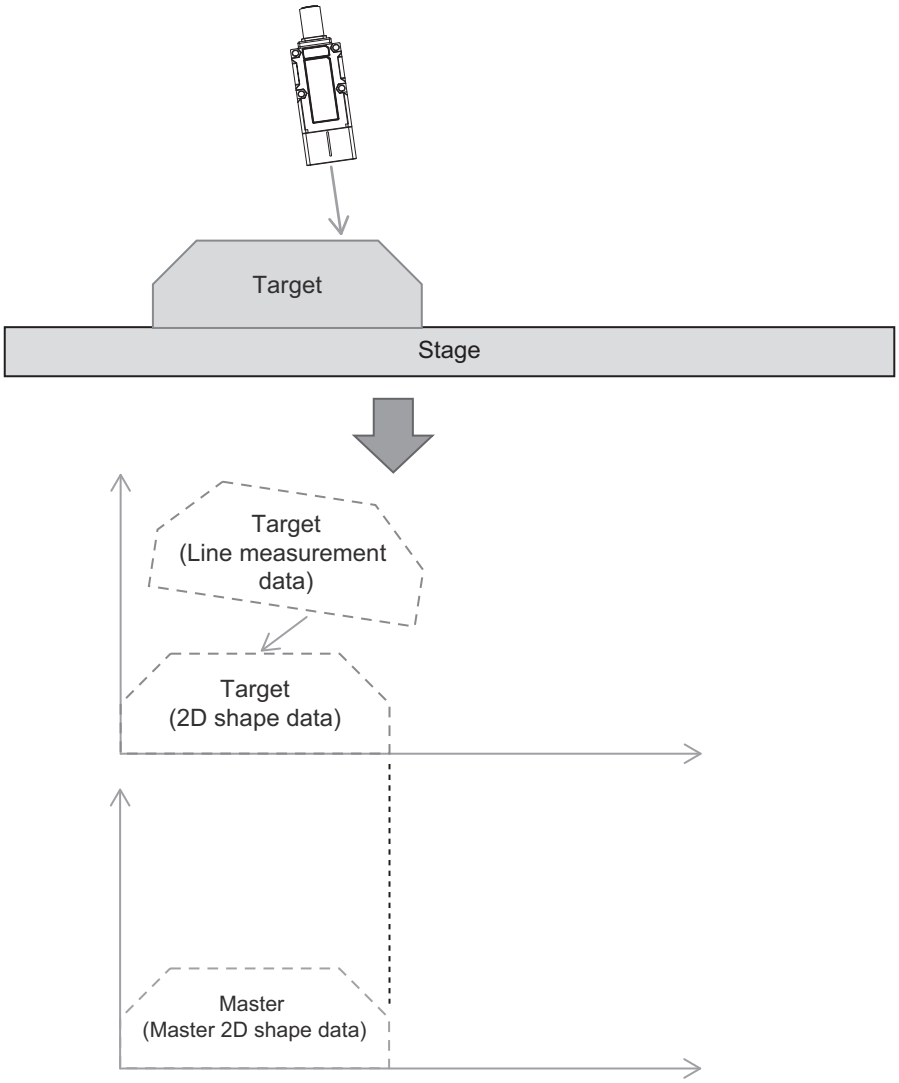
<When creating 2D shape data through target measurement only>

The target 2D shape data is created from the line measurement data that is obtained through target measurement and corrected according to the correction parameter that specifies the slope, height and position.



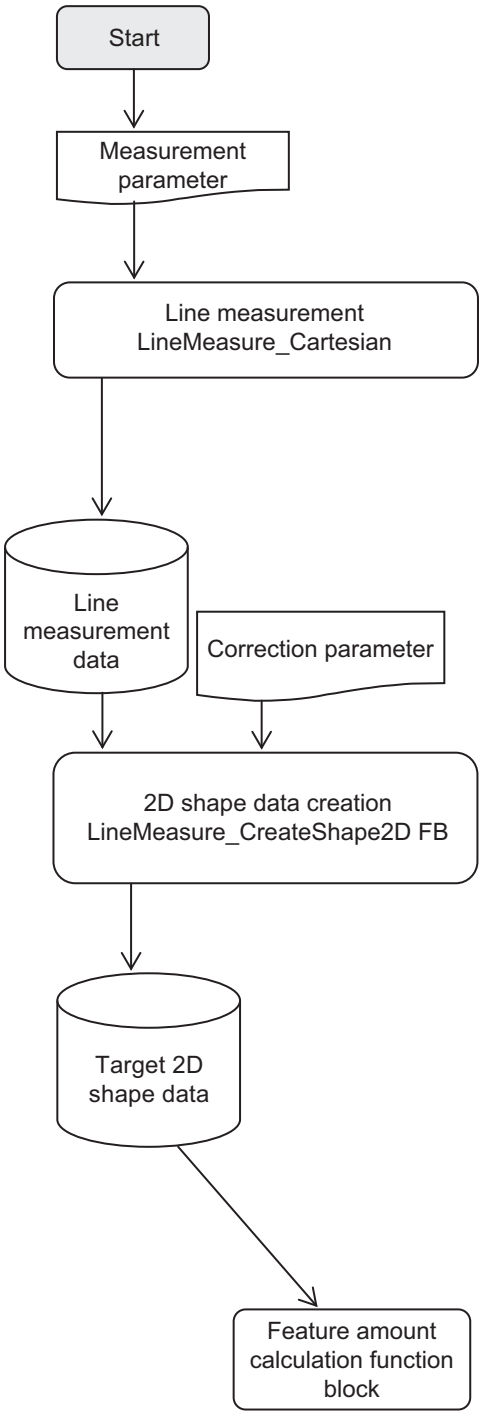
<When creating 2D shape data through target measurement based on the result of master measurement>

The target 2D shape data is created according to the slope and reference positions in X and Z directions of the master 2D shape data.



In the above example, the target is placed at different positions when target measurement and master measurement was performed. The correction function creates the target 2D shape data as if the target is measured at the same position as master measurement.

● Target 2D Shape Data Obtaining Process (When Master 2D Shape Data is Not Used)



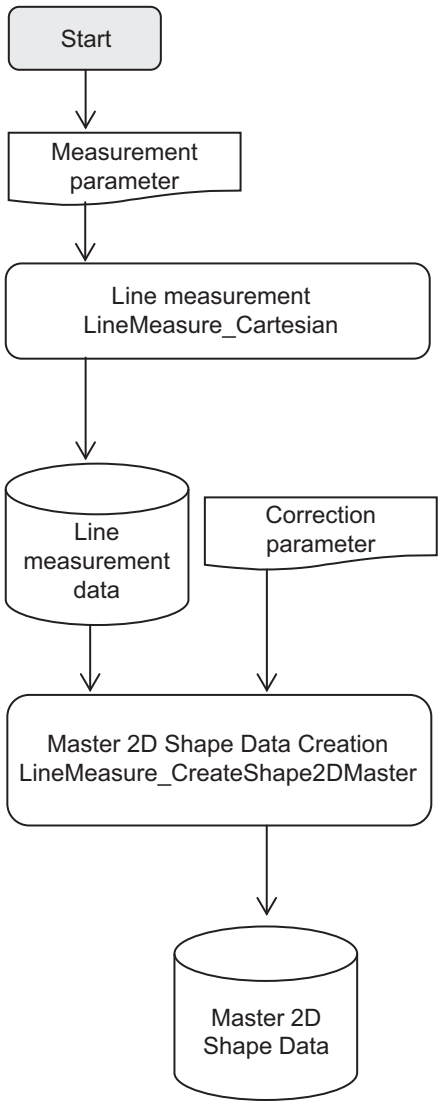
Set the measurement parameter (measurement range, axis velocity, etc.).

Execute the line measurement function block to create the line measurement data.

Set the correction parameter (slope, height and position) as necessary.

Execute the 2D shape data creation function block to create target 2D shape data. This data can be used to execute the feature amount calculation function block.

Master 2D Shape Data Obtaining Process



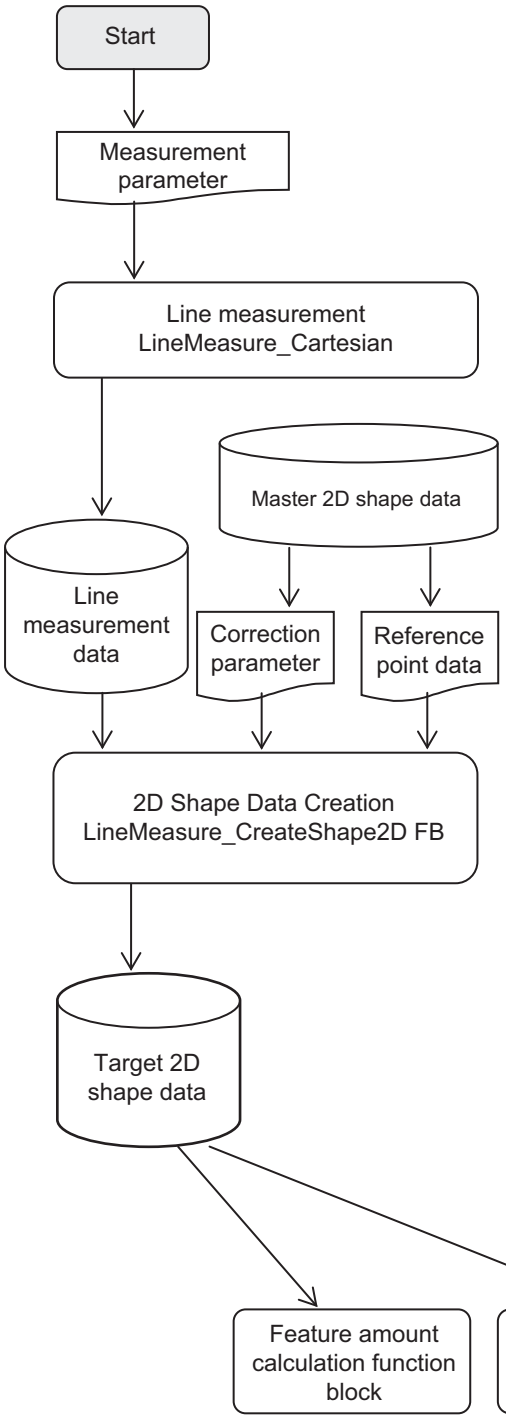
Set the measurement parameter (measurement range, axis velocity, etc.).

Execute the shape measurement of the master sample and create the line measurement data.

Set the correction parameter. For example, if the workpiece is tilted during measurement and you want to create a horizontal shape data, you can correct the shape data by setting the reference angle to 0°.

When the line measurement data and correction parameter are input and the master 2D shape data creation function block is executed, the master 2D shape data is created.

● Target 2D Shape Data Obtaining Process (When Master 2D Shape Data Is Used)



Set the measurement parameter (measurement range, axis velocity, etc.). Set the same resolution as the master 2D shape data. If a different value is set, the master and target cannot be compared properly.

Execute shape measurement to create the line measurement data.

Set the same correction parameter and reference point data as the master 2D shape data. If parameters are different, correction cannot be performed properly.

Execute the 2D shape data creation function block to create target 2D shape data. With this data, you can execute not only the feature amount calculation function block but also the waveform comparison function block.

Line Measurement Data

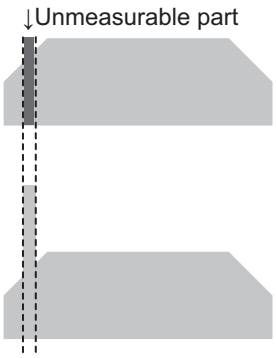
The line measurement function block (LineMeasure_Cartesian) creates the line measurement data which consists of the position (X), height (Z), amount of emitted light, and amount of received light.

The position represents a position at which measurement is performed, and the height, amount of emitted light and received light are the measurement results that the displacement sensor outputs at the measurement position.

The measurement data is output in ascending order of positions (X (n) ≤ X (n + 1)).

If height measurement is not possible, "∞" is set for *ZValue*.* 1

For example, when a part of the following sample cannot be measured, the unmeasurable part is recorded as a protrusion in the measurement data as shown in the following figure.



*1. Refer to *Unmeasurable Shapes* on page 139 for information on unmeasurable area.

2D Shape Data

The 2D shape data creation function block (LineMeasure_CreateShape2D) outputs the 2D shape data which consists of only heights (Z) at positions that are evenly spaced by the resolution.

The position for each height can be calculated as follows: position = (resolution × (array element number -1)).

Master 2D Shape Data

The master 2D shape data refers to a group of data that includes the correction parameter used to create the 2D shape data through master measurement, 2D shape data that is obtained, and reference points.

The resolution recorded in the master 2D shape data is used as the resolution for performing line measurement for a target.

The correction parameter recorded in the master 2D shape data is used as the correction parameter for creating the target 2D shape data.

Measurement Resolution and Measurement Record Position

The measurement data is recorded for each resolution interval between the measurement start position and measurement end position.

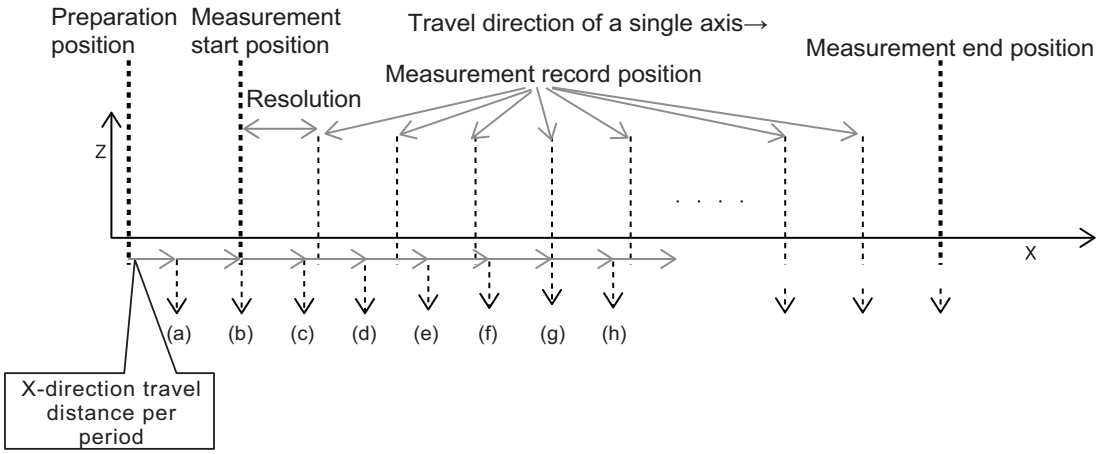
So, an error occurs if the measurement range divided by the resolution exceeds the array size of *XValue* and *ZValue* (20,000) in the line measurement data.

The positions determined by the measurement range divided by the resolution are called “measurement record positions”.

When the value of Xposition reaches or exceeds a measurement record position, the position, height, amount of emitted light and amount of received light are recorded in the line measurement data.

The travel velocity along the X axis to a target position is always the same. Therefore, if the X-direction travel distance per period that is calculated from the travel velocity is smaller than the resolution, the record positions will be deviated.

The X-direction travel distance per period is calculated by the following formula: X-axis velocity (X_Velocity) × Task period to which this function block is assigned.



In the above example, the measurement data is as follows.

(a) Data is not recorded becauses the measurement start position is not reached.

(b) The X-axis servo actual position is recorded in *XValue*[1].

(c) Data is not recorded because the measurement record position is not reached.

(d) The X-axis servo actual position is recorded in *XValue*[2].
The sum of the displacement sensor measurement value and Z-axis servo position is recorded in *ZValue*[2].

(e) The X-axis servo actual position is recorded in *XValue*[3].
The sum of the displacement sensor measurement value and Z-axis servo position is recorded in *ZValue*[3].

(f) The X-axis servo actual position is recorded in *XValue*[4].
The sum of the displacement sensor measurement value and Z-axis servo position is recorded in *ZValue*[4].

(g) Data is not recorded because the measurement record position is not reached.

(h) The X-axis servo actual position is recorded in *XValue*[5].
The sum of the displacement sensor measurement value and Z-axis servo position is recorded in *ZValue*[5].

Deviation can be reduced when the value of the resolution is close to an integer multiple of the X-direction travel distance per period.

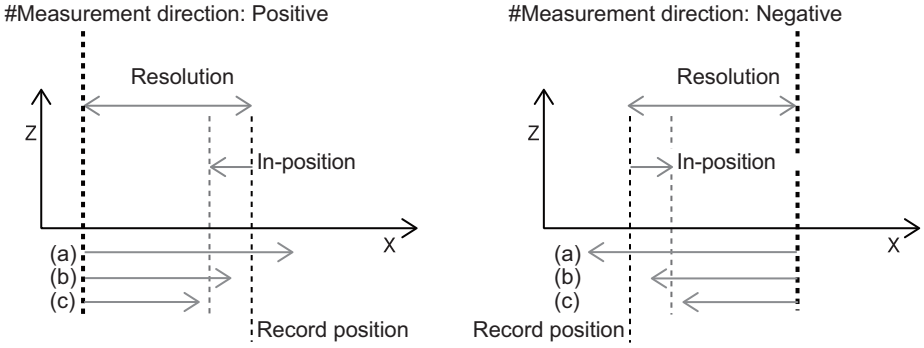
If the travel velocity is too high and the X-direction travel distance exceeds the resolution, an error occurs.

For example, if the resolution is set to 10 μm and the task period to which this function block is assigned is 1 ms, the velocity for the X axis must be 10 mm/s or less.

The acceleration, deceleration and jerk are not taken into account in the calculation.

By setting Measurement In-position, you can record data when the sensor reaches the amount of Measurement In-position before the measurement record position.

If the Measurement In-position is set to 1 or larger, the measurement data is recorded even when the X-axis position comes earlier than the record position.



In the above figures, the measurement data is recorded at (a) and (b).

The data is not recorded at (c).

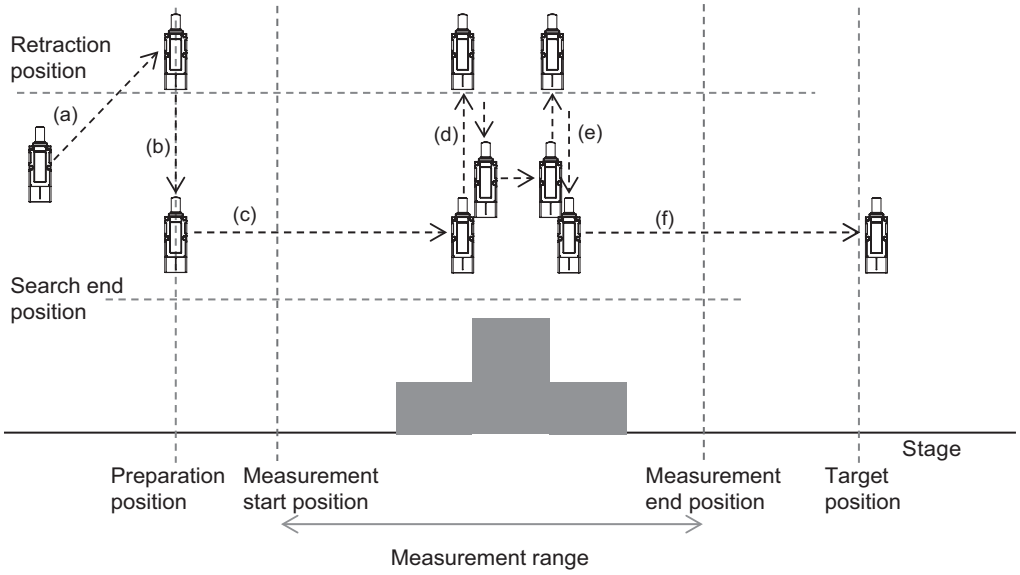
Use the Measurement In-position when you want to perform measurement even if the axis travel distance per period is smaller than the resolution, as shown in (b).

The value of Measurement In-position must be less than the resolution.

An error occurs if the value of Measurement In-position is equal to or larger than the resolution.

Surface Search

In the Surface Search, the shape of the surface is measured while the servo axis is moved. The axis is connected in such as way that the following positional relationship is formed between the displacement sensor and stage.



(a) The single axis moves in X direction to the preparation position (*X_PreparePosition*) and in Z direction to the retraction position (*Z_RetractPositon*).

The X preparation position and Z retraction position must be set to positions where the sensor does not touch the target object.

(b) The sensor performs “measurement positioning” at the preparation position.

“Measurement positioning” refers to a movement of the single axis to a height where the displacement sensor recognizes the measurement value of the measurement surface as 0.

The following procedure is used to perform measurement positioning.

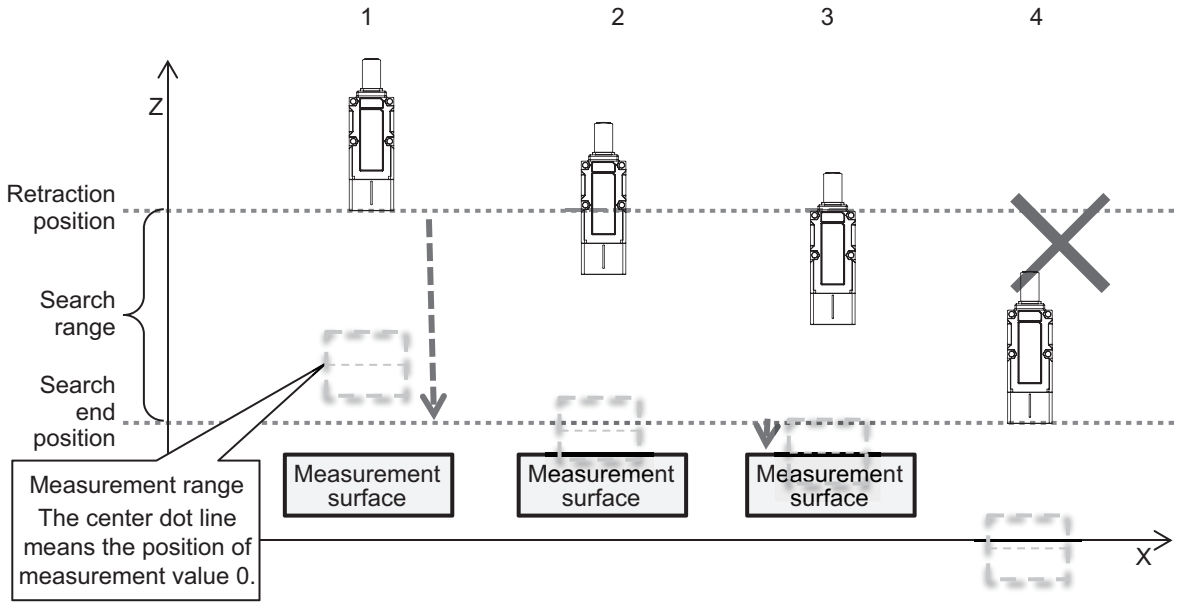
1 The single axis starts moving in Z direction to the search end position (*Z_SearchEndPosition*).

The search end position must be set to a position where the sensor does not touch the target to measure.

2 When the displacement sensor is enabled to measure, i.e. the measurement surface entered an area in which the sensor can perform measurement, the single axis moves to a height where the measurement value is 0.

3 The single axis stops at a height where the measurement value is 0.

4 If the axis reaches the search end position and measurement is still disabled, an error end occurs.



(c) The single axis moves in X direction to the target position.

Shape measurement is performed while the X position is within the range between the measurement start position (*X_StartPosition*) and measurement end position (*X_EndPosition*).

It is also possible to execute scanning in the negative direction (X start position > X end position, or preparation position > target position).

The measurement range must be equal to or less than the X-axis travel range.

At each measurement position, the position, height, amount of emitted light and received light are recorded in the line measurement data.

(d) When a state of measurement disabled is detected during measurement*, then measurement re-positioning is performed.

The operation of measurement re-positioning is the same as measurement positioning. In re-positioning, the measurement surface is searched and the single axis moves to a height where the displacement sensor recognizes the measurement value of the measurement surface as 0.

The following procedure is used to perform measurement re-positioning.

1 A deceleration stop is performed for X axis and Z axis.

X-axis Deceleration, X-axis Jerk, Z-axis Deceleration and Z-axis Jerk are used for the deceleration stop.

If X-axis Deceleration and Z-axis Deceleration are set to 0, the Maximum Deceleration axis parameter is used.

2 The axis moves to the present X-axis measurement and record position and Z-axis retraction position.

3 The axis starts moving to Z-axis Search End Position (*Z_SearchEndPosition*).

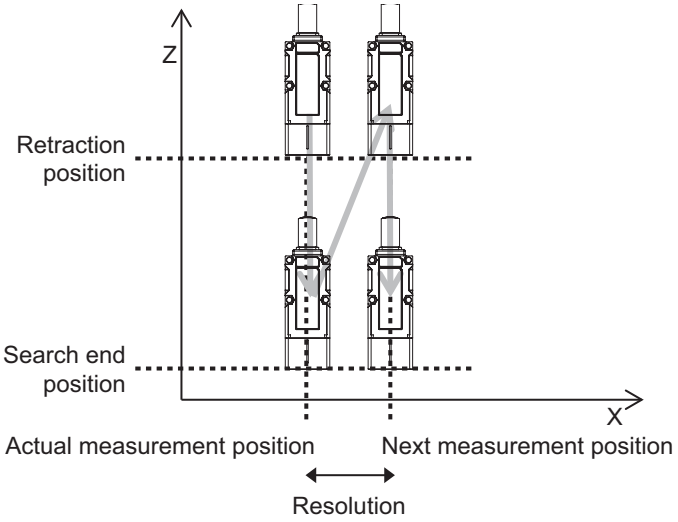
4 When the displacement sensor is enabled to measure, i.e. the measurement surface entered an area in which the sensor can perform measurement, the single axis moves to a height where the measurement value is 0.

When the axis finishes moving, it starts moving to the target position and measurement resumes.

5 If the axis reaches the search end position and measurement is still disabled, then the position, height, amount of emitted light and received light at present position are recorded in the line measurement data. For the height which cannot be measured, “∞” is set.

6 The value of Resolution is added to the measurement record position. The single axis moves to a new measurement and record position and performs re-positioning.

7 Measurement re-positioning is repeated until measurement re-positioning is successful or the measurement end position is exceeded.



* Refer to the displacement sensor user's manual for factors that cause measurement to fail.

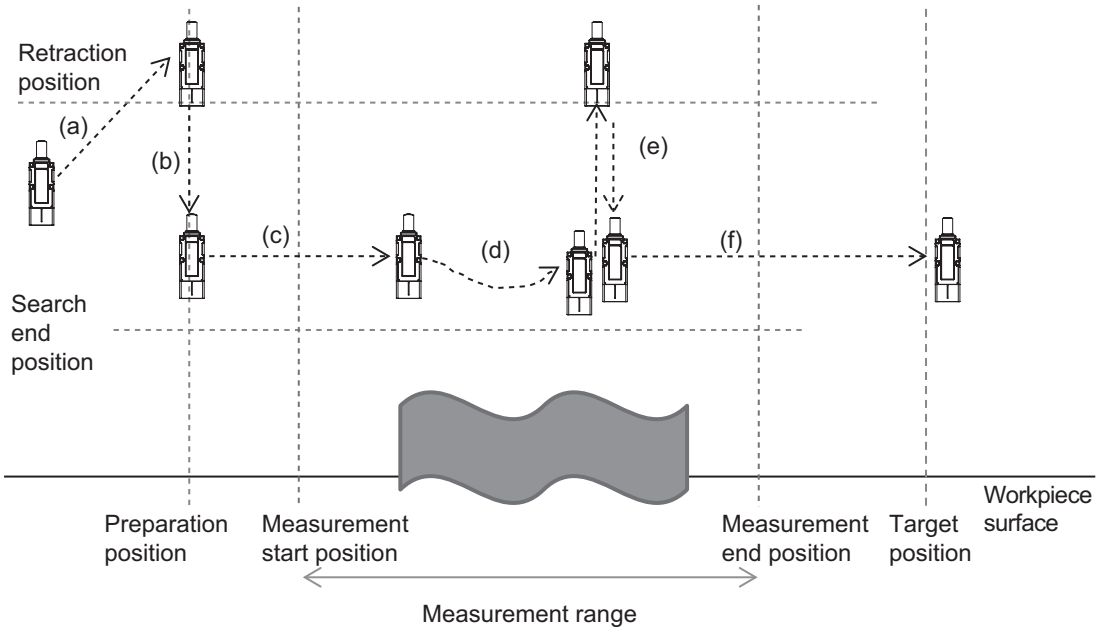
(e) When measurement is disabled again during measurement, then measurement re-positioning is performed.

(f) The following operation is repeated until the axis reaches the measurement end position: measurement → measurement disabled → measurement re-positioning → measurement → ...

When the target position is reached, the shape measurement is completed.

Tracer Control

In the Tracer Control, the shape of a surface is measured while the single axis, to which the displacement sensor is attached, is moved so that the displacement sensor always recognizes the measurement value (height) as 0.



The following procedure is used to perform measurement.

(a) The single axis moves in X direction to the preparation position (*X_PreparePosition*) and in Z direction to the retraction position (*Z_RetractPositon*).

The X preparation position and Z retraction position must be set to positions where the sensor does not touch the target object.

(b) The sensor performs “measurement positioning” at the preparation position.

This procedure is the same as the Surface Search.

(c) The single axis moves in X direction to the target position.

Shape measurement is performed while the X position is within the range between the measurement start position (*X_StartPosition*) and measurement end position (*X_EndPosition*).

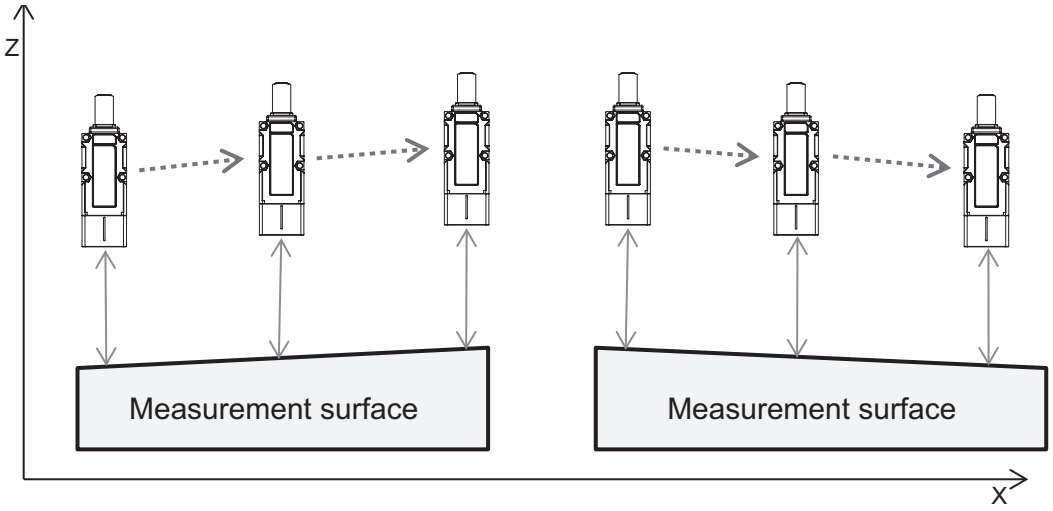
It is also possible to execute scanning in the negative direction (X start position > X end position, or preparation position > target position).

The measurement range must be equal to or less than the X-axis travel range.

(d) While the single axis moves within the measurement range, the height of the axis is adjusted according to the height of the measurement surface.

If the measurement value is not 0, the Z position is changed by the amount of deviation from 0.

For example, if the measurement value is 1 mm, the Z position is raised by 1 mm. If the measurement value is -1 mm, the Z position is lowered by 1 mm.



(e) When a state of measurement disabled is detected during measurement, then measurement re-positioning is performed.

This procedure of measurement re-positioning is the same as the Surface Search.

Once measurement re-positioning is successful, measurement in the Tracer Control resumes.

(f) The following operation is repeated until the axis reaches the measurement end position: measurement → measurement disabled → measurement re-positioning → measurement → ...

When the target position is reached, the shape measurement is completed.

Re-execution of Function Blocks

If a function block is re-executed when execution is in progress (*Busy* = TRUE), an error end occurs.

The measurement execution is terminated.

When you re-execute the function block, change *Abort* to TRUE to terminate the execution, and then change *Execute* to TRUE.

Multi-execution of Function Blocks

For details on multi-execution of function blocks, refer to the motion control user's manual.

Timing Charts

The timing charts are shown below.

● In a Normal State (Surface Search or Tracer Control)

When *Execute* changes to TRUE, *Busy* (Executing) changes to TRUE.

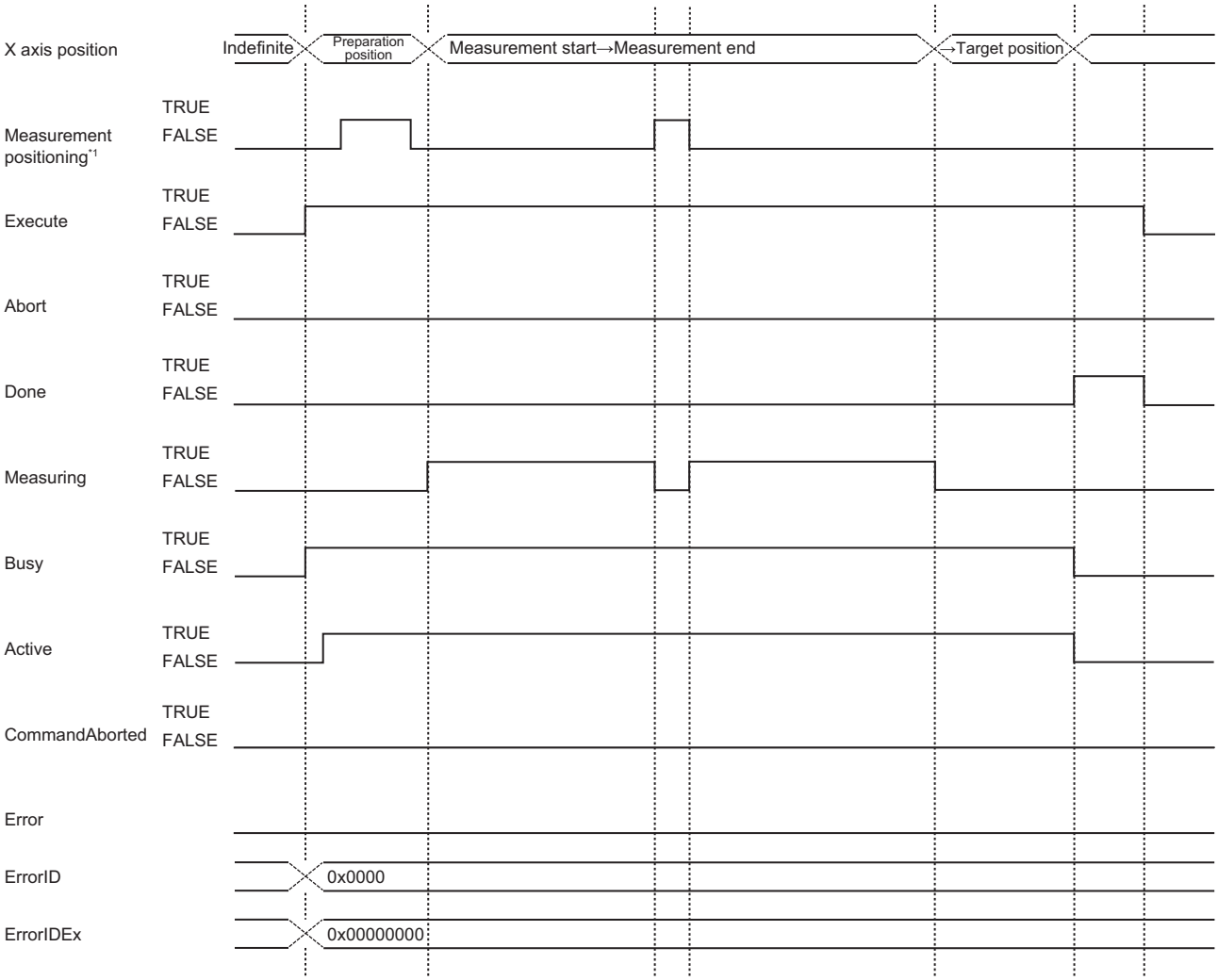
When the internal motion function block is executed, *Active* (Controlling) changes to TRUE.

Measuring is TRUE while the single axis is in the measurement range and shape measurement is performed.

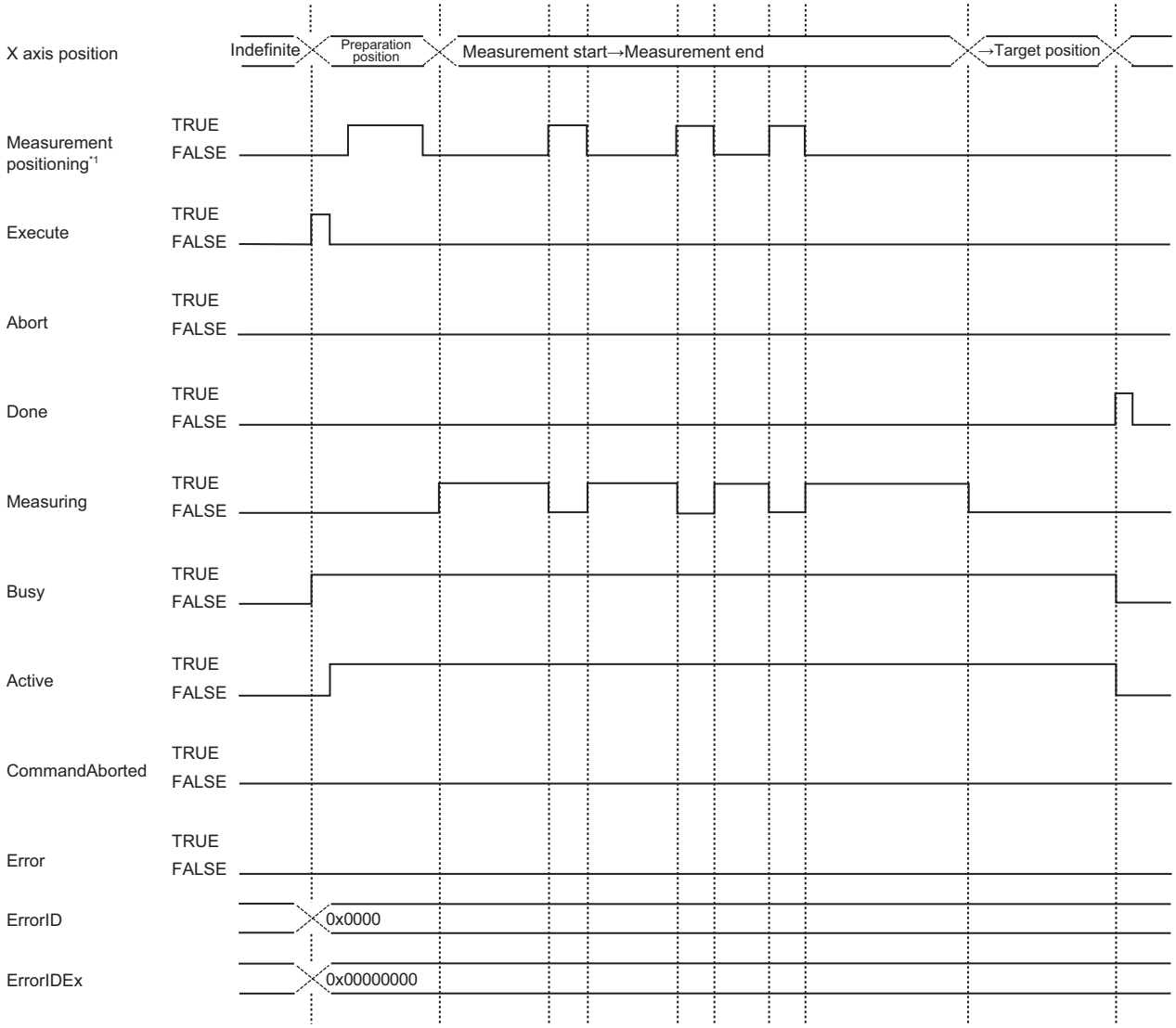
Measuring is FALSE while a state of measurement disabled is detected and the single axis is re-positioned.

When the single axis reaches the target position and the measurement is completed, *Busy* and *Active* change to FALSE, and *Done* changes to TRUE.

Done holds its value while *Execute* remains TRUE.



*1. Refer to *Surface Search* on page 126 for information on measurement positioning.

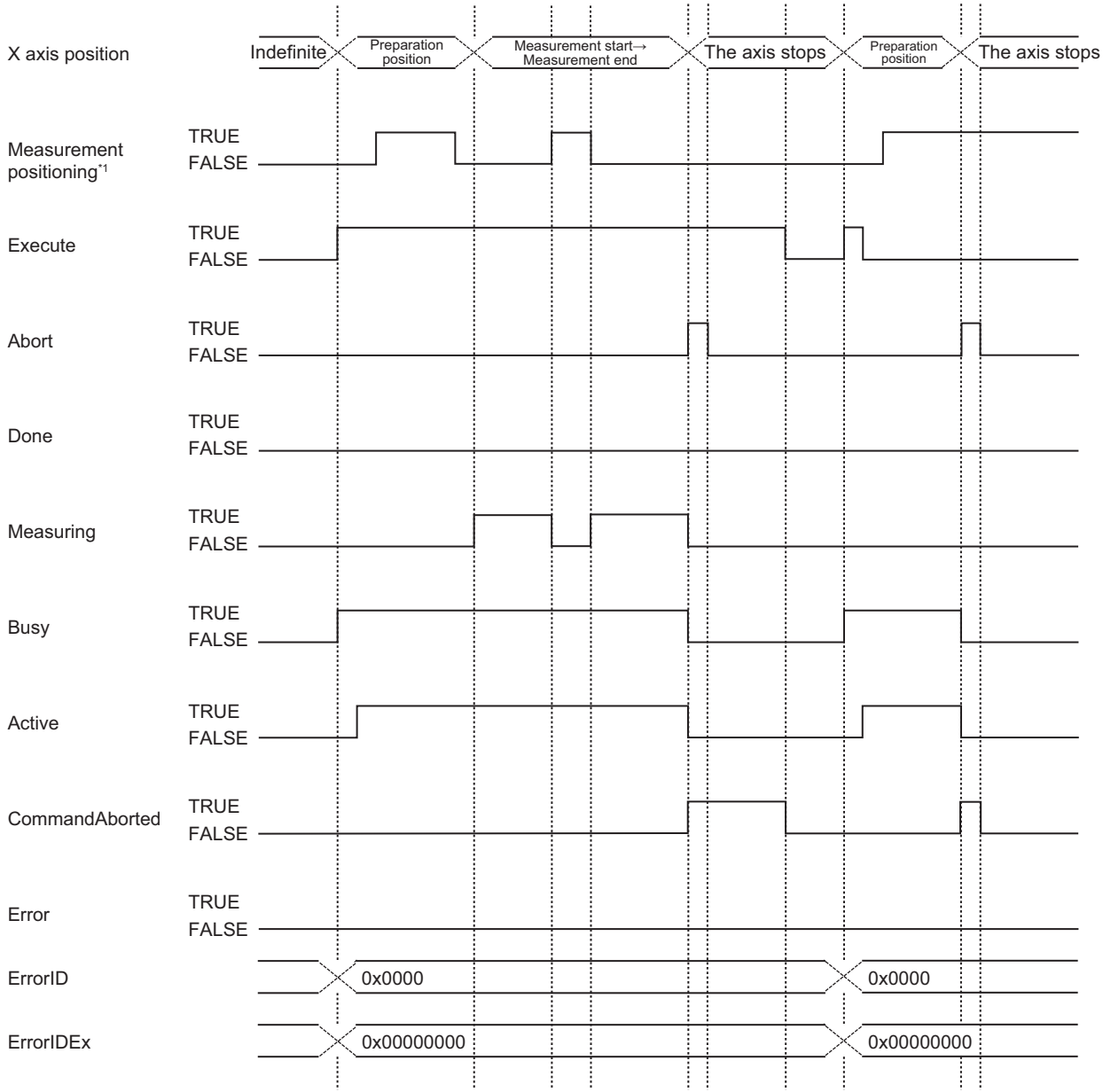


*1. Refer to *Surface Search* on page 126 for information on measurement positioning.

When the Function Block Is Aborted

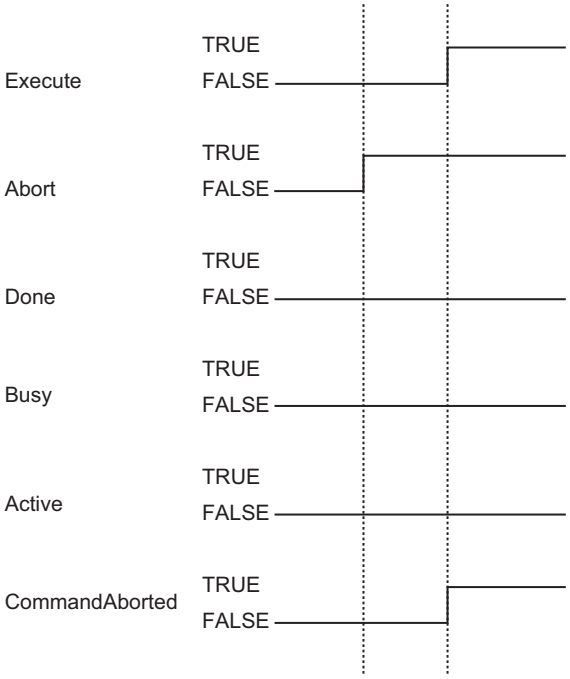
If *Abort* changes to TRUE when execution is in progress, *Active* changes to FALSE and *CommandAborted* (Interruption Completion) changes to TRUE.

CommandAborted holds its value while *Execute* remains TRUE.



*1. Refer to *Surface Search* on page 126 for information on measurement positioning.

If *Execute* changes to TRUE while *Abort* is TRUE, *CommandAborted* (Interruption Completion) changes to TRUE without performing the processing.



When an Error Occurs (Execution Check Error)

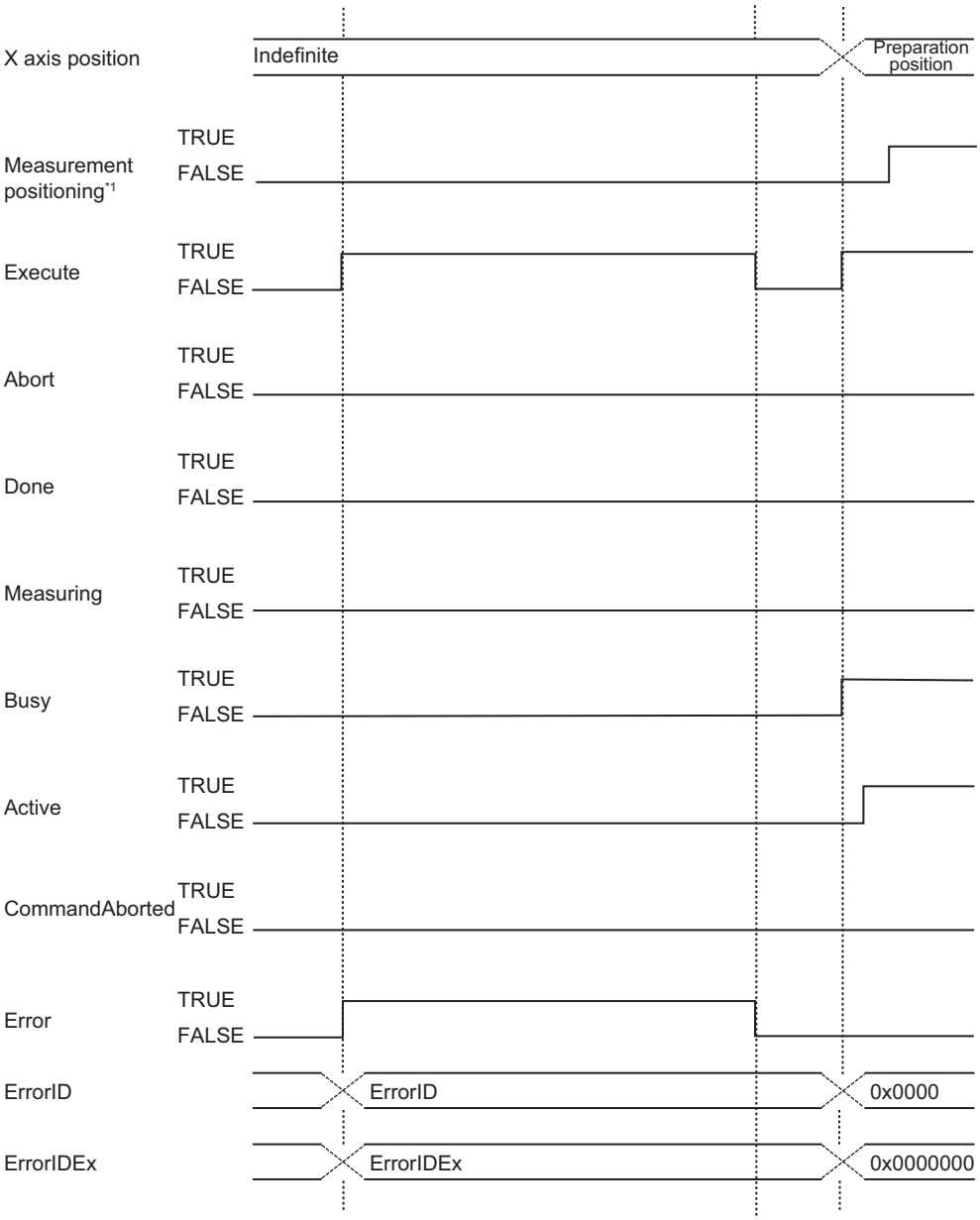
If an error is detected during the check performed when function block is executed, *Error* changes to TRUE.

The values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code).

Busy and *Active* remain FALSE.

Error holds its value while *Execute* remains TRUE.

ErrorID and *ErrorIDEx* hold their values until the function block is re-executed.



*1. Refer to *Surface Search* on page 126 for information on measurement positioning.

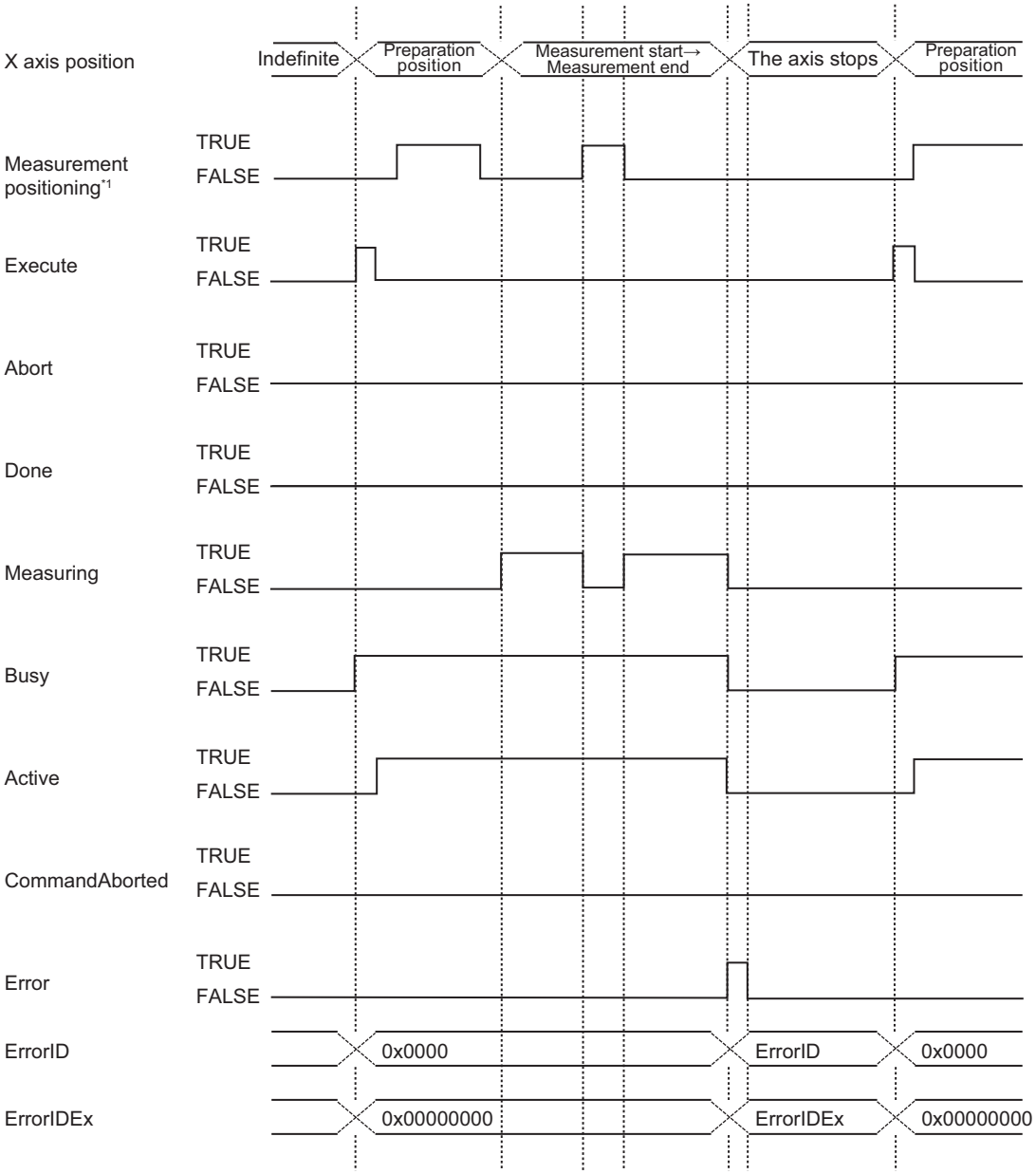
When an Error Occurs (Execution Error)

If an error occurs during execution of this function block, *Busy* and *Active* change to FALSE and *Error* changes to TRUE.

The values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code).

Error holds its value while *Execute* remains TRUE.

ErrorID and *ErrorIDEx* hold their values until the function block is re-executed.



*1. Refer to *Surface Search* on page 126 for information on measurement positioning.

Precautions for Correct Use

- For the settings of a displacement sensor, do not use the filter (median filter, average filter, or low-pass filter) in the output condition settings for Task 1 that outputs the height in principle. If the filter is used, a correct measurement result may not be obtained, or operation may become unstable.
- For details on filtering the measurement data, refer to the descriptions in sections of LineMeasure_CreateShape2DMaster and LineMeasure_CreateShape2D.
- Always execute this function block in the primary periodic task. If you execute it in the periodic task, a correct measurement result may not be obtained.
- The 2D shape data creation function blocks (LineMeasure_Shape2DMaster and LineMeasure_Shape2D) use the line measurement data as the in-output variables. These function blocks must be executed only after this function block is completed normally.
- Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3CD6	0x0000 0001	An X-axis error occurred.	*1
	0x0000 0002	An Z-axis error occurred.	*1
	0x0000 0003	The function block was executed in the event task.	Execute this function block in the primary periodic task.
	0x0000 0004	The unit of X-axis display is set to “pulse” or “degree”.	Select one of “mm”, “µm”, “nm”, or “inch”.
	0x0000 0005	The unit of Z-axis display is set to “pulse” or “degree”.	Select one of “mm”, “µm”, “nm”, or “inch”.
	0x0000 0006	The undefined Control Type (*CtrlType*) was specified.	Specify a value which is within the valid range.
	0x0000 0007	The value specified for X-axis Preparation Position (*X_PreparePosition*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0008	The value specified for X-axis Target Position (*X_TargetPosition*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0009	The value specified for X-axis Measurement Start Position (*X_StartPosition*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 000A	The value specified for X-axis Measurement End Position (*X_EndPosition*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 000B	The measurement range is outside the X-axis travel range.	Specify the values so that they meet the following condition. X-axis Preparation Position ≤ Measurement Start Position < Measurement End Position ≤ Target Position Or, X-axis Preparation Position ≥ Measurement Start Position > Measurement End Position ≥ Target Position
	0x0000 000C	The value specified for Z-axis Retraction Position (*Z_RetractPositon*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 000D	The value specified for Z-axis Search End Position (*Z_SearchEndPosition*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 000E	The value specified for Measurement Resolution (*Resolution*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 000F	The value specified for Measurement In-position is equal to or larger than Measurement Resolution.	Specify a value which is smaller than Measurement Resolution.

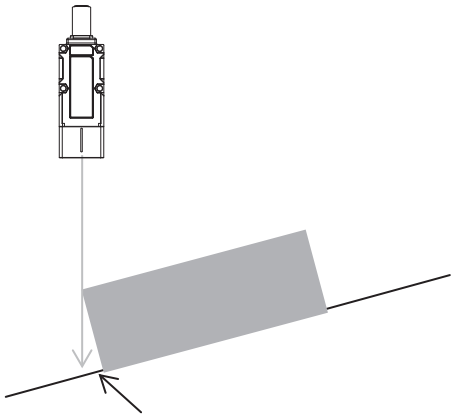
Error code	Expansion error code	Description	Corrective action
0x3CD6	0x0000 0010	The value specified for Motion Control Method for Tracer Control (*UseMotionCmd*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0011	The X-axis velocity is too high relative to the resolution.	Adjust the X-axis velocity, resolution, and task period in which the function block is executed.
	0x0000 0012	The number of measured and recorded points (measurement range divided by resolution) exceeded the capacity of the measurement data.	Correct the measurement range and resolution.
	0x0000 0013	Servo is not turned ON for X axis.	Turn ON the servo. When "Virtual Servo Axis" is set for the X axis Set the "Servo Axis".
	0x0000 0014	Home is not defined for X axis.	Define home before executing the function block.
	0x0000 0015	Servo is not turned ON for Z axis.	Turn ON the servo. When "Virtual Servo Axis" is set for the X axis Set the "Servo Axis".
	0x0000 0016	Home is not defined for Z axis.	Define home before executing the function block.
	0x0000 0017	Z-axis measurement position setting failed at the X-axis Preparation Position.	Adjust the measurement parameter (X-axis Preparation Position, Z-axis Temporary Position, and Threshold). Also, check that the measurement of the target is enabled.
	0x0000 0018	The function block was re-executed when execution is in progress.	Change *Abort* to TRUE to terminate the execution, and then re-execute the function block.

*1. Refer to the motion control user's manual.

Restriction

Unmeasurable Shapes

The displacement sensor is set straight above the target to measure. So, shapes such as shown below cannot be measured correctly.



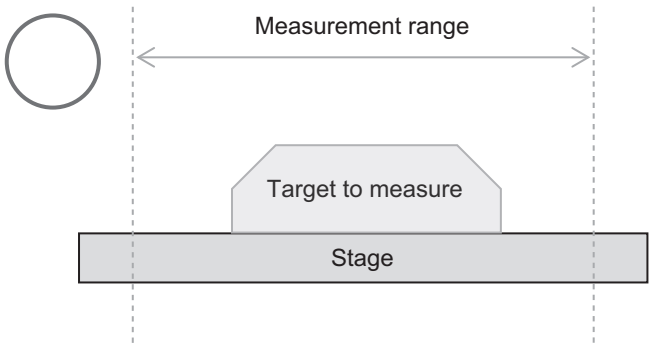
The height of the shaded part cannot be measured.
The measured shape data is recorded as follows.

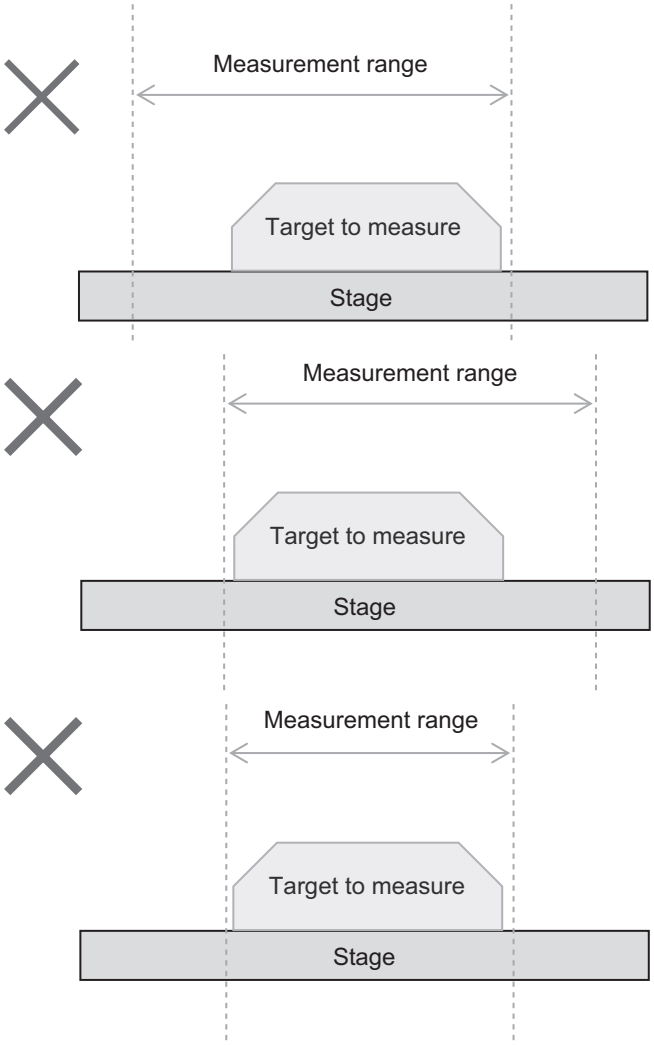
Measurement Range and Space

When executing shape data correction or feature amount calculation using the shape data, you may need to use some parameters to specify spaces where there is no target to measure.

For this purpose, when you make settings for measurement, you need to set a measurement range which is wider than the target to measure, and leave spaces before and after the target.

Although it is possible to measure objects whether spaces are ensured or not, the accurate result of correction or feature amount calculation may not be obtained if spaces are too small. To prevent this, the spaces before and after the target to measure must be set to values more than 10 times the measurement resolution.





Deformation Caused by Slope Correction

Depending on the shape of the target or rotation angle, the shape of the measured target may not be retained after slope correction.

If slope correction is executed for targets such as shown below, the measured shape may be deformed.

Figure 1: The target's edge line (segment A) meets the horizontal line at almost a right angle. When the measurement data of this target is rotated clockwise for correction, segment A moves to a position which cannot be expressed in 2D shape data. As the result, the shape is deformed as shown in figure 2, in which segment A is expressed as a perpendicular line dropped from the top of segment A.

The original data of the segment is lost after it was rotated and changed in shape. So, even if the rotation is performed again in a reverse direction, the original shape cannot be restored.

If the rotation is performed counter-clockwise, the segment opposite to segment A is deformed in the same way.

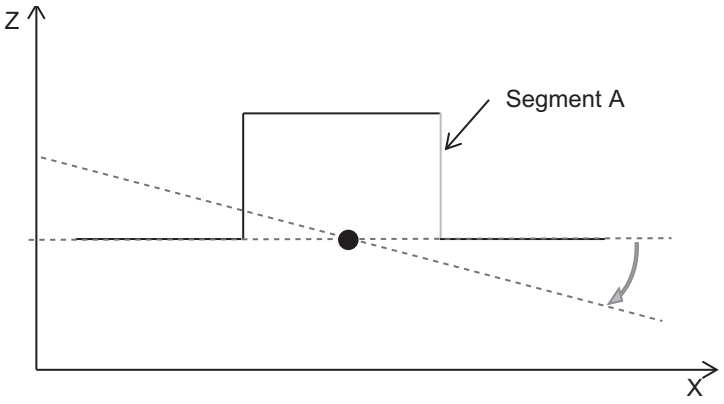


Figure 1

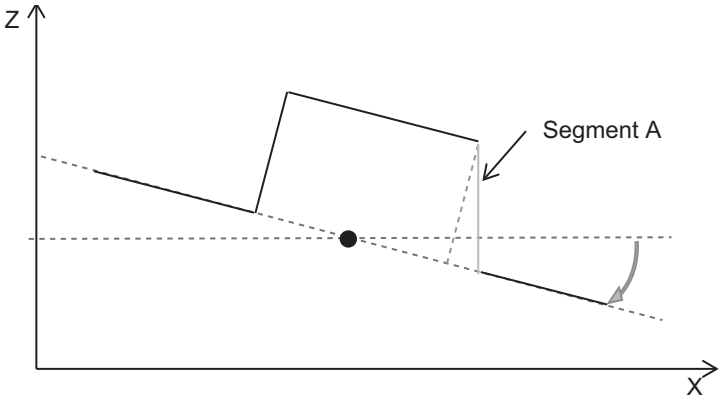


Figure 2

Data Loss Caused by Conversion to Shape Data

When the measurement data is converted to the shape data, a part of the data is lost without being converted to the shape data if the following conditions are met.

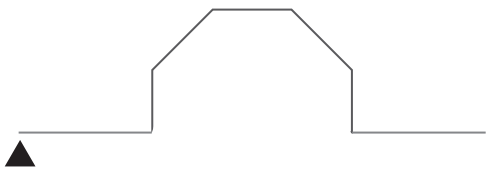
This occurs because the start position of the shape data is aligned with the start position of the master data when conversion to the shape data is performed.

This means, this problem does not occur when the master data correction is executed.

#Conditions of data loss#

- The position of the target to measure is deviated from the position where the master was measured.
- Position correction was executed.

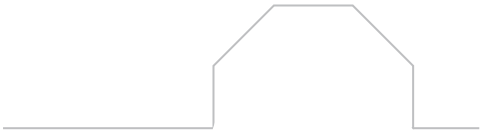
Master shape data



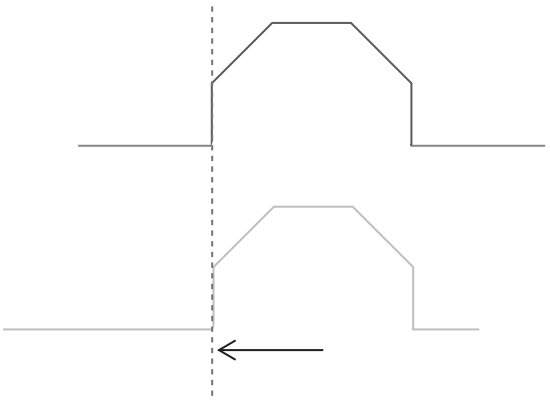
Start position

Measurement data

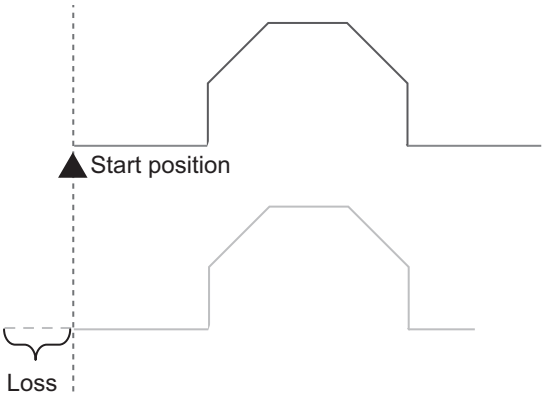
If the position of the target to measure is deviated rightward from the master measurement position,



the measurement data is moved leftward when position correction is performed in order to align the data with the master.



After position correction is completed, the measurement data is aligned with the start position of the master data for conversion to the shape data. At that moment, the height data before the start position is lost, as it cannot be saved in the shape data.



Sample Programming

Program Description

This program performs master measurement and target measurement.

In master measurement, the program uses the LineMeasure_Cartesian function block to create the line measurement data. Then, the program executes the LineMeasure_CreateShape2D_Master function block with the line measurement data and correction parameter to create the master 2D shape data.

For target measurement, you can select from the following two types of methods.

● Method to Create the Target 2D Shape Data without Using the Master 2D Shape Data

The target 2D shape data is created through the following procedure.

1 Execute the LineMeasure_Cartesian function block to create the line measurement data.

2 Execute the LineMeasure_CreateShape2D function block with the line measurement data and correction parameter to create the target 2D shape data.

● Method to Create the Target 2D Shape Data Using the Master 2D Shape Data

The target 2D shape data is created through the following procedure.

1 Create the master 2D shape data.
Execute the LineMeasure_Cartesian function block to create the line measurement data. Then, execute the LineMeasure_CreateShape2D_Master function block to create the master 2D shape data.

2 Create the line measurement data for creating the target 2D shape data.
The resolution is the same as one for the master measurement.

3 Create the target 2D shape data.
Input the following three types of data and execute the LineMeasure_CreateSahpe2D function block.

- The line measurement data that is obtained in step 2.
- The correction parameter same as the master 2D shaped data
- The reference point data that is obtained in step 1.

Preconditions

You must assign the user program that executes the LineMeasure_Cartesian function block in the primary periodic task.

Also, it may take time to process the LineMeaure_CreateShape2D and LineMeasure_CreateShaped2D_master function blocks. Pay attention to the task assignment and setting period for each processing so that the processing time will not exceed the task period. The function blocks are assigned to the following tasks in this sample programming.

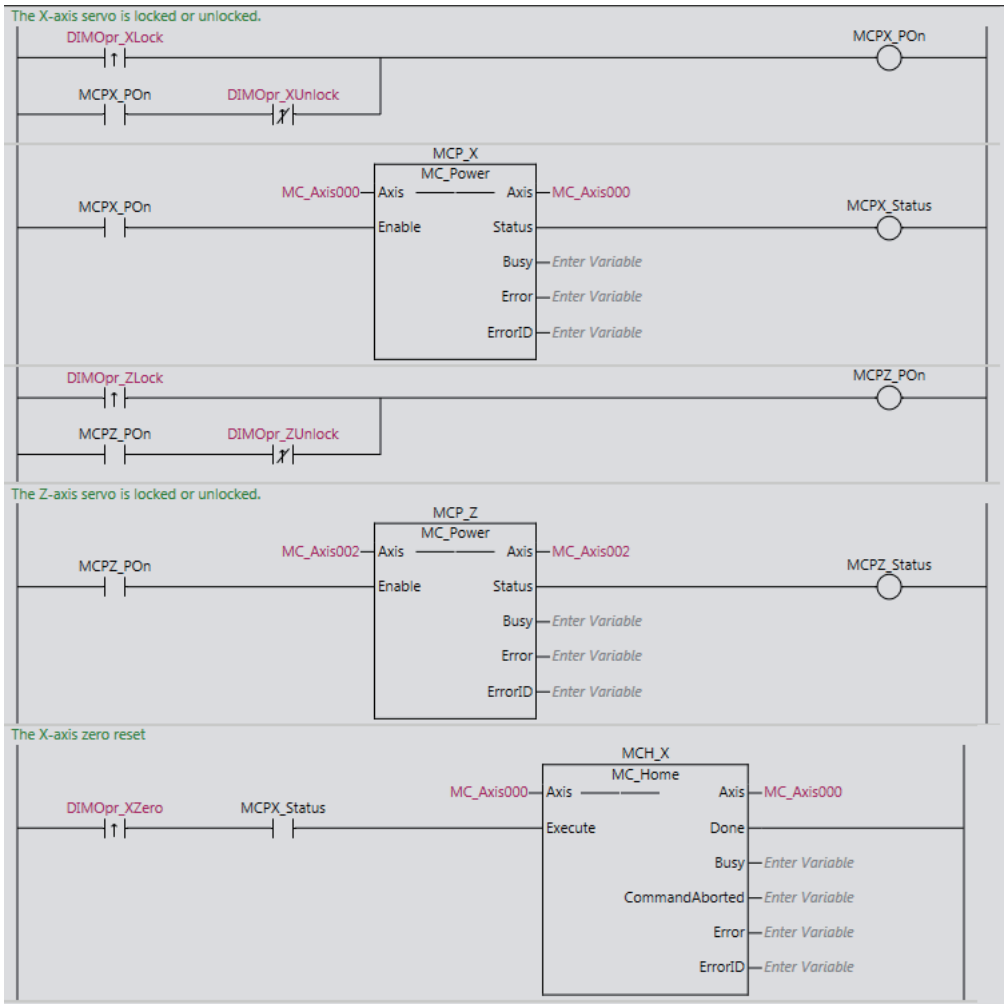
Function block	Assigned task
LineMeasure_Cartesian	Primary periodic task
LineMeasure_CreateShape2D	Periodic task
LineMeasure_CreateShape2D_Master	Periodic task

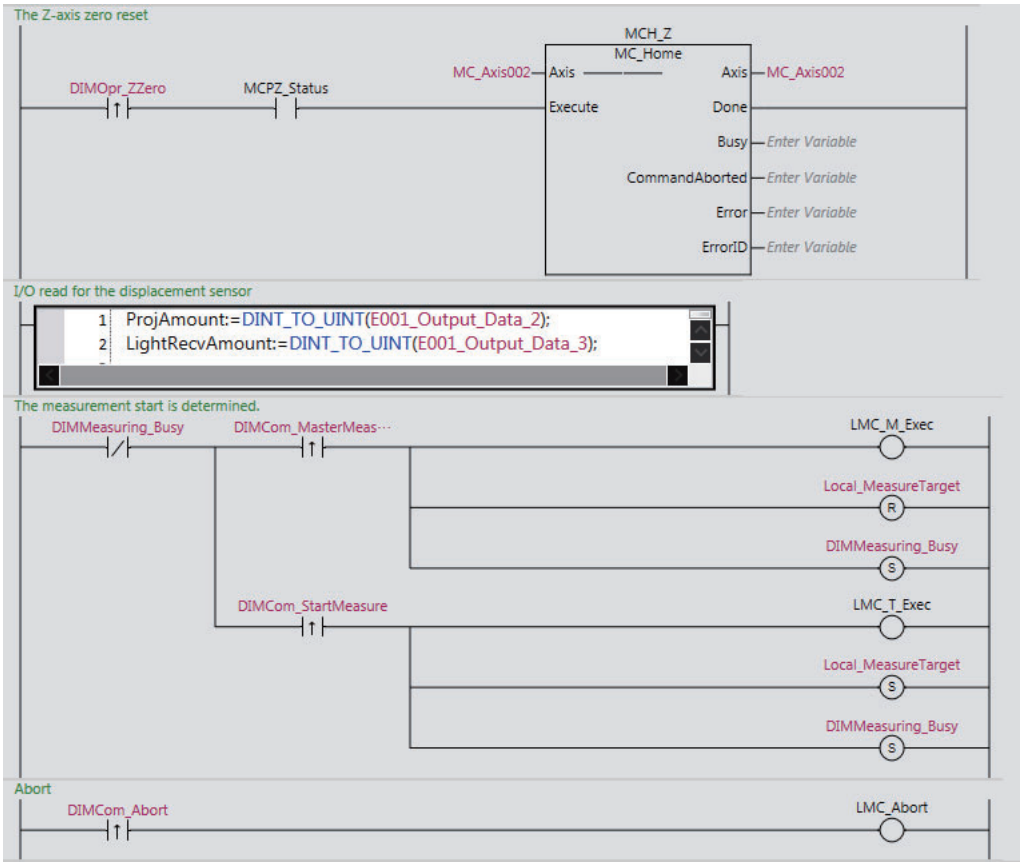
Main Variables

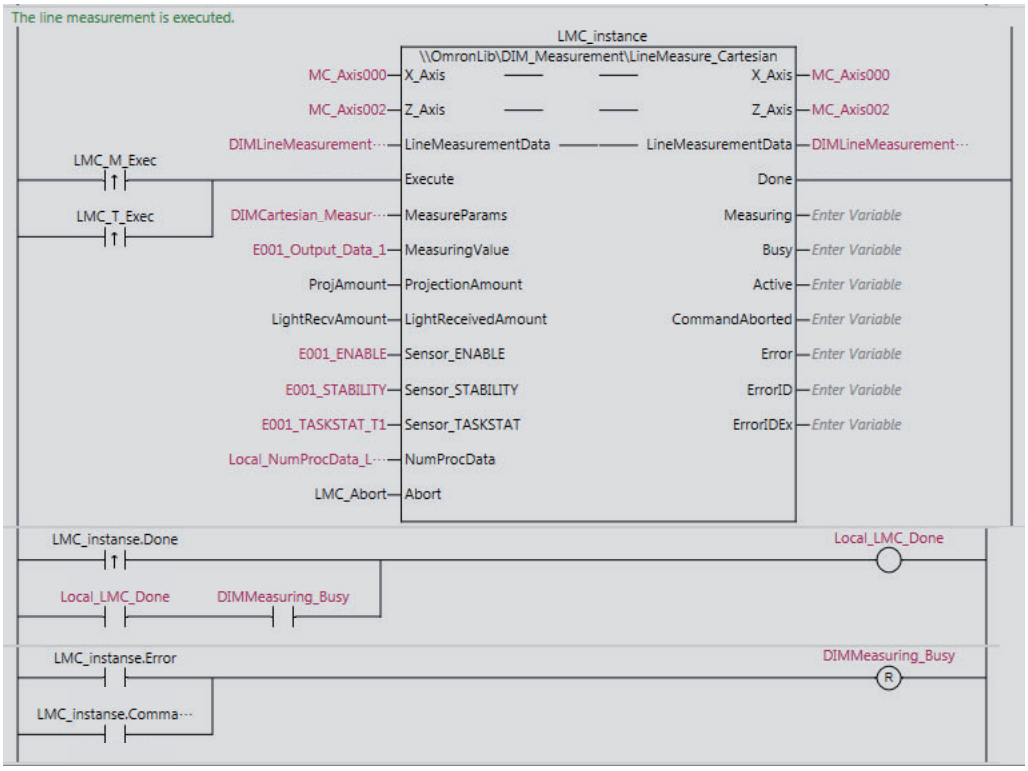
Name	Data type	Comment
DIMCartesian_MeasureParams	OmronLib\DIM_Measurement\sMeasureParams	Set the line measurement parameter.
DIMMeasuring_Busy	BOOL	It is TRUE during a series of process execution from an execution of the line measurement until a creation of the 2D shape data.
DIMCom_MasterMeasure	BOOL	Start trigger for the master measurement.
DIMCom_StartMeasure	BOOL	Start trigger for the target measurement.
Local_MeasureTarget	BOOL	Changes to TRUE when measuring the target and changes to FALSE when measuring the master.
Local_LMC_Done	BOOL	Changes to TRUE when the line measurement is completed and changes to FALSE when the 2D shape data creation is finished. Make the exclusive control setting so that this variable can be changed only in the primary periodic task.
DIMLineMeasurementData	OmronLib\DIM_Measurement\sLineMeasurementData	Line measurement data
DIMShape2D_Master	OmronLib\DIM_Measurement\sShapeMasterData	Master 2D shape data
DIMShape2D_Target	OmronLib\DIM_Measurement\sShape2D	2D shape data
DIMCom_StateMaster	BOOL	When it is TRUE, the target measurement by using the master 2D shape data is performed.
LMC_instance	OmronLib\DIM_Measurement\LineMeasure_Cartesian	Instance of the function block that creates the line measurement data.
ProjAmount	UINT	Values that the light emitting amount of the ZW-7000 are converted to the LREAL data.
LightRecvAmount	UINT	Values that the light receiving amount of the ZW-7000 are converted to the LREAL data.
LMCS_M_instance	OmronLib\DIM_Measurement\LineMeasure_CreateShape2D_Master	Instance of the function block that creates the master 2D shape data.
LMCS_instance	OmronLib\DIM_Measurement\LineMeasure_CreateShape2D	Instance of the function block that creates the 2D shape data.

Ladder Diagram

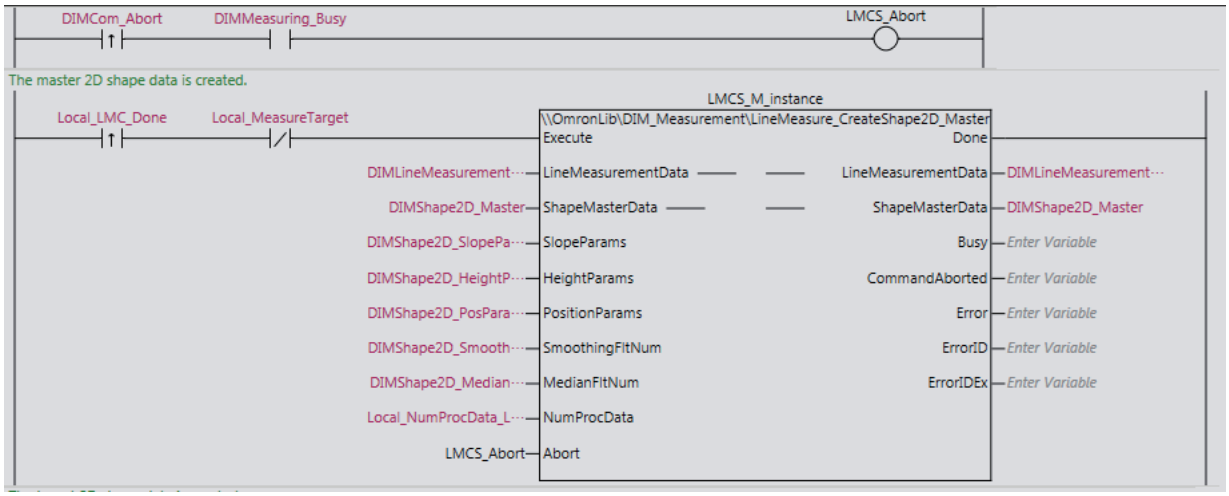
● User Program in the Primary Period Task

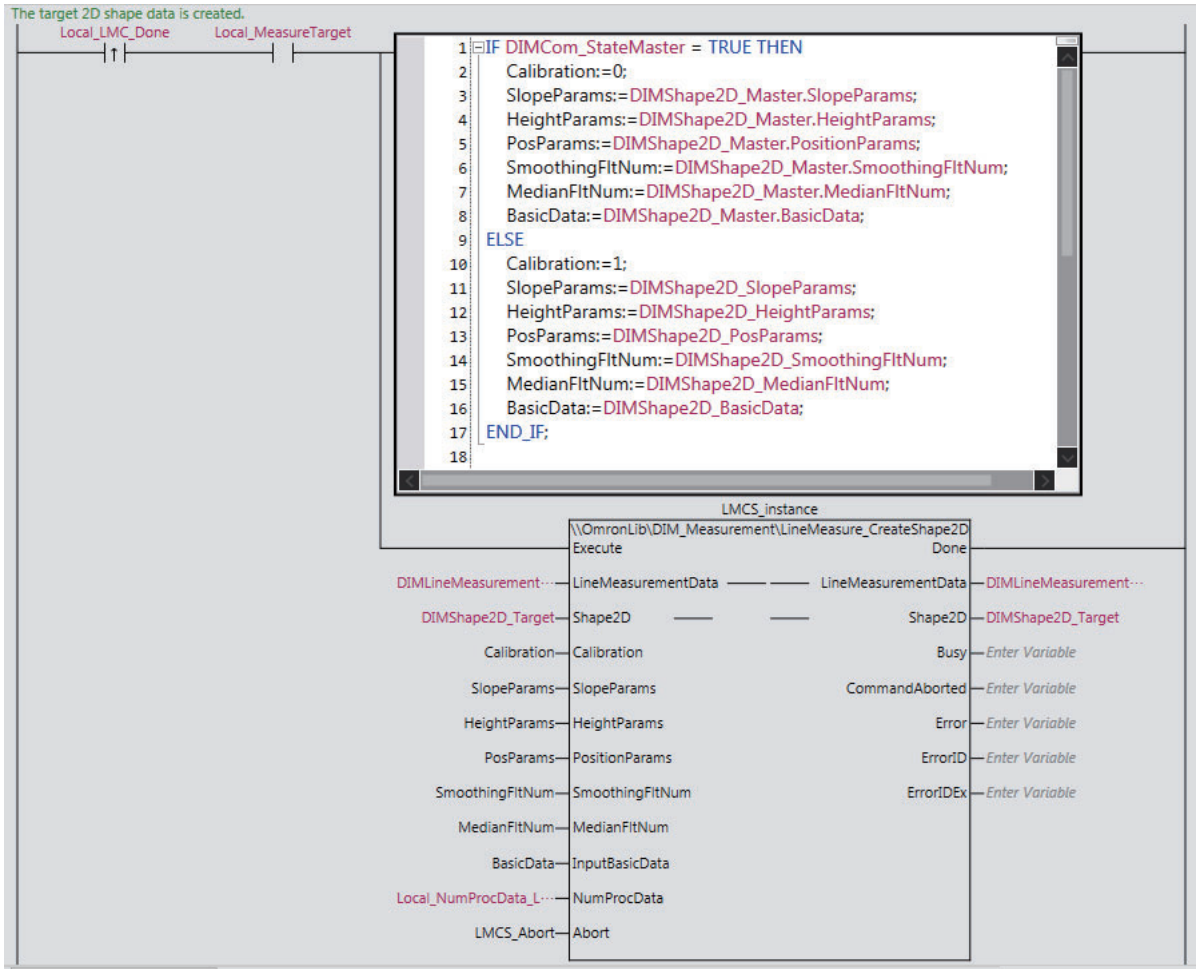


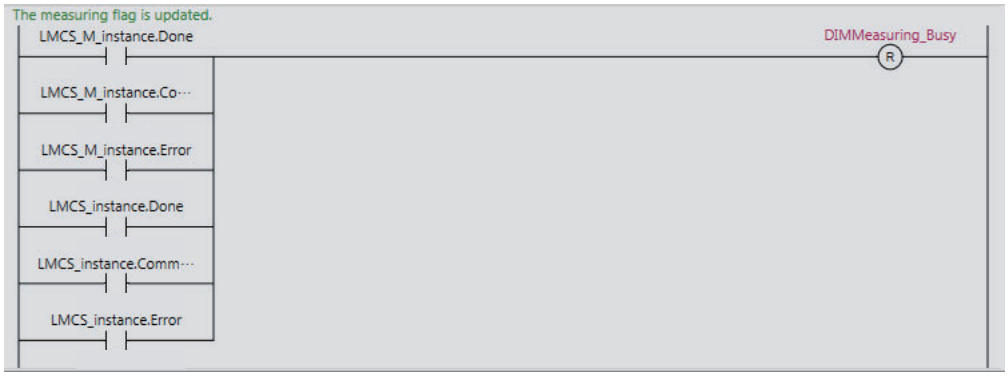

The Z-axis zero reset
MCH_Z
MC_Home
MC_Axis002
Axis
Axis
MC_Axis002
DIMOpr_ZZero
MCPZ_Status
Execute
Done
Busy
Enter Variable
CommandAborted
Enter Variable
Error
Enter Variable
ErrorID
Enter Variable
I/O read for the displacement sensor
1 ProjAmount:=DINT_TO_UINT(E001_Output_Data_2);
2 LightRecvAmount:=DINT_TO_UINT(E001_Output_Data_3);
The measurement start is determined.
DIMMeasuring_Busy
DIMCom_MasterMeas···
LMC_M_Exec
Local_MeasureTarget
R
DIMMeasuring_Busy
S
DIMCom_StartMeasure
LMC_T_Exec
Local_MeasureTarget
S
DIMMeasuring_Busy
S
Abort
DIMCom_Abort
LMC_Abort


The line measurement is executed.
LMC_instance
\\OmronLib\DIM_Measurement\LineMeasure_Cartesian
MC_Axis000
X_Axis
X_Axis
MC_Axis000
MC_Axis002
Z_Axis
Z_Axis
MC_Axis002
DIMLineMeasurement···
LineMeasurementData
LineMeasurementData
DIMLineMeasurement···
LMC_M_Exec
Execute
Done
LMC_T_Exec
DIMCartesian_Measur···
MeasureParams
Measuring
Enter Variable
E001_Output_Data_1
MeasuringValue
Busy
Enter Variable
ProjAmount
ProjectionAmount
Active
Enter Variable
LightRecvAmount
LightReceivedAmount
CommandAborted
Enter Variable
E001_ENABLE
Sensor_ENABLE
Error
Enter Variable
E001_STABILITY
Sensor_STABILITY
ErrorID
Enter Variable
E001_TASKSTAT_T1
Sensor_TASKSTAT
ErrorIDEx
Enter Variable
Local_NumProcData_L···
NumProcData
LMC_Abort
Abort
LMC_instanse.Done
Local_LMC_Done
Local_LMC_Done
DIMMeasuring_Busy
LMC_instanse.Error
DIMMeasuring_Busy
R
LMC_instanse.Comma···

● User Program in the Periodic Task







ST

● User Program in the Primary Periodic Task

```
R_TRIG_LMC_XLock(DIMOpr_XLock,XLock);
R_TRIG_LMC_XUnlock(DIMOpr_XUnLock,XUnLock);
R_TRIG_LMC_ZLock(DIMOpr_ZLock,ZLock);
R_TRIG_LMC_ZUnlock(DIMOpr_ZUnLock,ZUnLock);
R_TRIG_LMC_XZero(DIMOpr_XZero,XZero);
R_TRIG_LMC_ZZero(DIMOpr_ZZero,ZZero);
R_TRIG_LMC_MasterMeasure(DIMCom_MasterMeasure,MasterMeasure);
R_TRIG_LMC_StartMeasure(DIMCom_StartMeasure,StartMeasure);
R_TRIG_DIMCom_Abort(DIMCom_Abort,LMC_Abort);

//The X-axis servo is locked or unlocked.
IF XLock OR ( MCPX_POn AND XUnLock ) THEN
   MCPX_POn:=TRUE;
END_IF;

MCP_X
(
   Axis:=MC_Axis000,
   Enable:=MCPX_POn,
   Status=>MCPX_Status
);

//The Z-axis servo is locked or unlocked.
IF ZLock OR ( MCPZ_POn AND ZUnLock ) THEN
   MCPZ_POn:=TRUE;
END_IF;

MCP_Z
(
   Axis:=MC_Axis002,
   Enable:=MCPZ_POn,
   Status=>MCPZ_Status
);

//The X-axis zero reset
MCH_X
(
   Axis:=MC_Axis000,
   Execute:=(XZero AND MCPX_Status)
);
//The Z-axis zero reset
MCH_Z
(
   Axis:=MC_Axis002,
   Execute:=(ZZero AND MCPZ_Status)
);

//I/O read for the displacement sensor
ProjAmount:=DINT_TO_UINT(E001_Output_Data_2);
LightRecvAmount:=DINT_TO_UINT(E001_Output_Data_3);

//The measurement start is determined.
IF NOT(DIMMeasuring_Busy) AND MasterMeasure THEN
      LMC_M_Exec:=TRUE;
      Local_MeasureTarget:=FALSE;
      DIMMeasuring_Busy:=TRUE;
ELSIF NOT(DIMMeasuring_Busy) AND StartMeasure THEN
      LMC_T_Exec:=TRUE;
      Local_MeasureTarget:=TRUE;
```

```
      DIMMeasuring_Busy:=TRUE;
ELSE
      LMC_M_Exec:=FALSE;
      LMC_T_Exec:=FALSE;
END_IF;

//The line measurement is executed.
LMC_instance
(
X_Axis:=MC_Axis000,
Z_Axis:=MC_Axis002,
LineMeasurementData:=DIMLineMeasurementData,
Execute:=(LMC_M_Exec OR LMC_T_Exec),
MeasureParams:=DIMCartesian_MeasureParams,
MeasuringValue:=E001_Output_Data_1,
ProjectionAmount:=ProjAmount,
LightReceivedAmount:=LightRecvAmount,
Sensor_ENABLE:=E001_ENABLE,
Sensor_STABILITY:=E001_STABILITY,
Sensor_TASKSTAT:=E001_TASKSTAT_T1,
NumProcData:=Local_NumProcData_LMCar,
Abort:=LMC_Abort
);

IF LMC_instance.Done
   OR (Local_LMC_Done AND DIMMeasuring_Busy) THEN
   Local_LMC_Done:=TRUE;
ELSE
   Local_LMC_Done:=FALSE;
END_IF;

IF LMC_instance.Error OR LMC_instance.CommandAborted THEN
   DIMMeasuring_Busy:=FALSE;
END_IF;
```

● User Program in the Periodic Task

```
R_TRIG_LMC_Done(Local_LMC_Done,LMC_Done);
R_TRIG_LMC_Done(DIMCom_Abort,LMCS_Abort);

IF LMCS_Abort AND DIMMeasuring_Busy THEN
   LMCS_Abort:=TRUE;
END_IF;

IF LMC_Done AND NOT(Local_MeasureTarget) THEN
   LMCS_M_Exec:=TRUE;
ELSE
   LMCS_M_Exec:=FALSE;
END_IF;

//The master 2D shape data is created.
LMCS_M_instance
(
   Execute:=LMCS_M_Exec,
   LineMeasurementData:=DIMLineMeasurementData,
   ShapeMasterData:=DIMShape2D_Master,
   SlopeParams:=DIMShape2D_SlopeParams,
   HeightParams:=DIMShape2D_HeightParams,
   PositionParams:=DIMShape2D_PosParams,
   SmoothingFltNum:=DIMShape2D_SmoothingFltNum,
   MedianFltNum:=DIMShape2D_MedianFltNum,
   NumProcData:=Local_NumProcData_LMCre,
   Abort:=LMCS_Abort
);
```

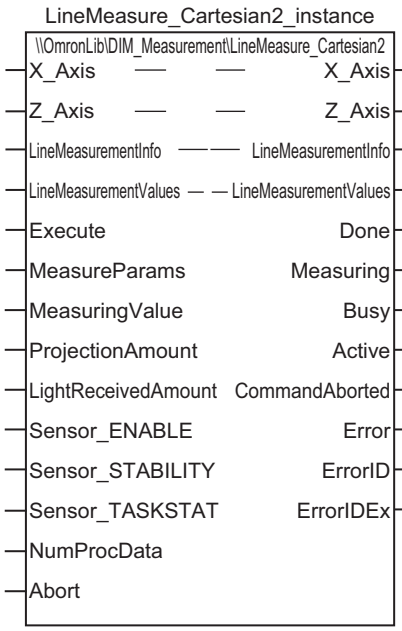
```
//The target 2D shape data is created.

IF LMC_Done AND Local_MeasureTarget THEN
   LMCS_Exec:=TRUE;
   IF DIMCom_StateMaster = TRUE THEN
      Calibration:=0;
      SlopeParams:=DIMShape2D_Master.SlopeParams;
      HeightParams:=DIMShape2D_Master.HeightParams;
      PosParams:=DIMShape2D_Master.PositionParams;
      SmoothingFltNum:=DIMShape2D_Master.SmoothingFltNum;
      MedianFltNum:=DIMShape2D_Master.MedianFltNum;
      BasicData:=DIMShape2D_Master.BasicData;
   ELSE
      Calibration:=1;
      SlopeParams:=DIMShape2D_SlopeParams;
      HeightParams:=DIMShape2D_HeightParams;
      PosParams:=DIMShape2D_PosParams;
      SmoothingFltNum:=DIMShape2D_SmoothingFltNum;
      MedianFltNum:=DIMShape2D_MedianFltNum;
      BasicData:=DIMShape2D_BasicData;
   END_IF;
ELSE
   LMCS_Exec:=FALSE;
END_IF;


LMCS_instance
(
   Execute:=LMCS_Exec,
   LineMeasurementData:=DIMLineMeasurementData,
   Shape2D:=DIMShape2D_Target,
   SlopeParams:=SlopeParams,
   HeightParams:=HeightParams,
   PositionParams:=PosParams,
   SmoothingFltNum:=SmoothingFltNum,
   MedianFltNum:=MedianFltNum,
   InputBasicData:=BasicData,
   NumProcData:=Local_NumProcData_LMCre,
   Abort:=LMCS_Abort
);


// The measuring flag is updated.

IF LMCS_M_instance.Done OR LMCS_M_instance.CommandAborted OR LMCS_M_instance.Error
   OR LMCS_instance.Done OR LMCS_instance.CommandAborted OR LMCS_instance.Error THEN
   DIMMeasuring_Busy:=FALSE;
END_IF;
```

LineMeasure_Cartesian2

The LineMeasure_Cartesian2 function block measures the height of measurement surfaces while moving a single axis with a displacement sensor in X and Z directions.

From the measurement result, this function block creates the line measurement data array *LineMeasurementValues[]*. *LineMeasurementValues[]* is a variable-length array here.

You can select from the following two methods to control the single axis during measurement: Surface Search and Tracer Control.

Function block and function name	Name	FB/ FUN	Graphic expression	ST expression
LineMeasure_Cartesian2	Line Measurement with Cartesian Coordinate System2 (Surface Search / Tracer Control)	FB		LineMeasure_Cartesian2_instance(X_Axis, Z_Axis, LineMeasurementInfo, LineMeasurementValues, Execute, MeasureParams, MeasuringValue, ProjectionAmount, LightReceivedAmount, Sensor_ENABLE, Sensor_STABILITY, Sensor_TASKSTAT, NumProcData, Abort, Done, Measuring, Busy, Active, CommandAborted, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V2_0.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00227
Source code	Not published

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
MeasureParams*1	Measurement parameter	Set the measurement parameter.	sMeasureParams	—	—
MeasuringValue	Displacement Sensor Measurement Value	Assign the measurement value that is output from the displacement sensor. The unit is [nm].	DINT	Depends on data type.	0
ProjectionAmount	Displacement Sensor Amount of Emitted Light	Assign the amount of emitted light that is output from the displacement sensor.	UINT	Depends on data type.	0
LightReceivedAmount	Displacement Sensor Amount of Light Received	Assign the amount of received light that is output from the displacement sensor.	UINT	Depends on data type.	0
Sensor_ENABLE	Displacement Sensor ENABLE output	Assign the ENABLE for the displacement sensor output.	BOOL	TRUE or FALSE	FALSE
Sensor_STABILITY	Displacement Sensor STABILITY output	Assign the STABILITY for the displacement sensor output.	BOOL	TRUE or FALSE	FALSE
Sensor_TASKSTAT	Displacement Sensor TASKSTAT output	Assign the TASKSTAT for the displacement sensor output.	BOOL	TRUE or FALSE	FALSE
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a period during segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
Measuring	Measuring	TRUE when measurement is in progress. FALSE when data is outside the measurement range or when re-positioning is in progress.	BOOL	TRUE or FALSE	—
Busy	Executing	TRUE when *Execute* is acknowledged.	BOOL	TRUE or FALSE	—
Active	Controlling	TRUE when control is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 166 for details.

In-Out Variables

Variable	Name	Description	Data type	Valid range
X_Axis	X axis	Specify the X axis.*2 Specify "Servo Axis".	_sAXIS_REF	—
Z_Axis	Z axis	Specify the Z axis.*2 Specify "Servo Axis".	_sAXIS_REF	—
LineMeasurementInfo	Line Measurement Data Information	The line measurement data information is recorded.	OmronLib\DIM_Measurement\sLineMeasurementInfo	—
LineMeasurementValues []*1	Line Measurement Data Array	The line measurement data is recorded.	ARRAY[*] OF OmronLib\DIM_Measurement\sLineMeasurementValue	—

*1. The maximum number of array elements is 20,000. The first number of array element should be 0.

*2 Specify a user-defined Axis Variable that was created in Axis Basic Settings of the Sysmac Studio (default: *MC_Axis****) or a system-defined axis variable name (*_MC_AX[*]*, *_MC1_AX[*]*, or *_MC2_AX[*]*).

Members of Structure sMeasureParams

Member	Name	Data type	Valid range	Description
CtrlType	Control Type	USINT	0, 1	0: Surface Search 1: Tracer Control
X_Prepare Position	X-axis Preparation Position	LREAL	Negative number, positive number, or 0	Set the measurement preparation position. The unit is [command unit].*1
X_Target Position	X-axis Target Position	LREAL	Negative number, positive number, or 0	Set the measurement target position. The unit is [command unit].*1
X_Start Position	X-axis Measurement Start Position	LREAL	Negative number, positive number, or 0	Set the measurement start position. The unit is [command unit].*1
X_EndPosition	X-axis Measurement End Position	LREAL	Negative number, positive number, or 0	Set the measurement end position. The unit is [command unit].*1
X_Prepare Velocity	X-axis Preparation Velocity	LREAL	Positive number	Set the velocity for the X-axis movement to the preparation position. The unit is [command unit/s].*1
X_Measuring Velocity	X-axis Measuring Velocity	LREAL	Positive number	Set the velocity for the X-axis measuring movement. The unit is [command unit/s].*1
X_Acceleration	X-axis Acceleration	LREAL	Positive number or 0	Set the acceleration for the X-axis movement. The unit is [command unit/s^2].*1
X_Deceleration	X-axis Deceleration	LREAL	Positive number or 0	Set the deceleration for the X-axis movement. The unit is [command unit/s^2].*1
X_Jerk	X-axis Jerk	LREAL	Positive number or 0	Set the jerk for the X-axis movement. The unit is [command unit/s^3].*1
Z_Retract Position	Z-axis Retraction Position	LREAL	Negative number, positive number, or 0	Set the position for the Z-axis retraction. The unit is [command unit].*1
Z_SearchEnd Position	Z-axis Search End Position	LREAL	Negative number, positive number, or 0	Set the position at which the displacement sensor ends a search for a position that allows measurement. The search range is a range between the retraction position and search end position. If measurement is not enabled within the search range, another search starts. The unit is [command unit].*1
Z_Velocity	Z-axis Velocity	LREAL	Positive number	Set the velocity for the Z-axis movement. The unit is [command unit/s].*1
Z_Acceleration	Z-axis Acceleration	LREAL	Positive number or 0	Set the acceleration for the Z-axis movement. The unit is [command unit/s^2].*1

Member	Name	Data type	Valid range	Description
Z_Deceleration	Z-axis Deceleration	LREAL	Positive number or 0	Set the deceleration for the Z-axis movement. The unit is [command unit/s^2].*1
Z_Jerk	Z-axis Jerk	LREAL	Positive number or 0	Set the jerk for the Z-axis movement. The unit is [command unit/s^3].*1
Resolution	Measurement Resolution	UINT	Positive number	Measurement is performed for each specified distance. The unit is [μm].
InPosition	Measurement In-position	UINT	Positive number or 0	Set the in-position width for the target position at which the measurement value is recorded. The unit is [μm].
UseMotionCmd	Motion Control Method for Tracer Control	USINT	0	Set how to control the motion of the single-axis during the tracer control. 0: Standard tracer control 1 or larger: (Reserved)
CopyErr TolerateNum	Tolerable Number of Measurement Errors in Tracer Control	USINT	0	Reserved

*1. Refer to *Unit Conversion Settings* in the motion control user's manual for information on command units.

Members of Structure sLineMeasurementInfo

Member	Name	Data type	Valid range	Description
DataCount	Number of Recorded Data	UINT	0 to 20,000	The number of measurement data recorded in *LineMeasurementValues[]*.
Resolution	Resolution	UINT	Positive number	The X-direction resolution that is used during measurement is stored. The unit is [μm].

Members of Structure sLineMeasurementValue

Member	Name	Data type	Valid range	Description
XValue	X Measurement Data	LREAL	Depends on data type.	The measurement positions are recorded. The unit is mm.
ZValue	Z Measurement Data	LREAL	Depends on data type.	The heights measured at the measurement positions are recorded. The unit is mm.
Projection Amount	Amount of Light Emitted Data	UINT	Depends on data type.	The amount of light emitted at the measurement positions are recorded.
LightReceivedAmount	Amount of Light Received Data	UINT	Depends on data type.	The amount of light received at the measurement positions are recorded.

Function

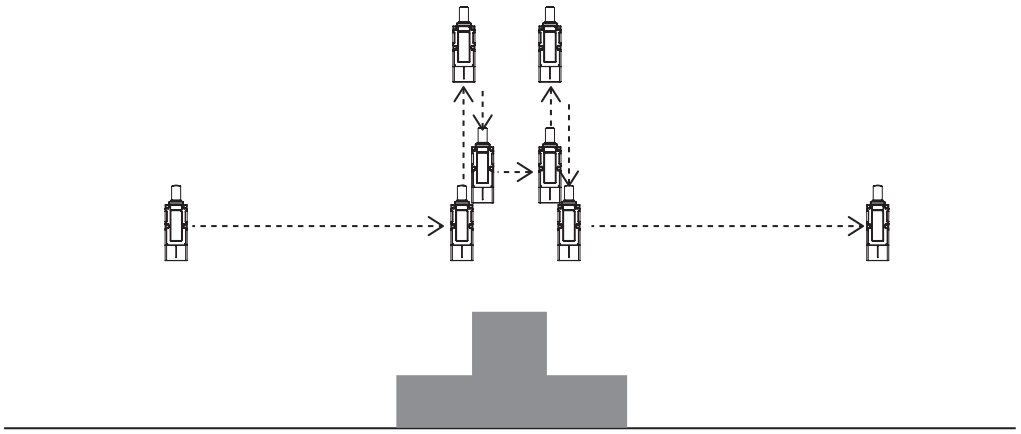
The LineMeasure_Cartesian2 function block measures the height of measurement surfaces while moving a single axis with a displacement sensor in X and Z directions.

From the measurement result, this function block creates the line measurement data array *LineMeasurementValues[]*.

● Shape measurement using the Surface Search (searching a measurement surface)

The height of the displacement sensor is fixed within the measurement range. The sensor scans and measures the target surfaces that come within the measurement range.

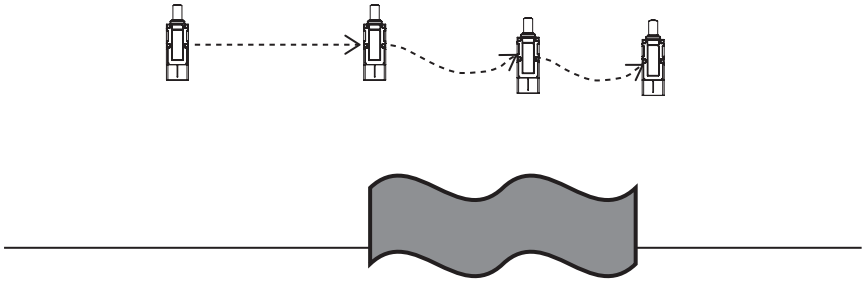
When the height of the measurement surface changes and it goes outside the measurement range of the displacement sensor, the height of the displacement sensor is readjusted and the measurement is continued. This method is suitable for measuring shapes whose height varies greatly and sharply.



● Shape Measurement using the Tracer Control

During measurement, the height of the displacement sensor is adjusted continuously to keep a constant distance between the sensor and measurement surface.

If the height of the measurement surface changes sharply and the measurement in the tracer control cannot be continued, the height is readjusted in the same way as the surface search so that the measurement in the tracer control can be resumed. This method is suitable for measuring shapes whose height varies gradually.



To obtain accurate line measurement data, always execute this function block in the primary periodic task.

(a) You can select from the following two methods to control the single axis during measurement: Surface Search and Tracer Control.

(b) The function block checks the input parameters when *Execute* changes to TRUE. If there is no error in the input parameters, either the Surface Search or Tracer Control is started. If there is an error in the input parameters, the measurement control is not performed, and the function block execution is aborted.
For details on the error codes, refer to *Troubleshooting* on page 166.

(c) For the unit of X-axis and Z-axis display, select “mm”, “µm”, “nm”, or “inch”.
An error end occurs if “pulse” or “degree” is selected.
The unit of axis display is not reflected in the measurement data.
The unit for the measurement data is always “mm”.

(d) An error occurs if the X-axis travel velocity (travel distance per period in the primary periodic task) exceeds the resolution.
Refer to *Surface Search* on page 158 for details.

(e) When *CtrlType* (Control Type) is 0, measurement by the surface search method is performed.
Refer to *Surface Search* on page 158 for function details on the Surface Search.

(f) When *CtrlType* (Control Type) is 1, measurement by the tracer control method is performed.
Refer to *Tracer Control* on page 158 for function details on the Tracer Control.

(g) Even when the measurement direction (travel direction along the X axis) is negative (*X_StartPosition* > *X_EndPosition*), the line measurement data is output in ascending order of X value.
Sort in ascending order is performed for each unit of the processed data points specified in *NumProcData*.
So, to prevent a Task Period Exceeded error, specify a value for this variable according to the time of the task period to which this function block is assigned.

(h) If *Abort* is changed to TRUE and motion control is in progress, the MC_Stop instruction is executed to stop axis operation.

(i) If an error occurs, *Error* changes to TRUE and the function block execution is aborted. In addition, the error codes are output to *ErrorID* and *ErrorIDEx*.
For details on the error codes, refer to *Troubleshooting* on page 166.

(j) If the function block execution is aborted or ended in an error, the line measurement data will not be created correctly.

(k) In the last stage of processing this FB, the line measurement data is sorted in ascending order. In the sort processing, the number of processed data points (*NumProcData*) per control period can be set.
When *NumProcData* = 0, sorting is completed in one control period.

Input Variables to Set Values Output from the Displacement Sensor

Refer to *Input Variables to Set Values Output from the Displacement Sensor* on page 116 in LineMeasure_Cartesian.

Process Flow from Creating Line Measurement Data to Creating 2D Shape Data

Refer to *Process Flow from Creating Line Measurement Data to Creating 2D Shape Data* on page 117 in LineMeasure_Cartesian.

Line Measurement Data

Refer to *Line Measurement Data* on page 123 in LineMeasure_Cartesian.

2D Shape Data

Refer to *2D Shape Data* on page 123 in LineMeasure_Cartesian.

Master 2D Shape Data

Refer to *Master 2D Shape Data* on page 123 in LineMeasure_Cartesian.

Measurement Resolution and Measurement Record Position

Refer to *Measurement Resolution and Measurement Record Position* on page 124 in LineMeasure_-Cartesian.

Surface Search

Refer to *Surface Search* on page 126 in LineMeasure_Cartesian.

Tracer Control

Refer to *Tracer Control* on page 128 in LineMeasure_Cartesian.

Re-execution of Function Blocks

If a function block is re-executed when execution is in progress (*Busy* = TRUE), an error end occurs.

The measurement execution is terminated.

When you re-execute the function block, change *Abort* to TRUE to terminate the execution, and then change *Execute* to TRUE.

Multi-execution of Function Blocks

For details on multi-execution of function blocks, refer to the motion control user's manual.

Timing Charts

The timing charts are shown below.

● In a Normal State (Surface Search or Tracer Control)

When *Execute* changes to TRUE, *Busy* (Executing) changes to TRUE.

When the internal motion function block is executed, *Active* (Controlling) changes to TRUE.

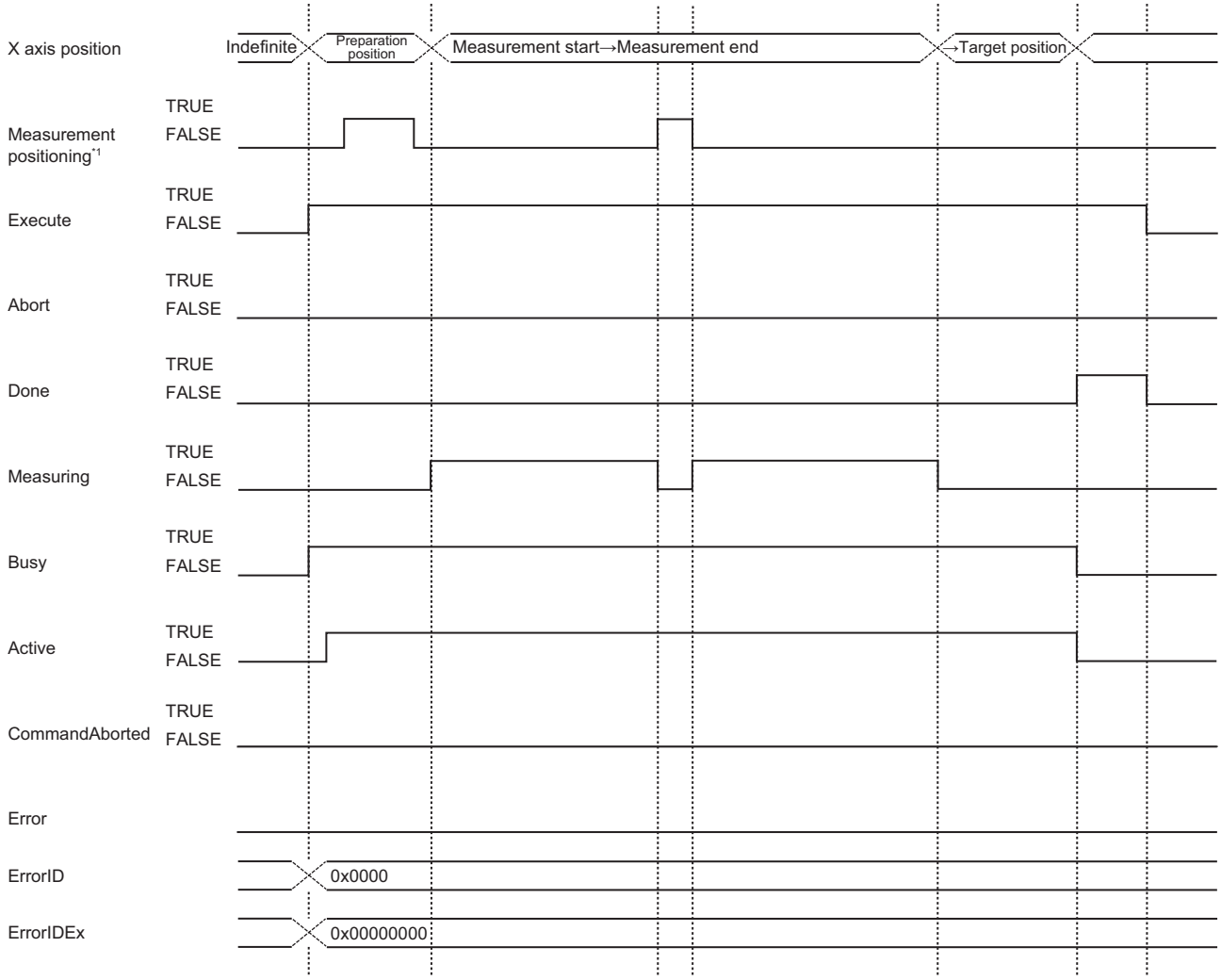
Measuring is TRUE while the single axis is in the measurement range and shape measurement is performed.

Measuring is FALSE while a state of measurement disabled is detected and the single axis is re-positioned.

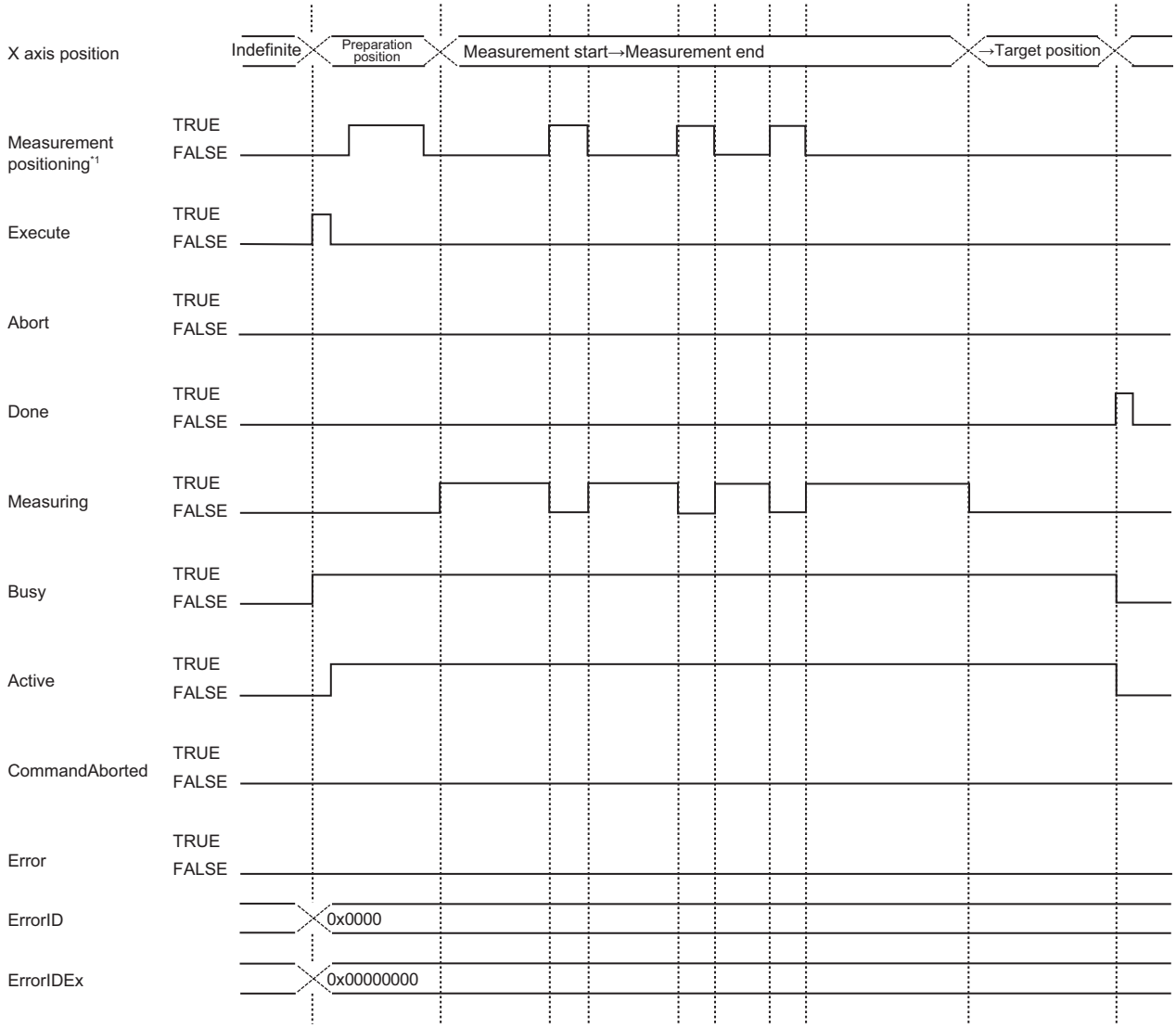
When the single axis reaches the target position and the measurement is completed, *Busy* and *Active* change to FALSE, and *Done* changes to TRUE.

Done holds its value while *Execute* remains TRUE.

If *Execute* is FALSE and *Done* changes to TRUE, *Done* changes to TRUE for only one task period.



*1. Refer to *Surface Search* on page 126 for information on measurement positioning.

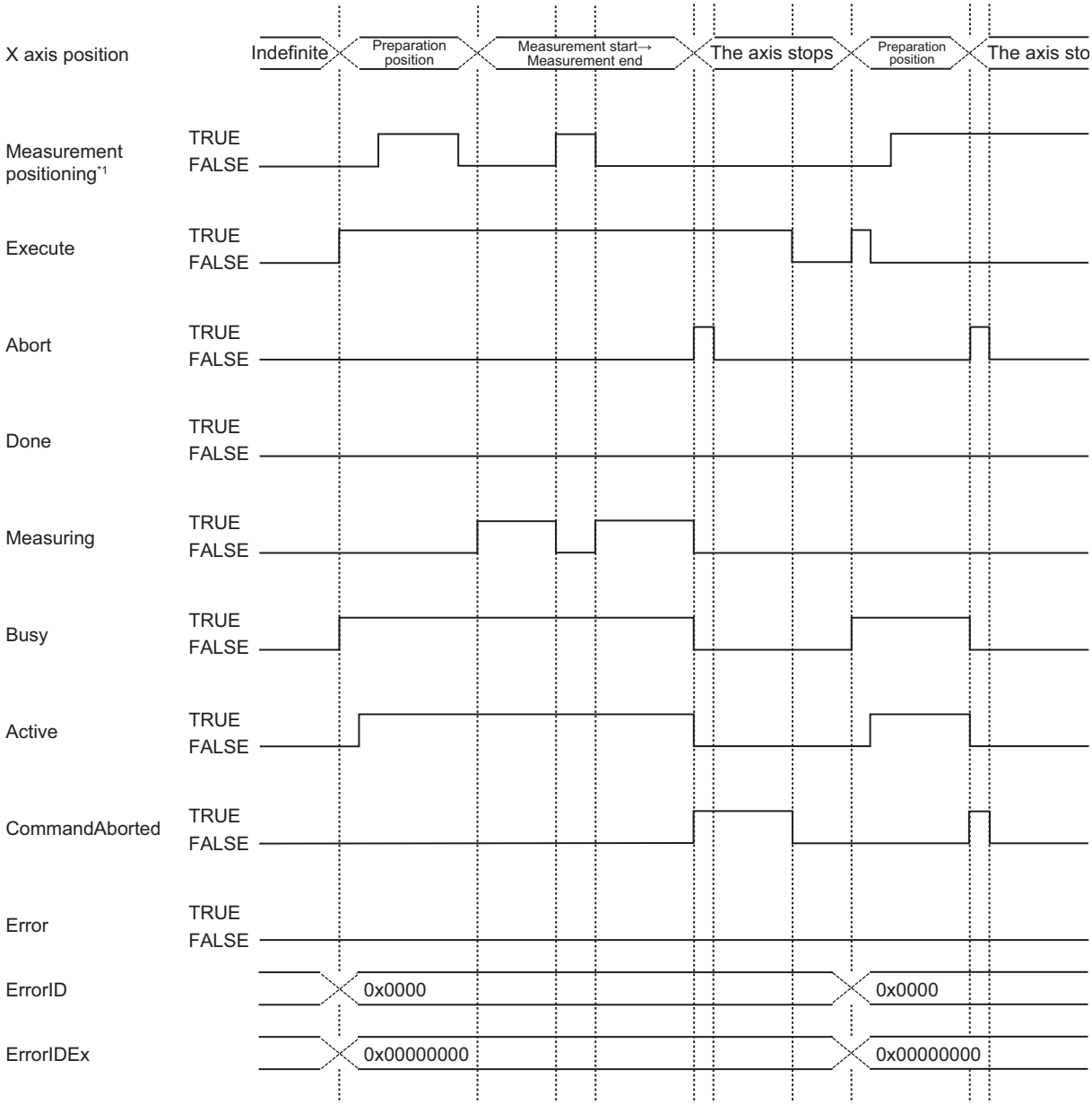


*1. Refer to *Surface Search* on page 126 for information on measurement positioning.

● When the Function Block Is Aborted

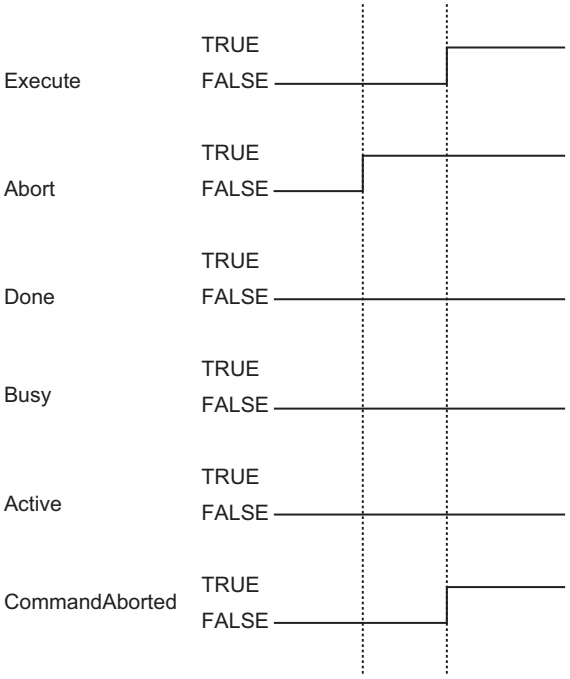
If *Abort* changes to TRUE when execution is in progress, *Active* changes to FALSE and *Command-Aborted* (Interruption Completion) changes to TRUE.

CommandAborted holds its value while *Execute* remains TRUE.



*1. Refer to *Surface Search* on page 126 for information on measurement positioning.

If *Execute* changes to TRUE while *Abort* is TRUE, *CommandAborted* (Interruption Completion) changes to TRUE without performing the processing.



When an Error Occurs (Execution Check Error)

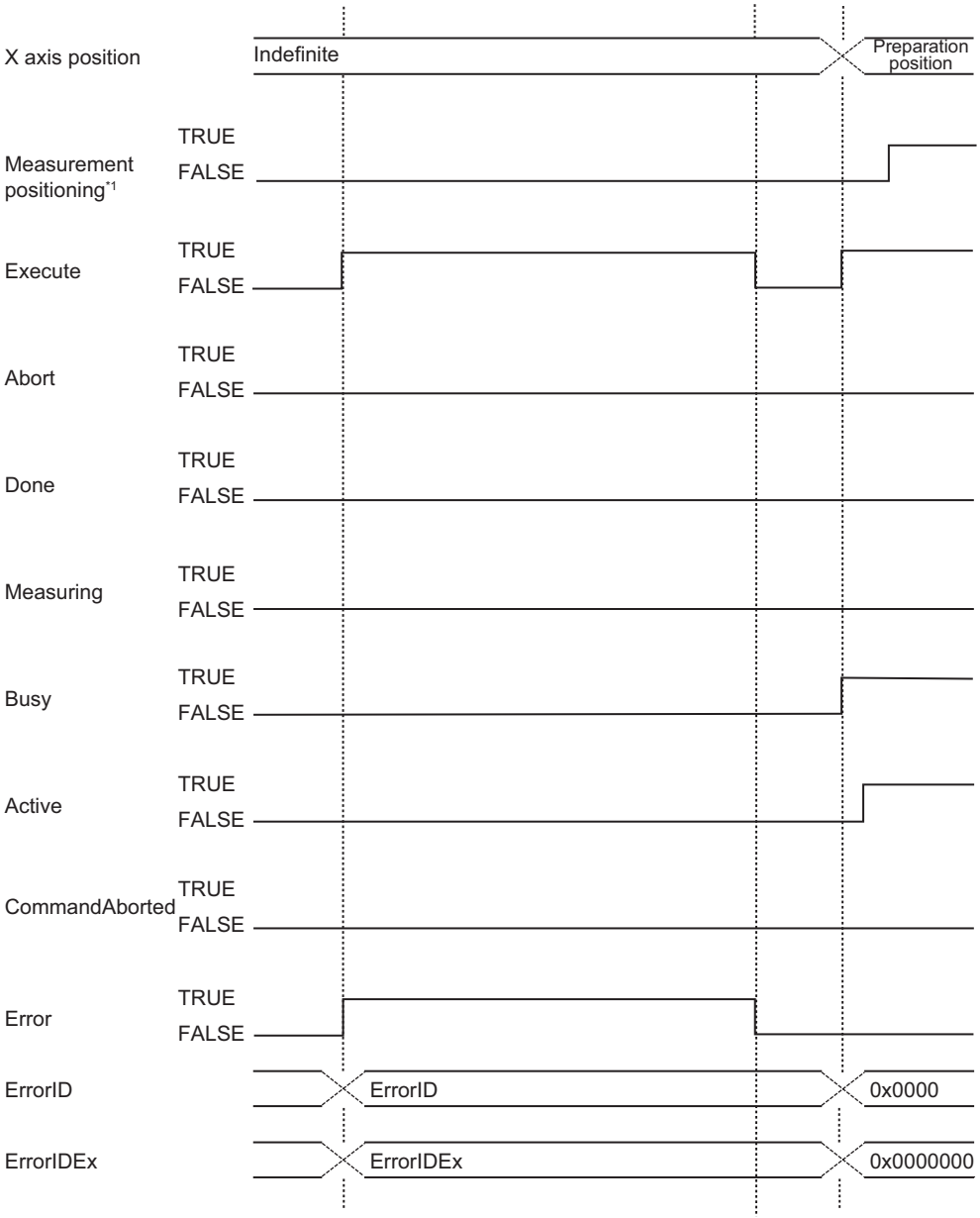
If an error is detected during the check performed when function block is executed, *Error* changes to TRUE.

The values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code).

Busy and *Active* remain FALSE.

Error holds its value while *Execute* remains TRUE.

ErrorID and *ErrorIDEx* hold their values until the function block is re-executed.



*1. Refer to *Surface Search* on page 126 for information on measurement positioning.

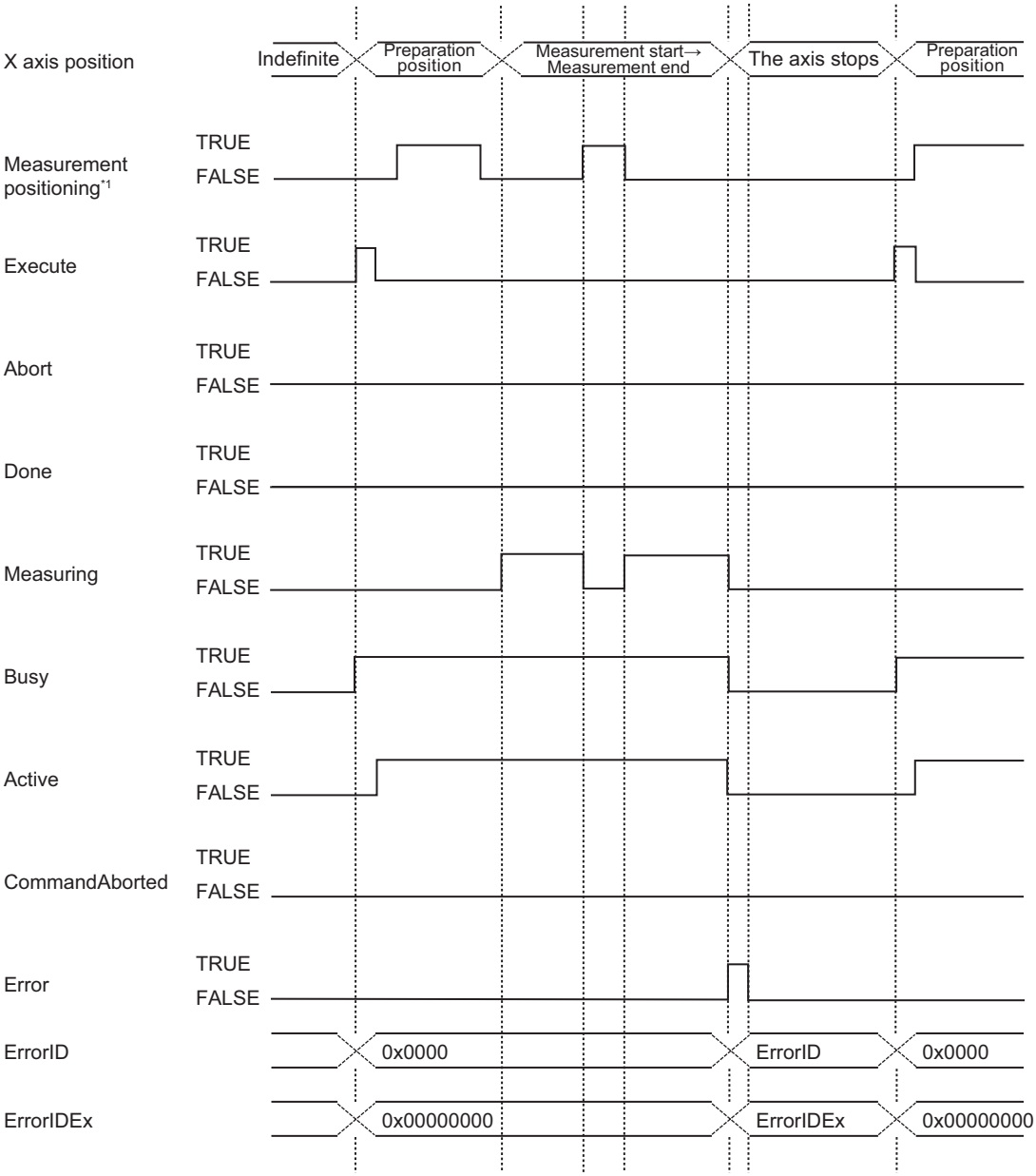
When an Error Occurs (Execution Error)

If an error occurs during execution of this function block, *Busy* and *Active* change to FALSE and *Error* changes to TRUE.

The values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code).

Error holds its value while *Execute* remains TRUE.

ErrorID and *ErrorIDEx* hold their values until the function block is re-executed.



*1. Refer to *Surface Search* on page 126 for information on measurement positioning.

Precautions for Correct Use

- For the settings of a displacement sensor, do not use the filter (median filter, average filter, or low-pass filter) in the output condition settings for Task 1 that outputs the height in principle. If the filter is used, a correct measurement result may not be obtained, or operation may become unstable.
- For details on filtering the measurement data, refer to the descriptions in sections of LineMeasure_CreateShape2DMaster2 and LineMeasure_CreateShape2D2.
- Always execute this function block in the primary periodic task. If you execute it in the periodic task, a correct measurement result may not be obtained.
- The 2D shape data creation function blocks (LineMeasure_Shape2DMaster2 and LineMeasure_Shape2D2) use the line measurement data as the in-output variables. These function blocks must be executed only after this function block is completed normally.
- Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x0000	0x00000000	Normal end	—
0x3CD6	0x00000001	An X-axis error occurred.	*1
	0x00000002	An Z-axis error occurred.	*1
	0x00000003	The function block was executed in the event task.	Execute this function block in the primary periodic task.
	0x00000004	The unit of X-axis display is set to “pulse” or “degree”.	Select one of “mm”, “µm”, “nm”, or “inch”.
	0x00000005	The unit of Z-axis display is set to “pulse” or “degree”.	Select one of “mm”, “µm”, “nm”, or “inch”.
	0x00000006	The undefined Control Type (*CtrlType*) was specified.	Specify a value which is within the valid range.
	0x00000007	The value specified for X-axis Preparation Position (*X_PreparePosition*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000008	The value specified for X-axis Target Position (*X_TargetPosition*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000009	The value specified for X-axis Measurement Start Position (*X_StartPosition*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000000A	The value specified for X-axis Measurement End Position (*X_EndPosition*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000000B	The measurement range is outside the X-axis travel range.	Specify the values so that they meet the following condition. X-axis Preparation Position ≤ Measurement Start Position < Measurement End Position ≤ Target Position Or, X-axis Preparation Position ≥ Measurement Start Position > Measurement End Position ≥ Target Position
	0x0000000C	The value specified for Z-axis Retraction Position (*Z_RetractPositon*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000000D	The value specified for Z-axis Search End Position (*Z_SearchEndPosition*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000000E	The value specified for Measurement Resolution (*Resolution*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000000F	The value specified for Measurement In-position is equal to or larger than Measurement Resolution.	Specify a value which is smaller than Measurement Resolution.

Error code	Expansion error code	Description	Corrective action
0x3CD6	0x00000010	The value specified for Motion Control Method for Tracer Control (*UseMotionCmd*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000011	The X-axis velocity is too high relative to the resolution.	Adjust the X-axis velocity, resolution, and task period in which the function block is executed.
	0x00000012	The number of measured and recorded points (measurement range divided by resolution) exceeded the capacity of the measurement data.	Correct the measurement range and resolution.
	0x00000013	Servo is not turned ON for X axis.	Turn ON the servo. When “Virtual Servo Axis” is set for the X axis Set the “Servo Axis”.
	0x00000014	Home is not defined for X axis.	Define home before executing the function block.
	0x00000015	Servo is not turned ON for Z axis.	Turn ON the servo. When “Virtual Servo Axis” is set for the Z axis Set the “Servo Axis”.
	0x00000016	Home is not defined for Z axis.	Define home before executing the function block.
	0x00000017	Z-axis measurement position setting failed at the X-axis Preparation Position.	Adjust the measurement parameters (X-axis preparation position, Z-axis Temporary Position, and Threshold). Also, check that the measurement of the target is enabled.
	0x00000018	The function block was re-executed when execution is in progress.	Change *Abort* to TRUE to terminate the execution, and then re-execute the function block.
	0x00000019	The specified array for the line measurement data array (*LineMeasurementValues*) is out of the valid range.	The first number of array elements should be 0. Specify a value that the number of array elements is 20,000 or less.

*1. Refer to the motion control user’s manual.

Restriction

Refer to *Restriction* on page 139 in LineMeasure_Cartesian.

Sample Programming

Program Description

This program performs master measurement and target measurement.

In master measurement, the program uses the LineMeasure_Cartesian2 function block to create the line measurement data. Then, the program executes the LineMeasure_CreateShape2D_Master2 function block with the line measurement data and correction parameter to create the master 2D shape data.

For target measurement, you can select from the following two types of methods.

● Method to Create the Target 2D Shape Data without Using the Master 2D Shape Data

The target 2D shape data is created through the following procedure.

1 Execute the LineMeasure_Cartesian2 function block to create the line measurement data.

2 Execute the LineMeasure_CreateShape2D2 function block with the line measurement data and correction parameter to create the target 2D shape data.

● Method to Create the Target 2D Shape Data Using the Master 2D Shape Data

The target 2D shape data is created through the following procedure.

1 Create the master 2D shape data.
Execute the LineMeasure_Cartesian2 function block to create the line measurement data. Then, execute the LineMeasure_CreateShape2D_Master2 function block to create the master 2D shape data.

2 Create the line measurement data for creating the target 2D shape data.
The resolution is the same as one for the master measurement.

3 Create the target 2D shape data.
Input the following three types of data and execute the LineMeasure_CreateShape2D2 function block.

- The line measurement data that is obtained in step 2.
- The correction parameter is the same as one for the master 2D shape data.
- The reference point data that is obtained in step 1.

Preconditions

You must assign the user program that executes the LineMeasure_Cartesian2 function block in the primary periodic task.

Also, it may take time to process the LineMeasure_CreateShape2D2 and LineMeasure_CreateShape2D_Master2 function blocks. Pay attention to the task assignment and setting period for each processing so that the processing time will not exceed the task period. The function blocks are assigned to the following tasks in this sample programming.

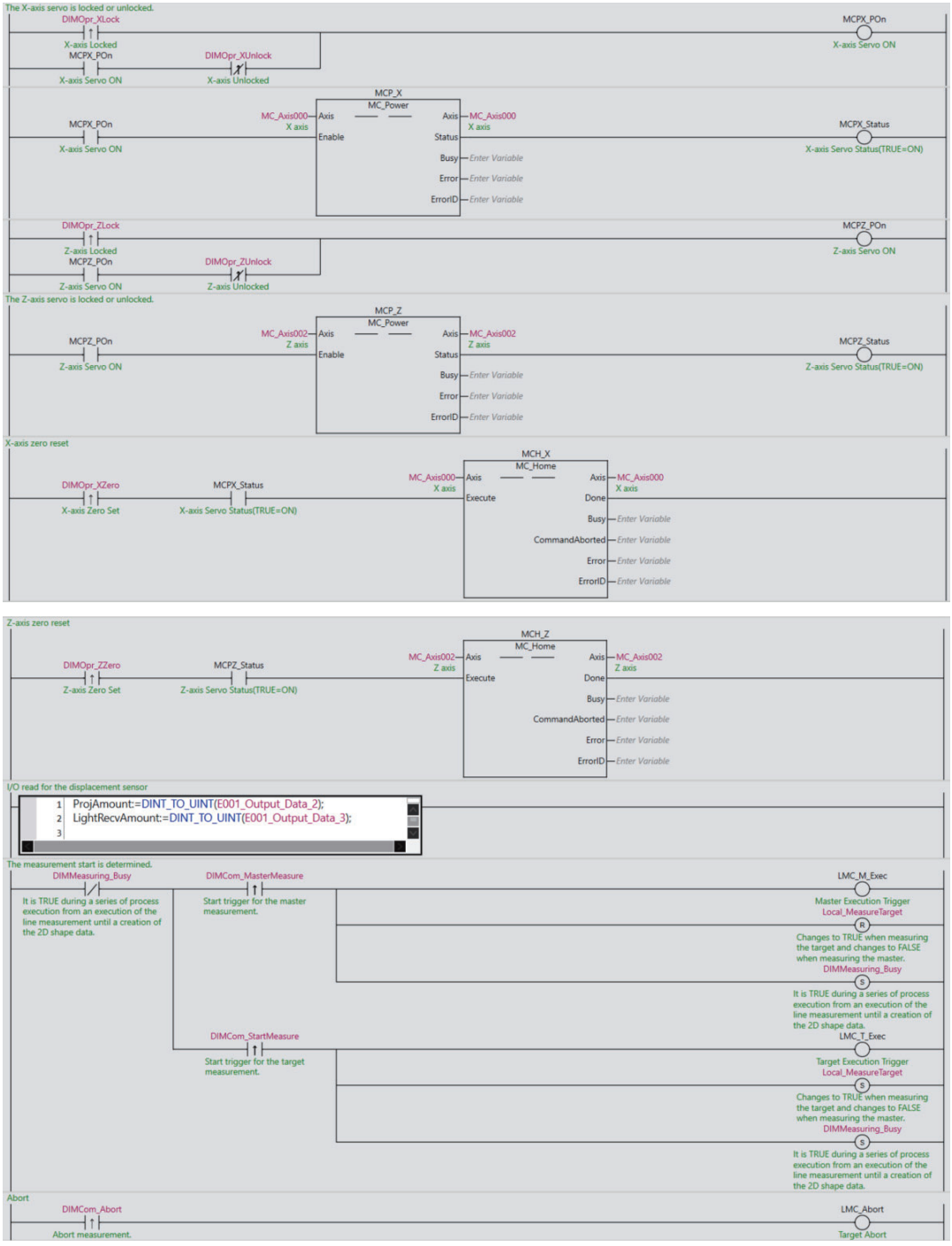
Function block	Assigned task
LineMeasure_Cartesian2	Primary periodic task
LineMeasure_CreateShape2D2	Periodic task
LineMeasure_CreateShape2D_Master2	Periodic task

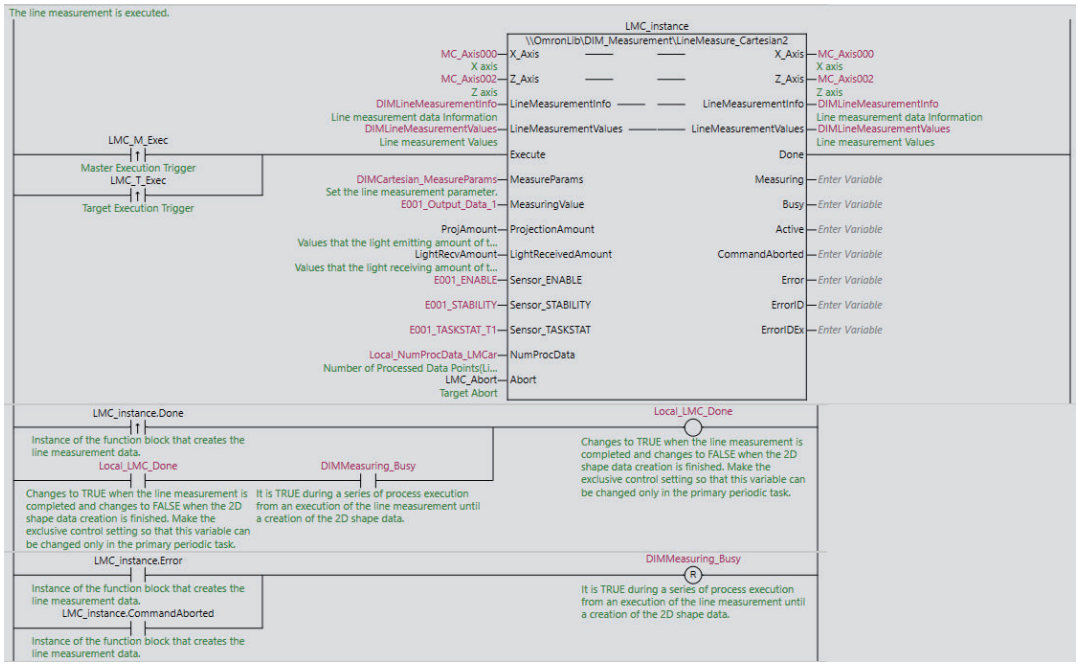
Main Variables

Name	Data type	Comment
DIMCartesian_MeasureParams	OmronLib\DIM_Measurement\sMeasureParams	Set the line measurement parameter.
DIMMeasuring_Busy	BOOL	It is TRUE during a series of process execution from an execution of the line measurement until a creation of the 2D shape data.
DIMCom_MasterMeasure	BOOL	Start trigger for the master measurement.
DIMCom_StartMeasure	BOOL	Start trigger for the target measurement.
Local_MeasureTarget	BOOL	Changes to TRUE when measuring the target and changes to FALSE when measuring the master.
Local_LMC_Done	BOOL	Changes to TRUE when the line measurement is completed and changes to FALSE when the 2D shape data creation is finished. Make the exclusive control setting so that this variable can be changed only in the primary periodic task.
DIMLineMeasurementInfo	OmronLib\DIM_Measurement\LineMeasurementInfo	Line measurement data information
DIMLineMeasurementValues	ARRAY [0..19999] OF OmronLib\DIM_Measurement\LineMeasurementValue	Line measurement data
DIMShape2D_Master	OmronLib\DIM_Measurement\sShapeMasterData2	Master 2D shape data information
DIMShapeZValue_Master	ARRAY[0..19999] OF LREAL	Master 2D shape data
DIMShape2D_Target	OmronLib\DIM_Measurement\sShape2DInfo	Target 2D shape data information
DIMShapeZValue_Target	ARRAY[0..19999] OF LREAL	Target 2D shape data
DIMCom_StateMaster	BOOL	When it is TRUE, the target measurement by using the master 2D shape data is performed.
LMC_instance	OmronLib\DIM_Measurement\LineMeasure_Cartesian2	Instance of the function block that creates the line measurement data.
ProjAmount	UINT	Values that the light emitting amount of ZW-7000 are converted into the UINT data.
LightRecvAmount	UINT	Values that the light receiving amount of ZW-7000 are converted into the UINT data.
LMCS_M_instance	OmronLib\DIM_Measurement\LineMeasure_CreateShape2D_Master2	An instance of the function block that creates the master 2D shape data.
LMCS_instance	OmronLib\DIM_Measurement\LineMeasure_CreateShape2D2	An instance of the function block that creates the 2D shape data.

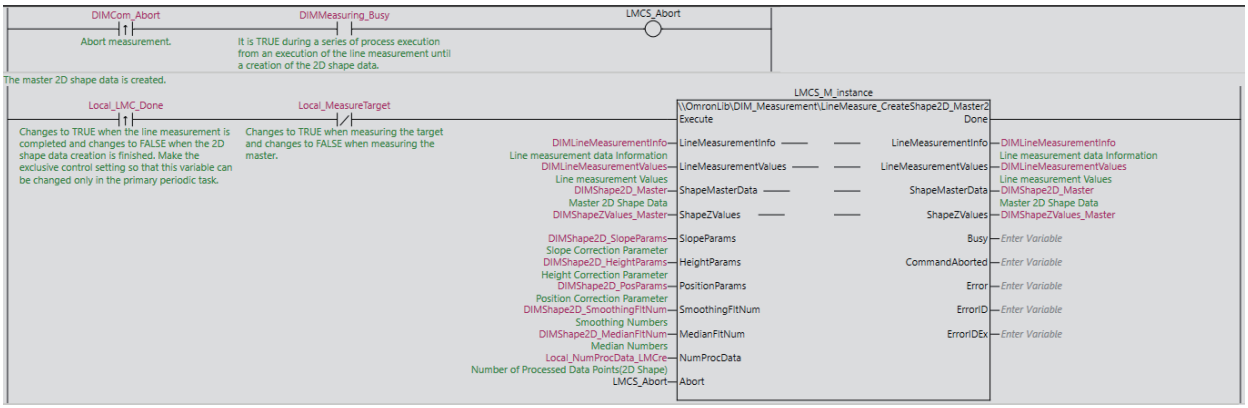
Ladder Diagram

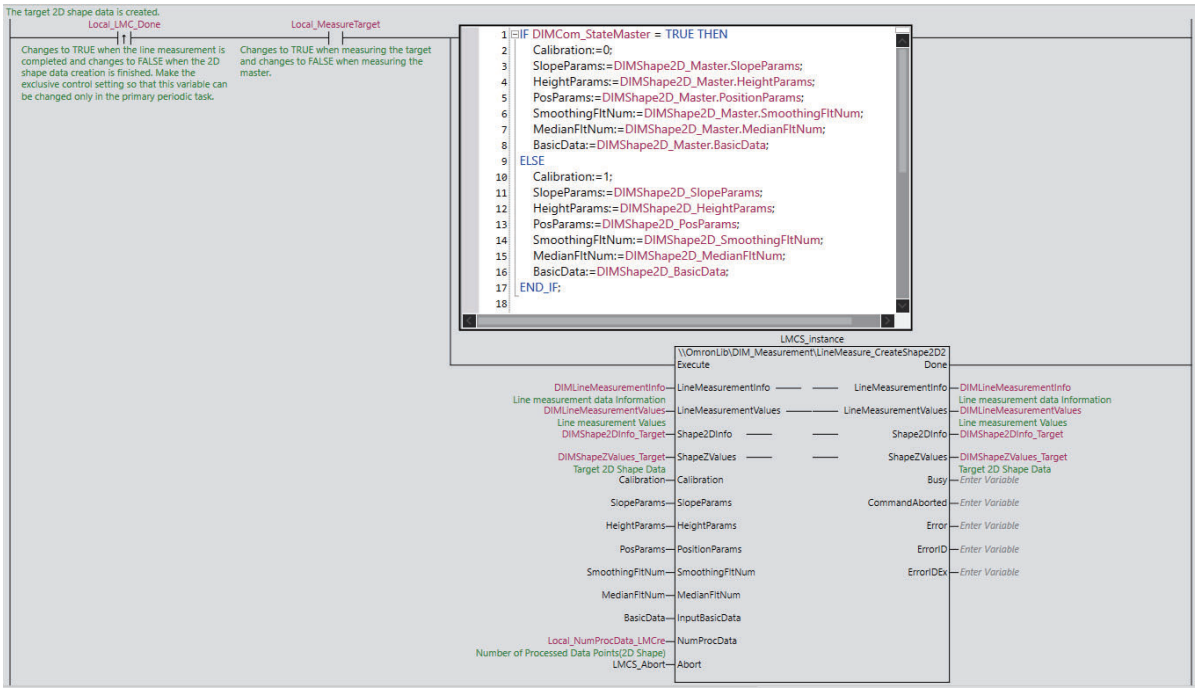
● User Program in the Primary Period Task

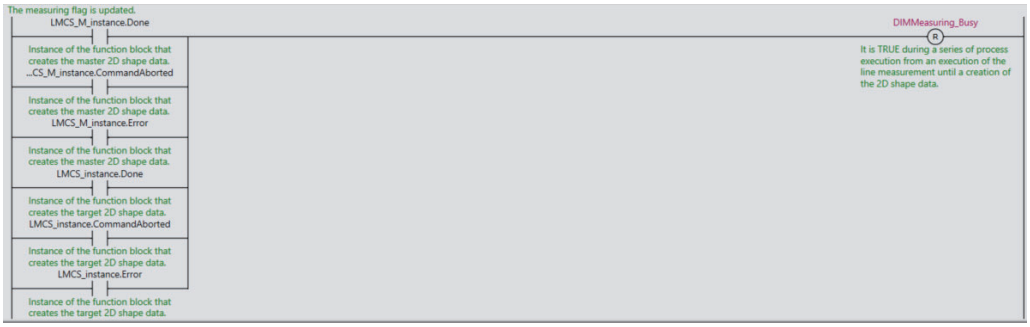



The line measurement is executed.
LMC_instance
\\OmronLib\DIM_Measurement\LineMeasure_Cartesian2
MC_Axis000
X axis
X_Axis
X_Axis
MC_Axis000
X axis
MC_Axis002
Z axis
Z_Axis
Z_Axis
MC_Axis002
Z axis
DIMLineMeasurementInfo
Line measurement data Information
LineMeasurementInfo
LineMeasurementInfo
DIMLineMeasurementInfo
Line measurement data Information
DIMLineMeasurementValues
Line measurement Values
LineMeasurementValues
LineMeasurementValues
DIMLineMeasurementValues
Line measurement Values
LMC_M_Exec
Master Execution Trigger
LMC_T_Exec
Target Execution Trigger
Execute
Done
DIMCartesian_MeasureParams
Set the line measurement parameter.
MeasureParams
Measuring
Enter Variable
E001_Output_Data_1
MeasuringValue
Busy
Enter Variable
ProjAmount
Values that the light emitting amount of t...
ProjectionAmount
Active
Enter Variable
LightRecvAmount
Values that the light receiving amount of t...
LightReceivedAmount
CommandAborted
Enter Variable
E001_ENABLE
Sensor_ENABLE
Error
Enter Variable
E001_STABILITY
Sensor_STABILITY
ErrorID
Enter Variable
E001_TASKSTAT_T1
Sensor_TASKSTAT
ErrorIDEx
Enter Variable
Local_NumProcData_LMCar
Number of Processed Data Points(Li...
NumProcData
LMC_Abort
Target Abort
Abort
LMC_instance.Done
Instance of the function block that creates the line measurement data.
Local_LMC_Done
Changes to TRUE when the line measurement is completed and changes to FALSE when the 2D shape data creation is finished. Make the exclusive control setting so that this variable can be changed only in the primary periodic task.
Local_LMC_Done
Changes to TRUE when the line measurement is completed and changes to FALSE when the 2D shape data creation is finished. Make the exclusive control setting so that this variable can be changed only in the primary periodic task.
DIMMeasuring_Busy
It is TRUE during a series of process execution from an execution of the line measurement until a creation of the 2D shape data.
LMC_instance.Error
Instance of the function block that creates the line measurement data.
DIMMeasuring_Busy
R
It is TRUE during a series of process execution from an execution of the line measurement until a creation of the 2D shape data.
LMC_instance.CommandAborted
Instance of the function block that creates the line measurement data.

User Program in the Periodic Task







ST

● User Program in the Primary Period Task

```
R_TRIG_LMC_XLock(DIMOpr_XLock,XLock);
R_TRIG_LMC_XUnlock(DIMOpr_XUnLock,XUnLock);
R_TRIG_LMC_ZLock(DIMOpr_ZLock,ZLock);
R_TRIG_LMC_ZUnlock(DIMOpr_ZUnLock,ZUnLock);
R_TRIG_LMC_XZero(DIMOpr_XZero,XZero);
R_TRIG_LMC_ZZero(DIMOpr_ZZero,ZZero);
R_TRIG_LMC_MasterMeasure(DIMCom_MasterMeasure,MasterMeasure);
R_TRIG_LMC_StartMeasure(DIMCom_StartMeasure,StartMeasure);
R_TRIG_DIMCom_Abort(DIMCom_Abort,LMC_Abort);

//The X-axis servo is locked or unlocked.
IF XLock OR ( MCPX_POn AND XUnLock ) THEN
        MCPX_POn:=TRUE;
END_IF;

MCP_X
(
   Axis:=MC_Axis000,
   Enable:=MCPX_POn,
   Status=>MCPX_Status
);


//The Z-axis servo is locked or unlocked.
IF ZLock OR ( MCPZ_POn AND ZUnLock ) THEN
   MCPZ_POn:=TRUE;
END_IF;

MCP_Z
(
   Axis:=MC_Axis002,
   Enable:=MCPZ_POn,
   Status=>MCPZ_Status
);

//The X-axis zero reset
MCH_X
(
   Axis:=MC_Axis000,
   Execute:=(XZero AND MCPX_Status)
);
//The Z-axis zero reset
MCH_Z
(
   Axis:=MC_Axis002,
   Execute:=(ZZero AND MCPZ_Status)
);

// I/O read for the displacement sensor
ProjAmount:=DINT_TO_UINT(E001_Output_Data_2);
LightRecvAmount:=DINT_TO_UINT(E001_Output_Data_3);

//The measurement start is determined.
IF NOT(DIMMeasuring_Busy) AND MasterMeasure THEN
        LMC_M_Exec:=TRUE;
        Local_MeasureTarget:=FALSE;
        DIMMeasuring_Busy:=TRUE;
ELSIF NOT(DIMMeasuring_Busy) AND StartMeasure THEN
        LMC_T_Exec:=TRUE;
```

```
        Local_MeasureTarget:=TRUE;
        DIMMeasuring_Busy:=TRUE;
ELSE
        LMC_M_Exec:=FALSE;
        LMC_T_Exec:=FALSE;
END_IF;

//The line measurement is executed.
LMC_instance
(
X_Axis:=MC_Axis000,
Z_Axis:=MC_Axis002,
LineMeasurementInfo:=DIMLineMeasurementInfo,
LineMeasurementValues:=DIMLineMeasurementValues,
Execute:=(LMC_M_Exec OR LMC_T_Exec),
MeasureParams:=DIMCartesian_MeasureParams,
MeasuringValue:=E001_Output_Data_1,
ProjectionAmount:=ProjAmount,
LightReceivedAmount:=LightRecvAmount,
Sensor_ENABLE:=E001_ENABLE,
Sensor_STABILITY:=E001_STABILITY,
Sensor_TASKSTAT:=E001_TASKSTAT_T1,
NumProcData:=Local_NumProcData_LMCar,
Abort:=LMC_Abort
);

IF LMC_instance.Done
   OR (Local_LMC_Done AND DIMMeasuring_Busy) THEN
   Local_LMC_Done:=TRUE;
ELSE
   Local_LMC_Done:=FALSE;
END_IF;

IF LMC_instance.Error OR LMC_instance.CommandAborted THEN
   DIMMeasuring_Busy:=FALSE;
END_IF;
```

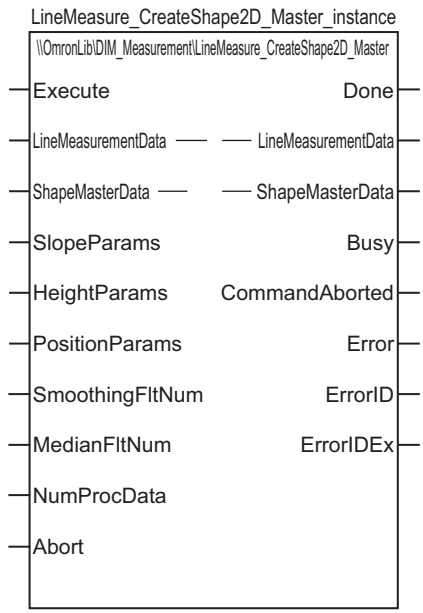
● User Program in the Periodic Task

```
R_TRIG_LMC_Done(Local_LMC_Done,LMC_Done);
R_TRIG_LMC_Done(DIMCom_Abort,LMCS_Abort);

IF LMCS_Abort AND DIMMeasuring_Busy THEN
   LMCS_Abort:=TRUE;
END_IF;

IF LMC_Done AND NOT(Local_MeasureTarget) THEN
   LMCS_M_Exec:=TRUE;
ELSE
   LMCS_M_Exec:=FALSE;
END_IF;

//The master 2D shape data is created.
LMCS_M_instance
(
   Execute:=LMCS_M_Exec,
   LineMeasurementInfo:=DIMLineMeasurementInfo,
   LineMeasurementValues:=DIMLineMeasurementValues,
   ShapeMasterData:=DIMShape2D_Master,
   ShapeZValues:=DIMShapeZValues_Master,
   SlopeParams:=DIMShape2D_SlopeParams,
   HeightParams:=DIMShape2D_HeightParams,
   PositionParams:=DIMShape2D_PosParams,
   SmoothingFltNum:=DIMShape2D_SmoothingFltNum,
   MedianFltNum:=DIMShape2D_MedianFltNum,
```

```
   NumProcData:=Local_NumProcData_LMCre,
   Abort:=LMCS_Abort
);

//The target 2D shape data is created.

IF LMC_Done AND Local_MeasureTarget THEN
   LMCS_Exec:=TRUE;
   IF DIMCom_StateMaster = TRUE THEN
         Calibration:=0;
         SlopeParams:=DIMShape2D_Master.SlopeParams;
         HeightParams:=DIMShape2D_Master.HeightParams;
         PosParams:=DIMShape2D_Master.PositionParams;
         SmoothingFltNum:=DIMShape2D_Master.SmoothingFltNum;
         MedianFltNum:=DIMShape2D_Master.MedianFltNum;
         BasicData:=DIMShape2D_Master.BasicData;
   ELSE
         Calibration:=1;
         SlopeParams:=DIMShape2D_SlopeParams;
         HeightParams:=DIMShape2D_HeightParams;
         PosParams:=DIMShape2D_PosParams;
         SmoothingFltNum:=DIMShape2D_SmoothingFltNum;
         MedianFltNum:=DIMShape2D_MedianFltNum;
         BasicData:=DIMShape2D_BasicData;
   END_IF;
ELSE
   LMCS_Exec:=FALSE;
END_IF;

LMCS_instance
(
   Execute:=LMCS_Exec,
   LineMeasurementInfo:=DIMLineMeasurementInfo,
   LineMeasurementValues:=DIMLineMeasurementValues,
   Shape2DInfo:=DIMShape2DInfo_Target,
   ShapeZValues:=DIMShapeZValues_Target,
   SlopeParams:=SlopeParams,
   HeightParams:=HeightParams,
   PositionParams:=PosParams,
   SmoothingFltNum:=SmoothingFltNum,
   MedianFltNum:=MedianFltNum,
   InputBasicData:=BasicData,
   NumProcData:=Local_NumProcData_LMCre,
   Abort:=LMCS_Abort
);

// The measuring flag is updated.

IF LMCS_M_instance.Done OR LMCS_M_instance.CommandAborted OR LMCS_M_instance.Error
   OR LMCS_instance.Done OR LMCS_instance.CommandAborted OR LMCS_instance.Error THEN
   DIMMeasuring_Busy:=FALSE;
END_IF;
```

LineMeasure_CreateShape2D_Master

The LineMeasure_CreateShape2D_Master function block creates the line measurement data and master 2D shape data from input parameters.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
LineMeasure_ CreateShape2D_- Master	Master 2D Shape Data Creation	FB		LineMeasure_CreateShape2D_instance(Execute, LineMeasurementData, ShapeMasterData, SlopeParams, HeightParams, PositionParams, SmoothingFltNum, MedianFltNum, NumProcData, Abort, Done, Busy, CommandAborted, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V1_X.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00148
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Refer to *Compatible Models* on page 109 in the LineMeasure_Cartesian.

Variable

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
SlopeParams*1	Slope Correction Parameter	Set the slope correction parameter.	sSlopeParams	—	—
HeightParams*1	Height Correction Parameter	Set the height correction parameter.	sHeightParams	—	—
Position Params*1	Position Correction Parameter	Set the position correction parameter.	sPositionParams	—	—
SmoothingFltNum*1	Number of Smoothing	0: No smoothing is performed 1: Travel averaging is performed twice 2: Travel averaging is performed 4 times 3: Travel averaging is performed 8 times 4: Travel averaging is performed 16 times 5: Travel averaging is performed 32 times 6: Travel averaging is performed 64 times 7: Travel averaging is performed 128 times 8: Travel averaging is performed 256 times	USINT	0 to 8	0
MedianFltNum*1	Median Range	0: No median processing is performed 1: Median Range is 3 2: Median Range is 5 3: Median Range is 7 4: Median Range is 9 5: Median Range is 15 6: Median Range is 31	USINT	0 to 6	0
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a task period in the segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
Busy	Executing	Changes to TRUE when the function block is acknowledged.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 192.

In-Out Variables

Variable	Name	Description	Data type	Valid range
LineMeasurement Data	Line measurement data	Set the line measurement data output by LineMeasure_Cartesian.	sLineMeasurement-Data	—
ShapeMasterData	Master 2D Shape Data	Stores the master 2D shape data.	sShapeMasterData	—

Members of Structure sSlopeParams

Member	Name	Data type	Valid range	Description
Apply	Apply Slope Correction	BOOL	TRUE or FALSE	Slope correction is performed when it is TRUE. Slope correction is not performed when it is FALSE.
Angle	Correction Angle	LREAL	-45° to 45°	Set the angle used for correction. The unit is [°]. This variable is valid only when slope correction is performed without using the reference point data.
Range1High	Straight Line Calculation Range 1 Upper Limit	LREAL	Negative number, positive number, or 0	Specify the upper limit of the X-axis measurement range 1 that is used for straight line calculation. The unit is [mm].
Range1Low	Straight Line Calculation Range 1 Lower Limit	LREAL	Negative number, positive number, or 0	Specify the lower limit of the X-axis measurement range 1 that is used for straight line calculation. The unit is [mm].
Range2High	Straight Line Calculation Range 2 Upper Limit	LREAL	Negative number, positive number, or 0	Specify the upper limit of the X-axis measurement range 2 that is used for straight line calculation. The unit is [mm].
Range2Low	Straight Line Calculation Range 2 Lower Limit	LREAL	Negative number, positive number, or 0	Specify the lower limit of the X-axis measurement range 2 that is used for straight line calculation. The unit is [mm].

Members of Structure sHeightParams

Member	Name	Data type	Valid range	Description
RangeHigh	Height Calculation Range Upper Limit	LREAL	Negative number, positive number, or 0	Specify the upper limit of the X-axis measurement range that is used for standard height calculation. The unit is [mm].
RangeLow	Height Calculation Range Lower Limit	LREAL	Negative number, positive number, or 0	Specify the lower limit of the X-axis measurement range that is used for standard height calculation. The unit is [mm].
HeightType	Height Type	USINT	0, 1, or 2	Specify the method to determine the height within the correction range. 0: Average 1: Peak top 2: Peak bottom

Members of Structure sPositionParams

Member	Name	Data type	Valid range	Description
EdgeHeight	Edge Height	LREAL	Negative number, positive number, or 0	Specify the height used to determine whether to detect the edge. Set the height to the 2D shape data that the correction is performed. The unit is [mm].
EdgeNumber	Number of Detected Edges	USINT	Positive number	Specify what number detected edge is used for correction.

Members of Structure sBasicData

For these member variables, set the values output by LineMeasure_CreateShape2D_Master.

Member	Name	Data type	Valid range	Description
Angle	Angle	LREAL	-45° to 45°	Set the reference angle calculated from the master 2D shape data. The unit is [θ].
EdgeIndex	Edge Detection Index	UINT	0 to the number of points to measure	Set the reference edge detection index that is stored in the master 2D shape data.

Members of Structure sShapeMasterData

Member	Name	Data type	Valid range	Description
Shape2D	2D Shape Data	sShape2D	—	Refer to *2D Shape Data* on page 123 for the members.
SlopeParams	Slope Correction Parameter	sSlopeParams	—	Refer to *Members of Structure sSlopeParams* on page 179 for the members.
HeightParams	Height Correction Parameter	sHeightParams	—	Refer to *Members of Structure sHeightParams* on page 179 for the members.
PositionParams	Position Correction Parameter	sPositionParams	—	Refer to *Members of Structure sBasicData* on page 180 for the members.
SmoothingFltNum	Number of Smoothing	USINT	0 to 8	Refer to *Input Variables* on page 177 for details.
MedianFltNum	Median Range	USINT	0 to 6	Refer to *Input Variables* on page 177 for details.
BasicData	Reference Point Data	sBasicData	—	Refer to *Members of Structure sBasicData* on page 180 for the members.

Members of Structure sShape2D

Member	Name	Data type	Valid range	Description
DataCount	Number of Data Points	UINT	0 to 20,000	The number of data points are stored.
Resolution	Resolution	UINT	Positive number	The resolution in X direction*1 The unit is μm.
ZValue	Z Measurement Data	ARRAY[0..19999] OF LREAL	Depends on data type.	The X-direction values of 2D shape data are stored. The unit is mm.

*1. Use the following expression to find the position of *sShape2D.ZValue[m]*.
Position[m] = UINT_TO_LREAL((m-1) * *sShape2D.Resolution*) / LREAL#1000

Function

The LineMeasure_CreateShape2D_Master function block provides the following functions.

(a) Conversion from the line measurement data to the 2D shape data

This function block converts the line measurement data to the 2D shape data.

Refer to *Line Measurement Data* on page 123 and *2D Shape Data* on page 123 for the difference in data structure between the line measurement data and 2D shape data.

(b) Measurement Correction

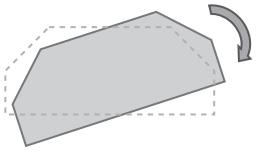
This function block corrects the slope and position that are detected when master measurement is performed.

a) Slope Correction

The function block corrects angular deviation that is caused by the tilt of workpiece during measurement.

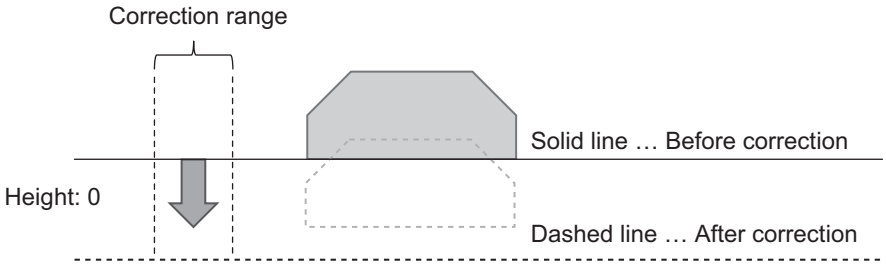
The slope correction is not performed if Apply of Slope Correction Parameter is set to FALSE.

For example, if Correction Angle of Slope Correction Parameter is set to 0°, the output 2D shape data contains the slope of 0° even when the workpiece is tilted.



b) Height Correction

The function block corrects the height of the measurement surface within the correction range so that it becomes 0.



(c) Filtering

Depending on the shape of the target or condition of the surface, the line measurement data may be unstable. In such a case, you can use the filters such as Smoothing and Median (X direction) to reduce noises.

a) Smoothing

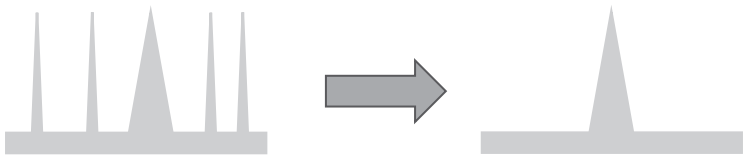
At each position in X direction, travel averaging is performed according to the specified number of smoothing. You can create the data of a smooth shape.

b) Median (X direction)

The function block sorts out the values of *ZValue* within the range specified by Median Range in which the X position is set as the center.

The center value is replaced with the *ZValue* at the X position.

By applying the Median filter, you can remove spikes that are contained in the shape data.



(d) Reference Point Data Calculation

This function block outputs the edge detection index and reference angle. These data are called the reference point data. The reference point data calculated by this function block is used for the setting when executing LineMeasure_CreateShape2D. This data is required to correct the target 2D shape data so that deviations caused by the difference in the measurement environment between master and target can be reduced. The details on the edge detection index and reference angle are described in *Reference Point Data Output* on page 187.

The following describes an overview of the entire processing.

a) The function block checks the input parameters when *Execute* changes to TRUE.

Only the parameters that are required for correction are checked.

If there is no error in the input parameters, the line measurement data (*LineMeasurementData*) that is input is corrected and filtered according to the correction and filtering settings. After the processing is completed, the line measurement data is converted to the master 2D shape data (*ShapeMasterData.Shape2D*) and stored.

The reference point data calculated from the shape data, and the correction parameter and filtering setting that are input are recorded in the master 2D shape data.

If an error is detected in the input parameters, the function block execution is aborted.

For details on the error codes, refer to *Troubleshooting* on page 192.

b) If there is no recorded data (*LineMeasurementData.DataCount* = 0), the function block execution ends without performing any operation.

The master 2D shape data is not created.

c) The unit for the measurement data is always “mm”.

d) According to the parameter settings, correction, filtering, data structure conversion and reference data are performed, and the master 2D shape data are created.

The details on each function are described in the following sections.

e) This function block corrects and converts a maximum of 20,000 measurement data points. So, it requires a considerable time to complete processing.

If this function block exclusively uses a long processing time, other processing may not be executed or a Task Period Exceeded error may occur. To avoid this, you need to set the number of processed data points *(NumProcData)* per period.

The function block performs correction and conversion for a specified number of data points within a period. In the next period, the processing is resumed where it was interrupted.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, consider the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

f) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

Note that the execution is completed if all correction processing is finished in the period at which *Abort* changes to TRUE.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

g) If an error occurs, *Error* changes to TRUE and the function block execution is aborted. In addition, the error codes are output to *ErrorID* and *ErrorIDEx*.

For details on the error codes, refer to *Troubleshooting* on page 192.

h) If the function block is aborted or ended in an error, the master 2D shape data will be undefined.

Conversion to 2D Shape Data

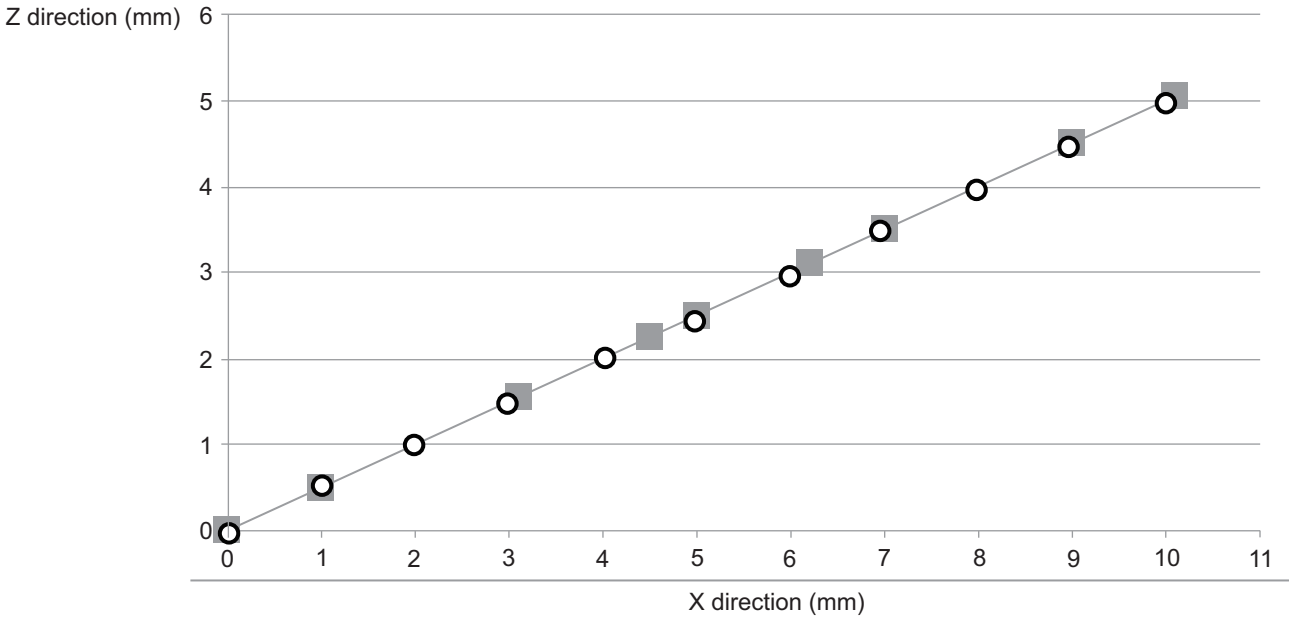
The function block converts the line measurement data to the 2D shape data.

For details on the line measurement data structure, refer to *Line Measurement Data* on page 123.

For details on the 2D shape data structure, refer to *2D Shape Data* on page 123.

The line measurement data with irregular intervals in the X direction is converted to the 2D shape data with regular intervals in the X direction.

The following is an example of conversion. The line measurement data with the resolution of 1,000 μm is expressed as "■" dots in the following figure. The data are not spaced regularly. The converted 2D shape data is expressed as "○" dots. The measurement points are set at intervals of the resolution so that the data is spaced regularly in X direction. If a Z-direction value does not exist in the line measurement data, it is calculated by interpolation using the neighboring measurement values.



The resolution used for measurement cannot be changed when the data is converted.

To change the resolution, you must redo the measurement.

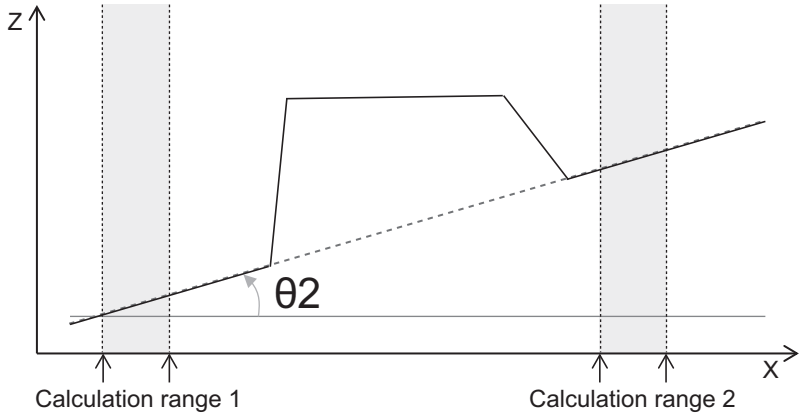
Slope Correction

When slope correction is selected (*sSlopeParams.Apply* = TRUE), the correction procedure is performed as follows.

(a) Input the correction angle θ1 (*sSlopeParams.BasicAngle*).

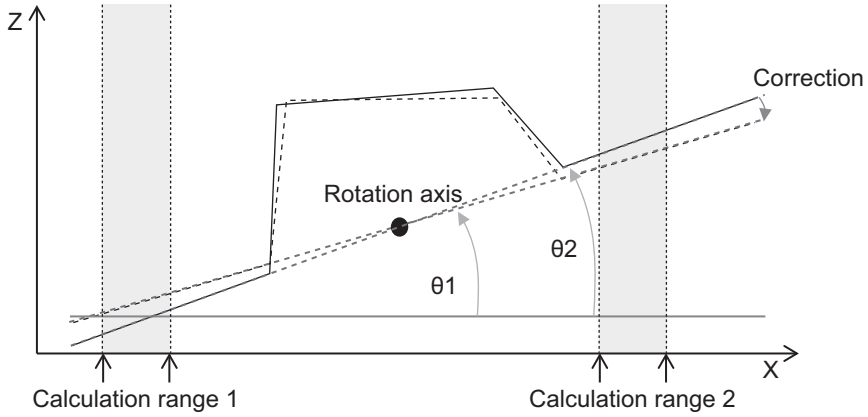
(b) When the slope of the measured object (θ2) is calculated, the straight line is created from all Z values included in Straight Line Calculation Range 1 (*sSlopeParams.Range1Low* to *sSlopeParams.Range1High*) and Straight Line Calculation Range 2 (*sSlopeParams.Range2Low* to *sSlopeParams.Range2High*).

If one or more array elements are not found in each range, a correction-impossible error occurs.



(c) If the slope of the measured object (θ2) is different from the correction angle θ1, the entire measurement data is rotated so that these angles can be the same.

The rotation axis is placed at the midpoint between *Range1High* and *Range2Low*.



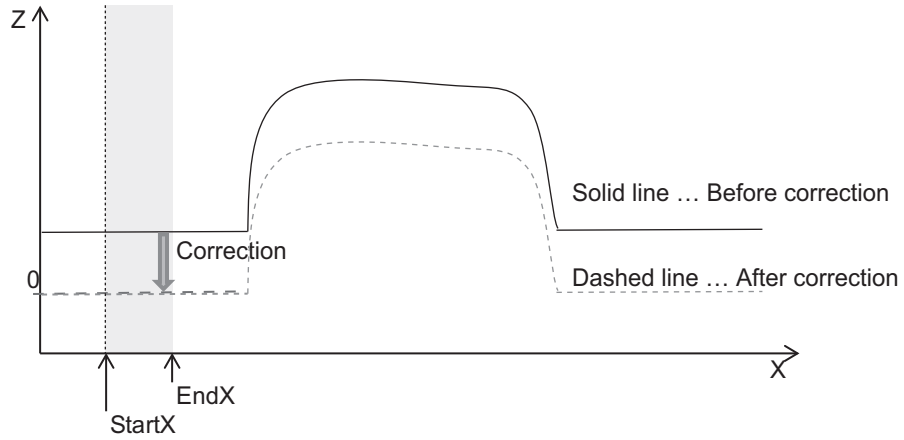
Height Correction

The height of the entire measurement data is corrected so that the heights within the height correction range can be 0.

(a) The function block determines the height of the measured object based on all Z values included in the height correction range (from *sHeightParams.RangeLow* to *sHeightParams.RangeHigh*) and the method specified in Height Type (*sHeightParams.HeightType*).

If one or more array elements are not found in each range, a compensation-impossible error occurs.

(b) If the calculated height is not 0, the entire measurement data is moved up/down through correction so that the height can be 0.



Position Correction

Position correction is not performed for the master data.

Smoothing Filter

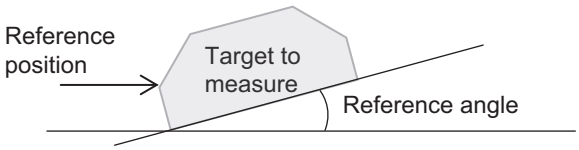
When the Number of Smoothing (*SmoothingFltNum*) is set to 1 or higher, the specified number of travel averaging is performed for each position.

Median Filter (X direction)

When the Median Range (*MedianFltNum*) is set to 1 or higher, the median processing is performed for each position within the specified range.

Reference Point Data Output

For the master 2D shape data, X-axis Measurement Start Position of the line measurement data is set as the origin of X coordinate. Then, the position at which the edge is detected is defined as the reference position. Also, the angle of the master 2D shape data relative to the horizontal surface is defined as the reference angle. The function block outputs Edge Detection Index and Angle, from which the reference position of the master 2D shape data can be calculated. These data are called the reference point data. The reference point data is used when LineMeasure_CreateShape2D is executed. This data is required to correct deviation caused by the difference in the measurement environment between the master and target.



Reference Angle

The slope of the measurement data calculated in *Slope Correction* on page 185 is used as the setting value.

When slope correction is applied, the angle after correction is used.

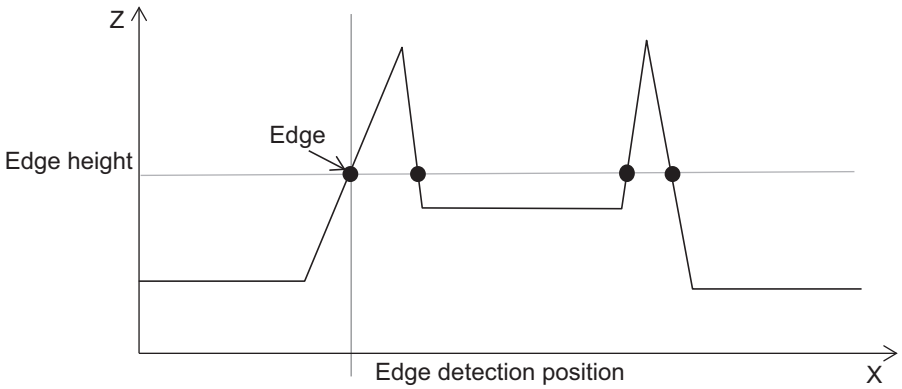
Edge Detection Index

Edge detection position = Resolution × Edge Detection Index

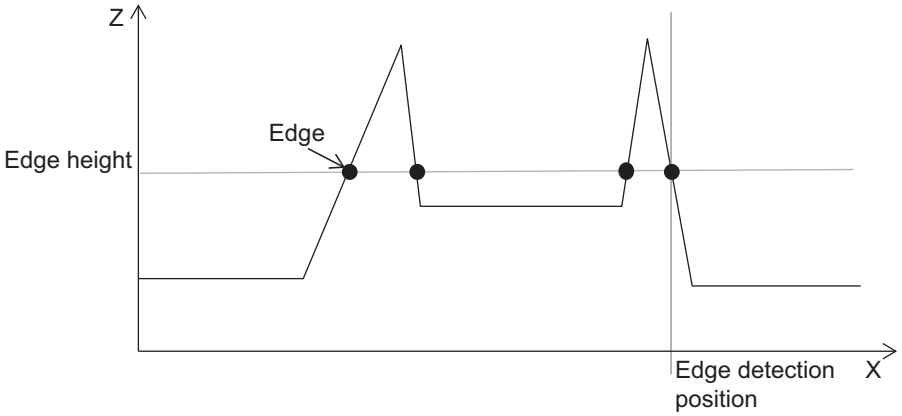
The edge detection index is calculated from the edge position detected through the following procedure.

a) The heights in the 2D shape data are checked in ascending order of X values.
When a height exceeds Edge Height (EdgeHeight), an edge is judged to be detected at that point.
A detected edge is determined to be the edge detection position according to the setting of Number of Edge Detection (*EdgeNumber*).

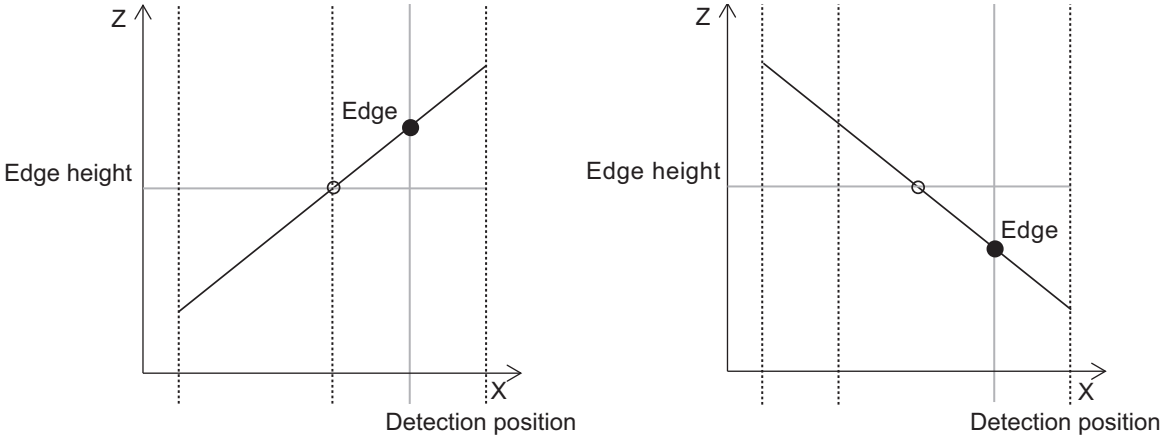
When Number of Edge Detection = 1



When Number of Edge Detection = 4



b) When a measured height is the same as Edge Height, it is not recognized as an edge.
Only a point after passing Edge Height is determined to be an edge detection position.



Re-execution of Function Blocks

If you change *Execute* to TRUE when execution is in progress (*Busy* = TRUE), the calculation in progress is aborted and the correction calculation is restarted with new parameters.

Multi-execution of Function Blocks

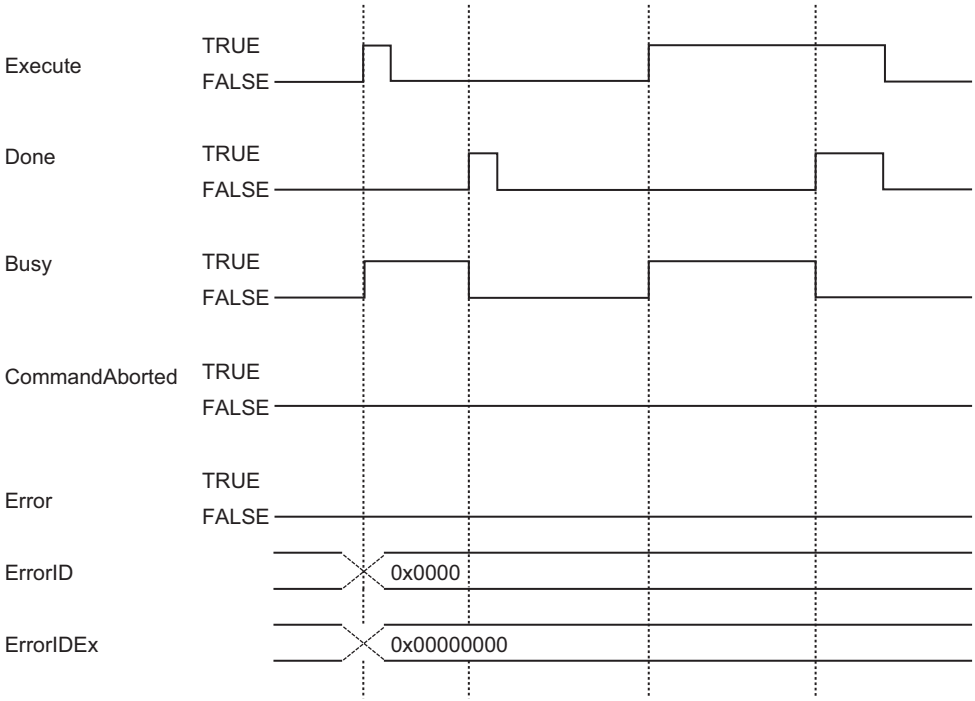
It is allowed to generate multiple instances from this function block and to execute multiple correction calculations simultaneously.

However, you must set different in-output variables for each instance when you assign them to the master 2D shape data.

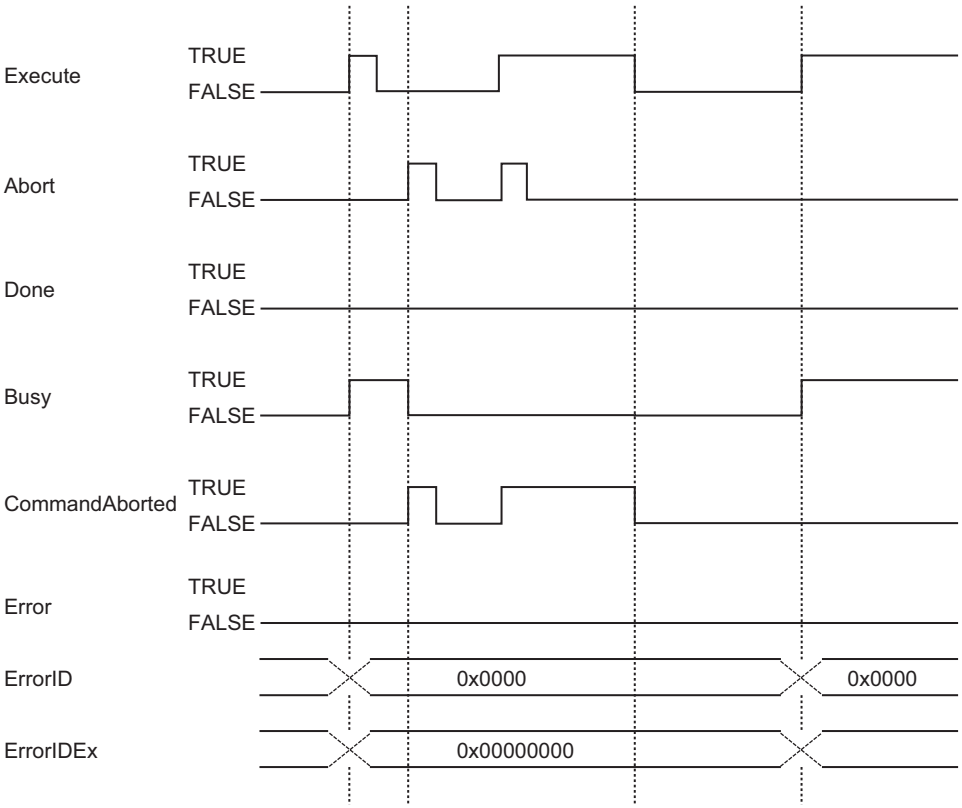
Timing Charts

The timing charts are shown below.

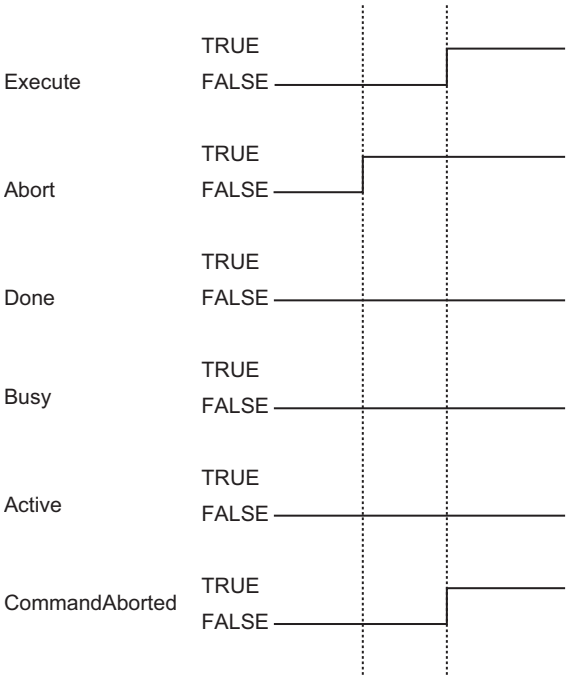
● Timing Chart in a Normal State



● Timing Chart When the Function Block Is Aborted

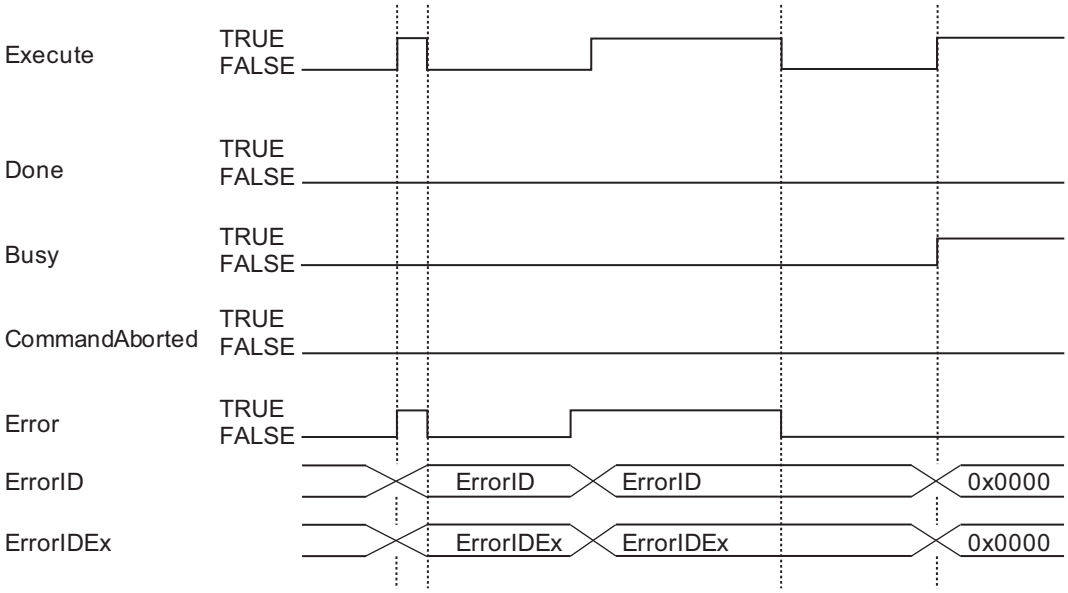


If *Execute* changes to TRUE while *Abort* is TRUE, *CommandAborted* (Interruption Completion) changes to TRUE without performing the processing.



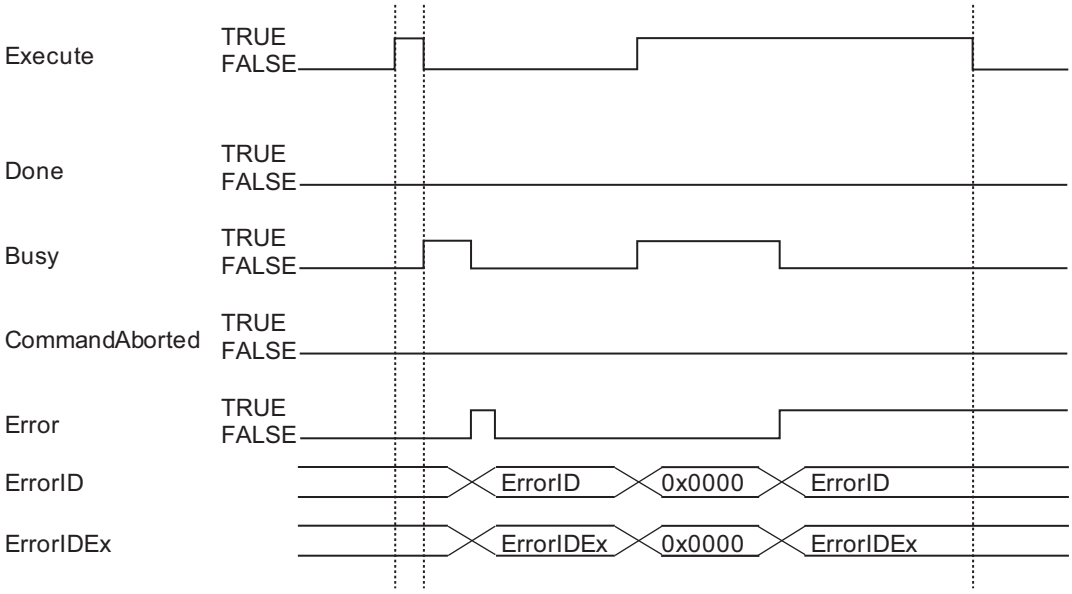
Timing Chart When an Error Occurs (Startup Error)

If a setting value error is detected for the input variables when the function block is executed, a startup error occurs. *Error* changes to TRUE and the values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code). When *Execute* changes to FALSE, *Error* changes to FALSE accordingly. However, *ErrorID* and *ErrorIDEx* retain the values until the next function block is executed.



Timing Chart When an Error Occurs (Error during Function Block Execution)

If an error is detected in the 2D shape data creation or correction during the function block execution, *Busy* changes to FALSE, *Error* changes to TRUE, and the values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code). When *Execute* changes to FALSE, *Error* changes to FALSE accordingly. However, *ErrorID* and *ErrorIDEx* retain the values until the next function block is executed.



Precautions for Correct Use

- Execute this function block only after the line measurement function block (LineMeasure_Cartesian) is completed normally. If the line measurement function block is aborted or ended in an error, the line measurement data is undefined. If you execute this function block with undefined line measurement data, the master 2D shape data cannot be created correctly.
- Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3CD7	0x0000 0001	Number of Recorded Data (*LineMeasurementData.DataCount*) was set to a value that exceeds the maximum number of stored measurement data.	Set a valid number of stored data.
	0x0000 0002	Resolution (*LineMeasurement-Data.Resolution*) was set to 0.	Correct the input data.
	0x0000 0003	The X values of the measurement data that was input is not stored in ascending order.	Correct the input data.
	0x0000 0004	A reference point (slope) exceeding the valid range was detected from the measurement data that was input.	Correct the input data and slope correction setting.
	0x0000 0005	Slope correction was executed based on the reference angle detected from the measurement data that was input and Reference Angle (*SlopeParams.Angle*) in the slope correction setting. As the result, the valid range was exceeded.	Correct the input data and slope correction setting.
	0x0000 0006	The reference point (edge detection position) was not detected from the measurement data that was input.	Correct the input data and position correction setting.
	0x0000 0007	The value specified for Number of Smoothing (*SmoothingFltNum*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0008	The value specified for Median Range (*MedianFltNum*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0010	Slope Correction Parameter The value specified for Reference Angle (*Angle*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0011	Slope Correction Parameter The value specified for Straight Line Calculation Range 1 Upper Limit (*Range1High*) is out of the valid range.	Specify a value which is within the valid range.

Error code	Expansion error code	Description	Corrective action
0x3CD7	0x0000 0012	Slope Correction Parameter The value specified for Straight Line Calculation Range 1 Lower Limit (*Range1Low*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0013	Slope Correction Parameter The value specified for Straight Line Calculation Range 2 Upper Limit (*Range1High*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0014	Slope Correction Parameter The value specified for Straight Line Calculation Range 2 Lower Limit (*Range2Low*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0015	Slope Correction Parameter The values specified for Straight Line Calculation Range 1 (*Range1Low* and *Range1High*) has met the following condition: lower limit ≥ upper limit.	Specify values that can meet the following condition: lower limit < upper limit.
	0x0000 0016	Slope Correction Parameter The values specified for Straight Line Calculation Range 2 (*Range2Low* and *Range2High*) has met the following condition: lower limit ≥ upper limit.	Specify values that can meet the following condition: lower limit < upper limit.
	0x0000 0017	Slope Correction Parameter The values specified for Straight Line Calculation Range 1 and 2 have met the following condition: *Range1High* ≥ *Range2Low*.	Specify values that can meet the following condition: *Range1High* < *Range2Low*.
	0x0000 0018	Slope Correction Parameter There is no measurement data that corresponds to the Straight Line Calculation Range 1.	Set Straight Line Calculation Range 1 to values that include the measurement data.
	0x0000 0019	Slope Correction Parameter There is no measurement data that corresponds to the Straight Line Calculation Range 2.	Set Straight Line Calculation Range 2 to values that include the measurement data.
	0x0000 0020	Height Correction Parameter The value specified for Height Calculation Range Upper Limit (*RangeHigh*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0021	Height Correction Parameter The value specified for Height Calculation Range Lower Limit (*RangeLow*) is out of the valid range.	Specify a value which is within the valid range.

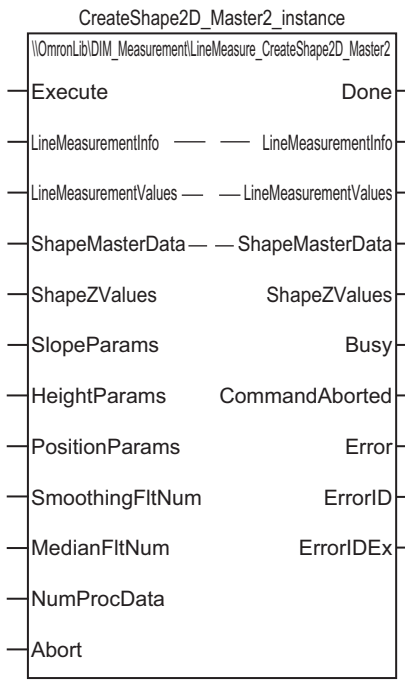
Error code	Expansion error code	Description	Corrective action
0x3CD7	0x0000 0022	Height Correction Parameter The values specified for Height Calculation Range (*RangeLow* and *RangeHigh*) has met the following condition: lower limit ≥ upper limit.	Specify values that can meet the following condition: lower limit < upper limit.
	0x0000 0023	Height Correction Parameter The value specified for Height Type (*HeightType*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0024	Height Correction Parameter There is no measurement data that corresponds to the Height Calculation Range.	Set Height Calculation Range to values that include the measurement data.
	0x0000 0030	Position Correction Parameter The value specified for Edge Height (*EdgeHeight*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0031	Position Correction Parameter The value specified for Number of Edge Detection (*EdgeNumber*) is out of the valid range.	Specify a value which is within the valid range.

Sample Programming

Refer to *Sample Programming* on page 143 for LineMeasure_Cartesian.

LineMeasure_CreateShape2D_Master2

The LineMeasure_CreateShape2D_Master2 function block creates master 2D shape data from the line measurement data and input parameters. 2D shape data is a variable-length data array.

Function block and function name	Name	FB/ FUN	Graphic expression	ST expression
LineMeasure_CreateShape2D_Master2	Master 2D Shape Data Creation 2	FB		CreateShape2D_Master2_instance(Execute, LineMeasurementInfo, LineMeasurementValues, ShapeMasterData, ShapeZValues, SlopeParams, HeightParams, PositionParams, SmoothingFltNum, MedianFltNum, NumProcData, Abort, Done, Busy, CommandAborted, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V2_0.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00228
Source code	Not published

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
SlopeParams*1	Slope Correction Parameter	Set the slope correction parameter.	sSlopeParams	—	—
HeightParams*1	Height Correction Parameter	Set the height correction parameter.	sHeightParams	—	—
Position Params*1	Position Correction Parameter	Set the position correction parameter.	sPositionParams	—	—
SmoothingFlt Num*1	Number of Smoothing	0: No smoothing is performed 1: Travel averaging is performed twice 2: Travel averaging is performed 4 times 3: Travel averaging is performed 8 times 4: Travel averaging is performed 16 times 5: Travel averaging is performed 32 times 6: Travel averaging is performed 64 times 7: Travel averaging is performed 128 times 8: Travel averaging is performed 256 times	USINT	0 to 8	0
MedianFltNum*1	Median Range	0: No median processing is performed 1: Median Range is 3 2: Median Range is 5 3: Median Range is 7 4: Median Range is 9 5: Median Range is 15 6: Median Range is 31	USINT	0 to 6	0
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a period during segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
Busy	Executing	TRUE when *Execute* is acknowledged.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 212 for details.

In-Out Variables

Variable	Name	Description	Data type	Valid range
LineMeasurement Info	Line Measurement Information	Set the line measurement information output by LineMeasure_Cartesian2.	sLineMeasurement Info	—
LineMeasurement Values[] *1	Line Measurement Data Array	Set the line measurement array output by LineMeasure_Cartesian2.	ARRAY[*] OF sLineMeasurement Value	—
ShapeMasterData	Master 2D Shape Data	Stores the master 2D shape data.	sShapeMasterData2	—
ShapeZValues[] *1	2D Shape Data Array	Stores the 2D shape data array.	ARRAY[*] OF LREAL	—

*1. The maximum number of array elements is 20,000. The first number of array element should be 0.

Members of Structure sSlopeParams

Member	Name	Data type	Valid range	Description
Apply	Apply Slope Correction	BOOL	TRUE or FALSE	Slope correction is performed when it is TRUE. Slope correction is not performed when it is FALSE.
Angle	Correction Angle	LREAL	-45° to 45°	Set the angle used for correction. The unit is [°]. This variable is valid only when slope correction is performed without using the reference point data.
Range1High	Straight Line Calculation Range 1 Upper Limit	LREAL	Negative number, positive number, or 0	Specify the upper limit of the X-axis measurement range 1 that is used for straight line calculation. The unit is [mm].
Range1Low	Straight Line Calculation Range 1 Lower Limit	LREAL	Negative number, positive number, or 0	Specify the lower limit of the X-axis measurement range 1 that is used for straight line calculation. The unit is [mm].
Range2High	Straight Line Calculation Range 2 Upper Limit	LREAL	Negative number, positive number, or 0	Specify the upper limit of the X-axis measurement range 2 that is used for straight line calculation. The unit is [mm].
Range2Low	Straight Line Calculation Range 2 Lower Limit	LREAL	Negative number, positive number, or 0	Specify the lower limit of the X-axis measurement range 2 that is used for straight line calculation. The unit is [mm].

Members of Structure sHeightParams

Member	Name	Data type	Valid range	Description
RangeHigh	Height Calculation Range Upper Limit	LREAL	Negative number, positive number, or 0	Specify the upper limit of the X-axis measurement range that is used for standard height calculation. The unit is [mm].
RangeLow	Height Calculation Range Lower Limit	LREAL	Negative number, positive number, or 0	Specify the lower limit of the X-axis measurement range that is used for standard height calculation. The unit is [mm].
HeightType	Height Type	USINT	0, 1, or 2	Specify the method to determine the height within the correction range. 0: Average 1: Peak top 2: Peak bottom

Members of Structure sPositionParams

Member	Name	Data type	Valid range	Description
EdgeHeight	Edge Height	LREAL	Negative number, positive number, or 0	Specify the height used to determine whether to detect the edge. Set the height to the 2D shape data that the correction is performed. The unit is [mm].
EdgeNumber	Number of Detected Edges	USINT	Positive number	Specify what number detected edge is used for correction.

Members of Structure sBasicData

Member	Name	Data type	Valid range	Description
Angle	Angle	LREAL	-45° to 45°	Set the reference angle calculated from the master 2D shape data. The unit is [°].
EdgeIndex	Edge Detection Index	UINT	Positive number	Set the reference edge detection index that is stored in the master 2D shape data array.

Members of Structure sLineMeasurementInfo and sShape2DInfo

Member	Name	Data type	Valid range	Description
DataCount	Number of Data Points	UINT	0 to 20,000	The number of data points are stored.
Resolution	Resolution	UINT	Positive number	The resolution in X direction.*1 The unit is μm.

*1. Use the following expression to find the position of *ShapeZValues[m]*.
Position[m] = UINT_TO_LREAL((m-1) * *Shape2DInfo.Resolution*) / LREAL#1000

Members of Structure sLineMeasurementValue

Member	Name	Data type	Valid range	Description
XValue	X Measurement Data	LREAL	Depends on data type.	The measurement positions are recorded. The unit is [mm].
ZValue	Z Measurement Data	LREAL	Depends on data type.	The heights measured at the measurement positions are recorded. The unit is [mm].
Projection Amount	Amount of Light Emitted Data	UINT	Depends on data type.	The amount of light emitted at the measurement positions are recorded.
LightReceivedAmount	Amount of Light Received Data	UINT	Depends on data type.	The amount of light received at the measurement positions are recorded.

Members of Structure sShapeMasterData2

Member	Name	Data type	Valid range	Description
Shape2DInfo	2D Shape Data Information	sShape2DInfo	—	Refer to *Members of Structure sLineMeasurementInfo and sShape2DInfo* on page 199 for the members.
SlopeParams	Slope Correction Parameter	sSlopeParams	—	Refer to *Members of Structure sSlopeParams* on page 198 for the members.
HeightParams	Height Correction Parameter	sHeightParams	—	Refer to *Members of Structure sHeightParams* on page 198 for the members.
PositionParams	Position Correction Parameter	sPositionParams	—	Refer to *Members of Structure sPositionParams* on page 199 for the members.
SmoothingFltNum	Number of Smoothing	USINT	0 to 8	Refer to *Input Variables* on page 196 for details.
MedianFltNum	Median Range	USINT	0 to 6	Refer to *Input Variables* on page 196 for details.
BasicData	Reference Point Data	sBasicData	—	Refer to *Members of Structure sBasicData* on page 199 for the members.

Function

The LineMeasure_CreateShape2D_Master2 function block provides the following functions.

(a) Conversion from the line measurement data to the 2D shape data

This function block converts the line measurement data to the 2D shape data.

Refer to *Line Measurement Data* on page 123 and *2D Shape Data* on page 123 of LineMeasure_-Cartesian for the difference in data structures between the line measurement data and 2D shape data.

(b) Measurement Correction

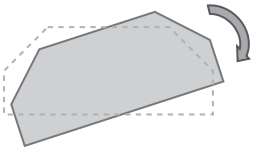
This function block corrects the slope and position that are detected when master measurement is performed.

a) Slope Correction

The function block corrects angular deviation that is caused by the tilt of workpiece during measurement.

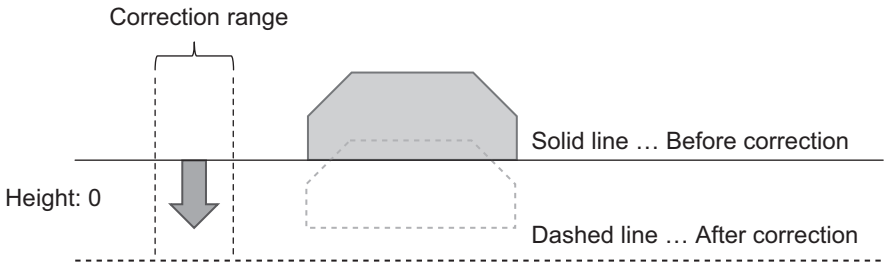
The slope correction is not performed if Apply of Slope Correction Parameter is set to FALSE.

For example, if Correction Angle of Slope Correction Parameter is set to 0°, the output 2D shape data contains the slope of 0° even when the workpiece is tilted.



b) Height Correction

The function block corrects the height of the measurement surface within the correction range so that it becomes 0.



(c) Filtering

Depending on the shape of the target or condition of the surface, the line measurement data may be unstable.In such a case, you can use the filters such as Smoothing and Median (X direction) to reduce noises.

a) Smoothing

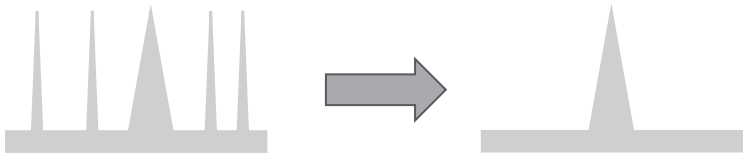
At each position in X direction, travel averaging is performed according to the specified number of smoothing. You can create the data of a smooth shape.

b) Median (X direction)

The function block sorts out *ZValue* of the line measurement data within a range specified by Median Range in which the X position is set as the center.

The center value is replaced with *ZValue* at the X position.

By applying the Median filter, you can remove spikes that are contained in the shape data.



(d) Reference Point Data Calculation

This function block outputs the edge detection index and reference angle. These data are called the reference point data. The reference point data calculated by this function block is used for the setting when executing LineMeasure_CreateShape2D2. This data is required to correct the target 2D shape data so that deviations caused by the difference in the measurement environment between master and target can be reduced. The details on the edge detection index and reference angle are described in *Reference Point Data Output* on page 206.

By using this function, deviation caused by differences in the measurement environment is corrected and you can compare the difference in shape between the master and target.

The following describes an overview of the entire processing.

a) The function block checks the input parameters when *Execute* changes to TRUE.

Only the parameters that are required for correction are checked.

If there is no error in the input parameters, the line measurement data array (*LineMeasurementValues[].ZValue*) that is input is corrected and filtered according to the correction and filtering settings. After the processing is completed, the line measurement data array is converted to the master 2D shape data array (*ShapeZValues[]*) and stored.

The reference point data calculated from the shape data, and the correction parameter and filtering setting that are input are recorded in the master 2D shape data.

If an error is detected in the input parameters, the function block execution is aborted.

For details on the error codes, refer to *Troubleshooting* on page 212.

b) If there is no recorded data (*LineMeasurementInfo.DataCount* = 0), the function block execution ends without performing any operation.

The master 2D shape data is not created.

c) The unit for the measurement data is always “mm”.

d) According to the parameter settings, correction, filtering, data structure conversion and reference data out are performed, and the master 2D shape data is created.

The details on each function are described in the following sections.

e) This function block corrects and converts a maximum of 20,000 measurement data points. So, it requires a considerable time to complete processing.

If this function block exclusively uses a long processing time, other processing may not be executed or a Task Period Exceeded error may occur. To avoid this, you need to set *NumProcData* (the number of processed data points) per period.

The function block performs correction and conversion for a specified number of data points within a period. In the next period, the processing is resumed where it was interrupted.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, consider the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

f) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

Note that the execution is completed when all the correction processing is finished in the period at which *Abort* changes to TRUE.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

g) If an error occurs, *Error* changes to TRUE and the function block execution is aborted. In addition, the error codes are output to *ErrorID* and *ErrorIDEx*.

For details on the error codes, refer to *Troubleshooting* on page 212.

h) If the function block is aborted or ended in an error, the master 2D shape data will be undefined.

Conversion to 2D Shape Data

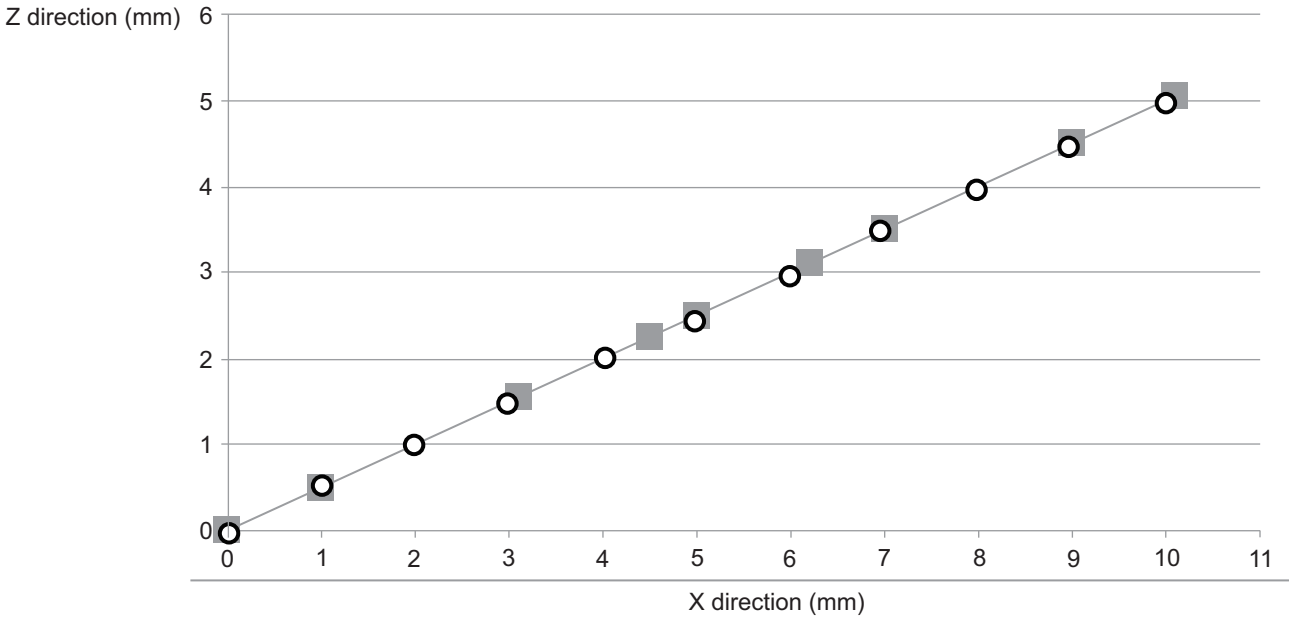
The function block converts the line measurement data to the 2D shape data.

For details on the line measurement data structure, refer to *Line Measurement Data* on page 157.

For details on the 2D shape data structure, refer to *2D Shape Data* on page 157.

The line measurement data with irregular intervals in the X direction is converted to the 2D shape data with regular intervals in the X direction.

The following is an example of conversion. The line measurement data with the resolution of 1,000 μm is expressed as "■" dots in the following figure. The data are not spaced regularly. The converted 2D shape data is expressed as "○" dots. The measurement points are set at intervals of the resolution so that the data is spaced regularly in X direction. If a Z-direction value does not exist in the line measurement data, it is calculated by interpolation using the neighboring measurement values.



The resolution used for measurement cannot be changed when the data is converted.

To change the resolution, you must redo the measurement.

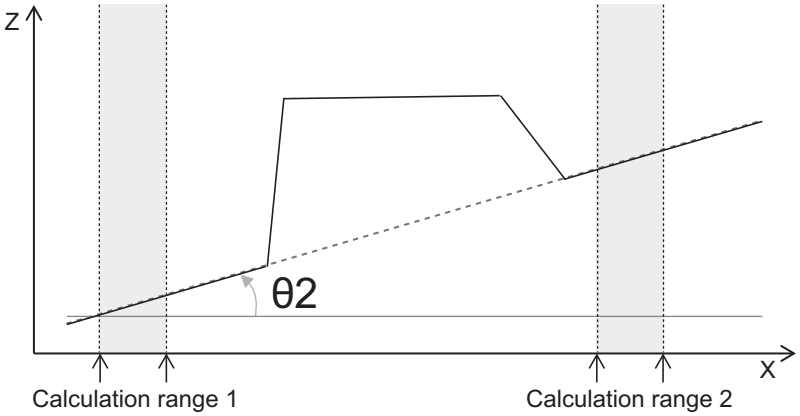
Slope Correction

When slope correction is selected (*sSlopeParams.Apply* = TRUE), the correction procedure is performed as follows.

(a) Input the correction angle θ1 (*sSlopeParams.BasicAngle*).

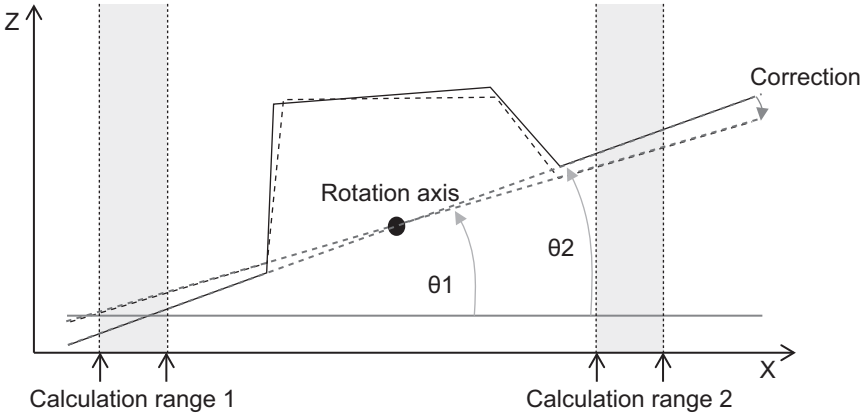
(b) When the slope of the measured object (θ2) is calculated, the straight line is created from all Z values included in Straight Line Calculation Range 1 (*sSlopeParams.Range1Low* to *sSlopeParams.Range1High*) and Straight Line Calculation Range 2 (*sSlopeParams.Range2Low* to *sSlopeParams.Range2High*).

If one or more array elements are not found in each range, a correction-impossible error occurs.



(c) If the slope of the measured object (θ2) is different from the correction angle θ1, the entire measurement data is rotated so that these angles can be the same.

The rotation axis is placed at the midpoint between *Range1High* and *Range2Low*.



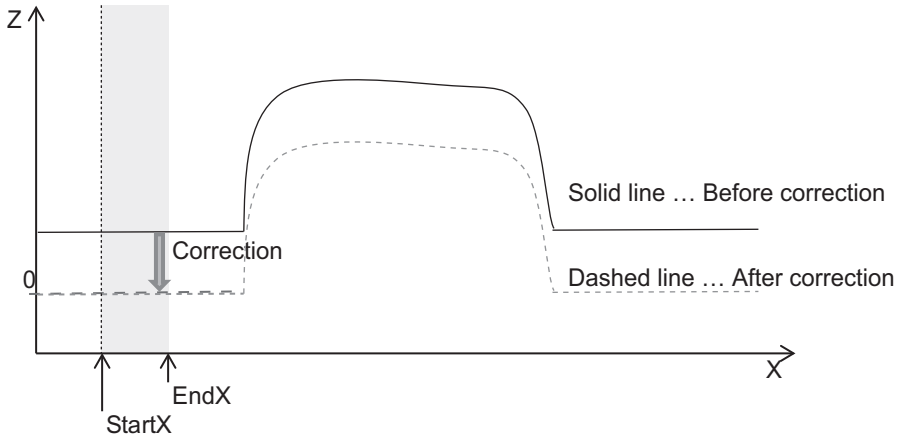
Height Correction

The height of the entire measurement data is corrected so that the heights within the height correction range can be 0.

(a) The function block determines the height of the measured object based on all Z values included in the height correction range (*sHeightParams.RangeLow* to *sHeightParams.RangeHigh*) and the method specified in Height Type (*sHeightParams.HeightType*).

If one or more array elements are not found in the range, a correction-impossible error occurs.

(b) If the calculated height is not 0, entire measurement data is moved up/down through correction so that the height can be 0.



Position Correction

Position correction is not performed for the master data.

Smoothing Filter

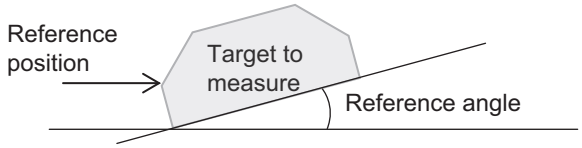
When the Number of Smoothing (*SmoothingFltNum*) is set to 1 or higher, the specified number of travel averaging is performed for each position.

Median Filter (X direction)

When the Median Range (*MedianFltNum*) is set to 1 or higher, the median processing is performed for each position within the specified range.

Reference Point Data Output

For the master 2D shape data, X-axis Measurement Start Position of the line measurement data is set as the origin of X coordinate. Then, the position at which the edge is detected is defined as the reference position. Also, the angle of the master 2D shape data relative to the horizontal surface is defined as the reference angle. The function block outputs the Edge Detection Index and Angle, from which the reference position of the master 2D shape data can be calculated. These data are called the reference point data. The reference point data is used when LineMeasure_CreateShape2D2 is executed. This data is required to correct deviation caused by the difference in the measurement environment between the master and target.



Reference Angle

The slope of the measurement data calculated in *Slope Correction* on page 205 is used as the setting value.

When slope correction is applied, the angle after correction is used.

Edge Detection Index

Edge detection position = Resolution × Edge detection index

It is expressed in the above-mentioned formula.

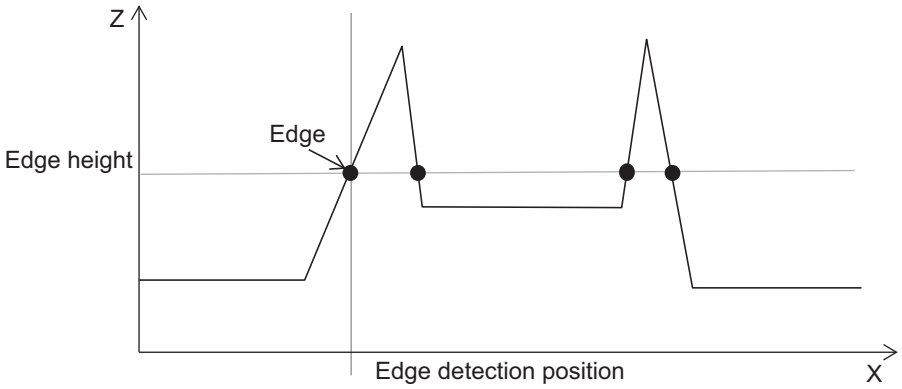
The edge detection index is calculated from the edge position detected through the following procedure.

a) The heights in the 2D shape data are checked in ascending order of X values.

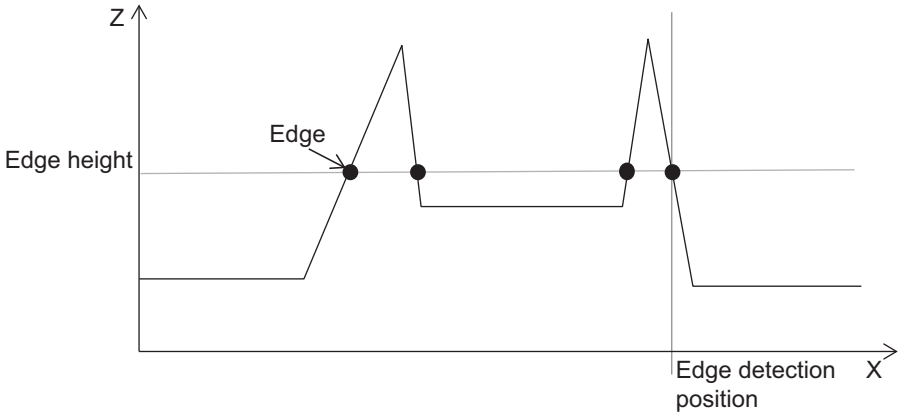
When a height exceeds Edge Height (*EdgeHeight*), an edge is judged to be detected at that point.

A detected edge is determined to be the edge detection position according to the setting of Number of Edge Detection (*EdgeNumber*).

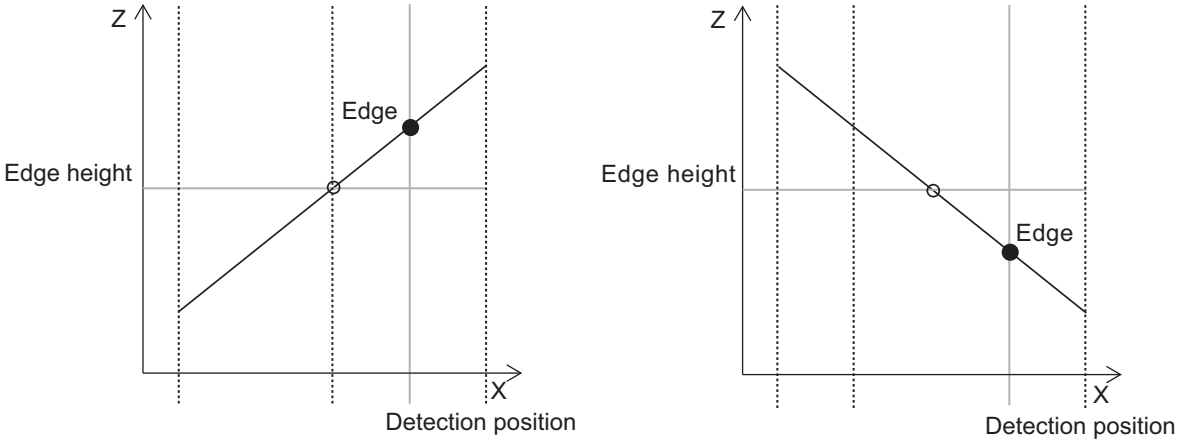
When Number of Edge Detection = 1



When Number of Edge Detection = 4



b) When a measured height is the same as Edge Height, it is not recognized as an edge.
Only a point after passing Edge Height is determined to be an edge detection position.



Re-execution of Function Blocks

If *Execute* is changed to TRUE again when execution is in progress (*Busy* = TRUE), the calculation in progress is aborted and the correction calculation is restarted with new parameters.

Multi-execution of Function Blocks

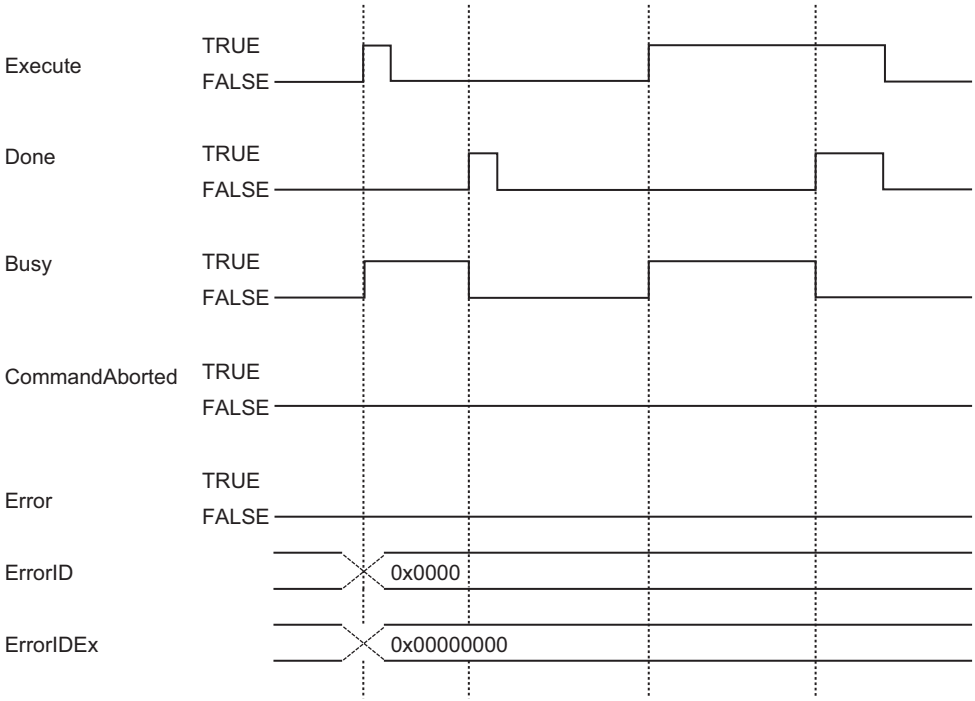
It is allowed to generate multiple instances from this function block and to execute multiple correction calculations simultaneously.

However, you must set different in-output variables for each instance when you assign them to the master 2D shape data.

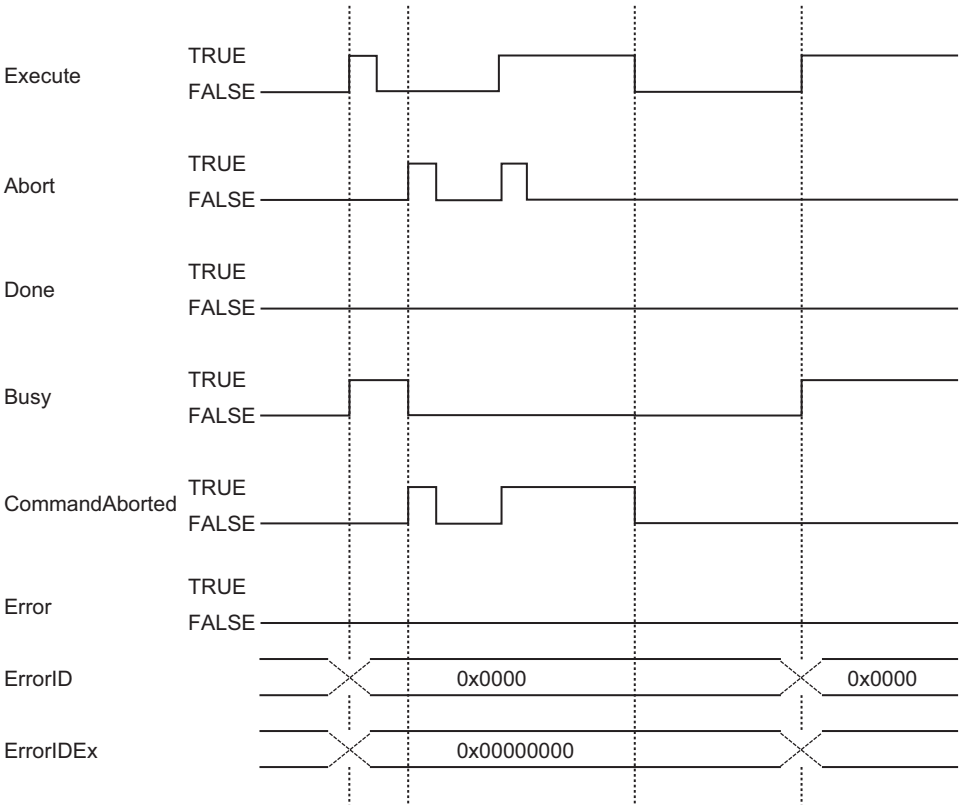
Timing Charts

The timing charts are shown below.

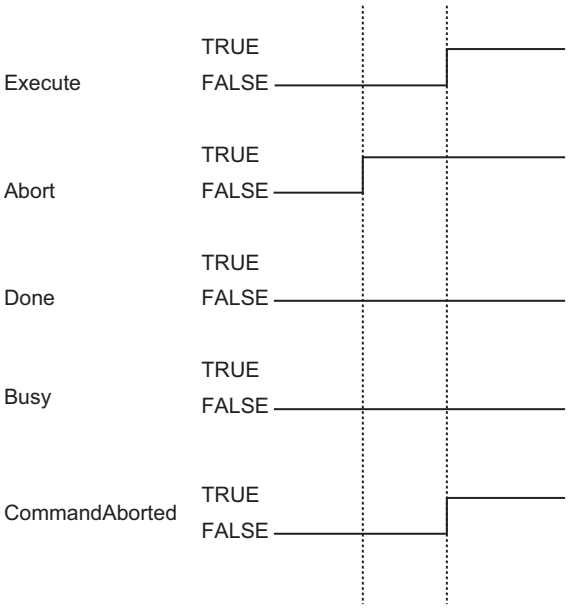
● Timing Chart in a Normal State



● Timing Chart When the Function Block Is Aborted

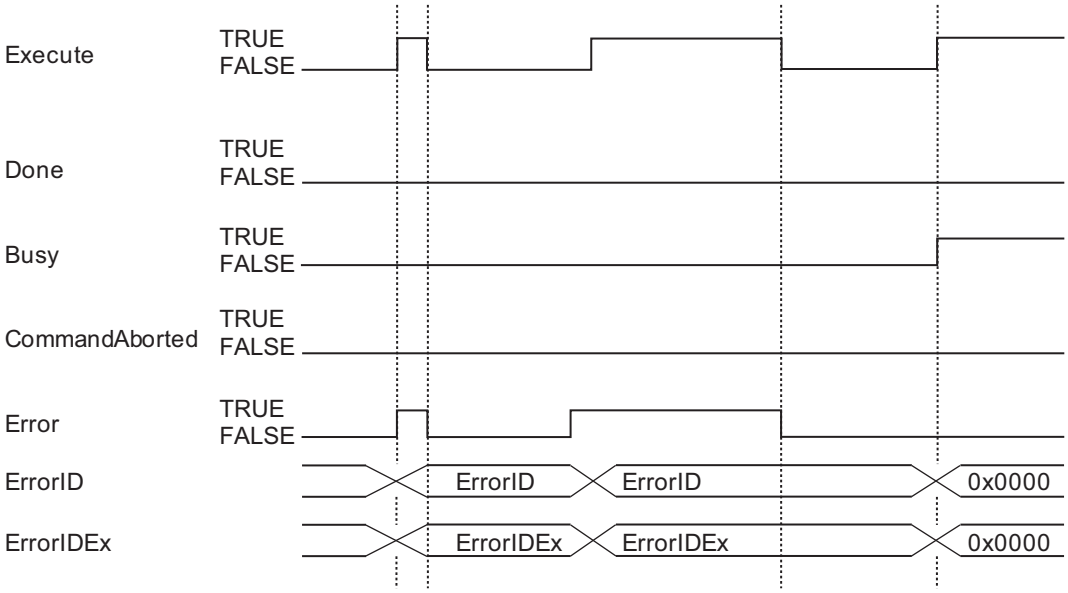


If *Execute* changes to TRUE while *Abort* is TRUE, *CommandAborted* (Interruption Completion) changes to TRUE without performing the processing.



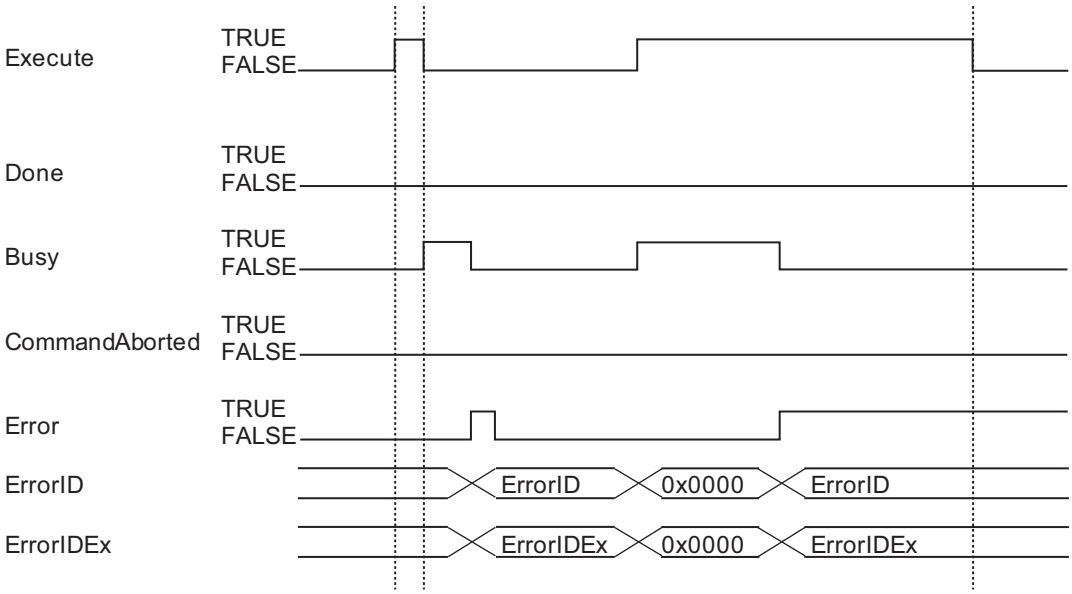
● Timing Chart When an Error Occurs (Startup Error)

If a setting value error is detected for the input variables when the function block is executed, a startup error occurs. *Error* changes to TRUE and the values are output to *ErrorID* (Error Code) and *ErrorIDE*x (Expansion Error Code). When *Execute* changes to FALSE, *Erro*r changes to FALSE accordingly. However, *ErrorID* and *ErrorIDEx* retain the values until the next function block is executed.



● Timing Chart When an Error Occurs (Error during Function Block Execution)

If an error is detected in the 2D shape data creation or correction during the function block execution, *Busy* changes to FALSE, *Error* changes to TRUE, and the values are output to *ErrorID* (Error Code) and *ErrorIDE*x (Expansion Error Code). When *Execute* changes to FALSE, *Erro*r changes to FALSE accordingly. However, *ErrorID* and *ErrorIDEx* retain the values until the next function block is executed.



Precautions for Correct Use

- Execute this function block only after the line measurement function block (LineMeasure_Cartesian2) ended normally. If the line measurement function block is aborted or ended in an error, the line measurement data is undefined. If you execute this function block with undefined line measurement data, the master 2D shape data cannot be created correctly.
- Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x0000	0x00000000	Normal end	—
0x3CD7	0x00000001	Number of Recorded Data (*LineMeasurementInfo.DataCount*) was set to a value that has exceeded the maximum number of stored measurement data.	Set a valid number of stored data so that the number of stored data does not exceed the maximum number of stored measurement data.
	0x00000002	Resolution (*LineMeasurementInfo.Resolution*) was set to 0.	Correct the input data.
	0x00000003	A reference point (slope) exceeding the valid range was detected from the measurement data that was input.	Correct the input data and slope correction setting.
	0x00000004	A reference point (slope) exceeding the valid range was detected from the measurement data that was input.	Correct the input data and slope correction setting.
	0x00000005	Slope correction was executed based on the reference angle detected from the measurement data that was input and Reference Angle (*SlopeParams.Angle*) in the slope correction setting. As the result, the valid range was exceeded.	Correct the input data and slope correction setting.
	0x00000006	The reference point (edge detection position) was not detected from the measurement data that was input.	Correct the input data and position correction setting.
	0x00000007	The value specified for Number of Smoothing (*SmoothingFltNum*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000008	The value specified for Median Range (*MedianFltNum*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000010	The value specified for Reference Angle of Slope Correction Parameter (*Angle*) is out of the valid range.	Specify a value which is within the valid range.

Error code	Expansion error code	Description	Corrective action
0x3CD7	0x00000011	Slope Correction Parameter The value specified for Straight Line Calculation Range 1 Upper Limit (*Range1High*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000012	Slope Correction Parameter The value specified for Straight Line Calculation Range 1 Lower Limit (*Range1Low*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000013	Slope Correction Parameter The value specified for Straight Line Calculation Range 2 Upper Limit (*Range2High*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000014	Slope Correction Parameter The value specified for Straight Line Calculation Range 2 Lower Limit (*Range2Low*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000015	Slope Correction Parameter The values specified for Straight Line Calculation Range 1 (*Range1Low and Range1High*) has met the following condition: lower limit ≥ upper limit.	Specify values that can meet the following condition: lower limit < upper limit.
	0x00000016	Slope Correction Parameter The values specified for Straight Line Calculation Range 2 (*Range2Low and Range2High*) has met the following condition: lower limit ≥ upper limit.	Specify values that can meet the following condition: lower limit < upper limit.
	0x00000017	Slope Correction Parameter The values specified for Straight Line Calculation Range 1 and 2 have met the following condition: *Range1High* ≥ *Range2Low*.	Specify values that can meet the following condition: *Range1High* < *Range2Low.*
	0x00000018	Slope Correction Parameter There is no measurement data that corresponds to the Straight Line Calculation Range 1.	Set Straight Line Calculation Range 1 to values that include the measurement data.
	0x00000019	Slope Correction Parameter There is no measurement data that corresponds to the Straight Line Calculation Range 2.	Set Straight Line Calculation Range 2 to values that include the measurement data.
	0x00000020	Height Correction Parameter The value specified for Height Calculation Range Upper Limit (*RangeHigh*) is out of the valid range.	Specify a value which is within the valid range.

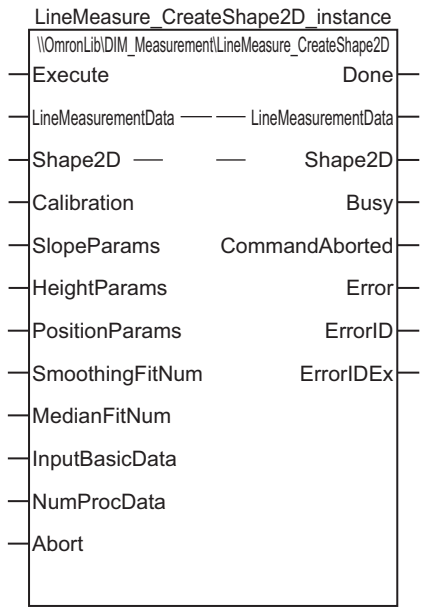
Error code	Expansion error code	Description	Corrective action
0x3CD7	0x00000021	Height Correction Parameter The value specified for Height Calculation Range Lower Limit (*RangeLow*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000022	Height Correction Parameter The values specified for Height Calculation Range (*RangeLow and RangeHigh*) has met the following condition: lower limit ≥ upper limit.	Specify values that can meet the following condition: lower limit < upper limit.
	0x00000023	Height Correction Parameter The value specified for Height Type (*HeightType*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000024	Height Correction Parameter There is no measurement data that corresponds to the Height Calculation Range.	Set Height Calculation Range to values that include the measurement data.
	0x00000030	Position Correction Parameter The value specified for Edge Height (*EdgeHeight*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000031	The value specified for Number of Edge Detection (*EdgeNumber*) of Position Correction Parameter is out of the valid range.	Specify a value which is within the valid range.
	0x00000050	The specified array for the line measurement data array (*LineMeasurementValues*) is out of the valid range.	The first number of array elements should be 0. Specify a value that the number of array elements is 20,000 or less.
	0x00000051	The specified array for the 2D shape data array (*ShapeZValues*) is out of the valid range.	The first number of array elements should be 0. Specify a value that the number of array elements is 20,000 or less.

Sample Programming

Refer to *Sample Programming* on page 168 for LineMeasure_Cartesian2.

LineMeasure_CreateShape2D

The LineMeasure_CreateShape2D function block creates the line measurement data and 2D shape data from input parameters.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
LineMeasure_ CreateShape2D	2D Shape Data Creation	FB		LineMeasure_CreateShape2D_instance(Execute, LineMeasurementData, Shape2D, Calibration, SlopeParams, HeightParams, PositionParams, InputBasicData, NumProcData, Abort, Done, Busy, CommandAborted, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V1_X.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00149
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Refer to *Compatible Models* on page 109 in the LineMeasure_Cartesian.

Variable

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
Calibration*1	Correction Method	0: Correction is performed based on the correction parameter and input reference data. (during target measurement) 1: Correction is performed based on the correction parameter. (during one-shot measurement)	USINT	0 to 1	0
SlopeParams *1	Slope Correction Parameter	Set the slope correction parameter.	sSlopeParams	—	—
HeightParams *1	Height Correction Parameter	Set the height correction parameter.	sHeightParams	—	—
Position Params*1	Position Correction Parameter	Set the position correction parameter.	sPositionParams	—	—
SmoothingFltNum*1	Number of Smoothing	0: No smoothing is performed 1: Travel averaging is performed twice 2: Travel averaging is performed 4 times 3: Travel averaging is performed 8 times 4: Travel averaging is performed 16 times 5: Travel averaging is performed 32 times 6: Travel averaging is performed 64 times 7: Travel averaging is performed 128 times 8: Travel averaging is performed 256 times	USINT	0 to 8	0
MedianFltNum *1	Median Range	0: No median processing is performed 1: Median Range is 3 2: Median Range is 5 3: Median Range is 7 4: Median Range is 9 5: Median Range is 15 6: Median Range is 31	USINT	0 to 6	0

Variable	Name	Description	Data type	Valid range	Default
InputBasic Data*1	Input Reference Data	Set the value that is output by LineMeasure_CreateShape2D_Master.	sBasicData	—	—
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a task period in the segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
Busy	Executing	Changes to TRUE when the function block is acknowledged.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 223.

In-Out Variables

Variable	Name	Description	Data type	Valid range
LineMeasurement Data	Line measurement data	Set the line measurement data output by LineMeasure_Cartesian.	sLineMeasurementData	—
Shape2D	2D Shape Data	Stores the 2D shape data after correction is performed.	sShape2D	—

Structure

Refer to the Structure for *LineMeasure_CreateShape2D_Master* on page 176.

Function

The LineMeasure_CreateShape2D function block provides the following functions.

(a) Conversion from the line measurement data to the 2D shape data

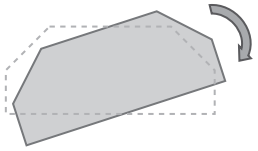
Refer to *Conversion from the line measurement data to the 2D shape data* on page 182.

(b) Measurement Correction

When Correction Method (*Calibration*) is set to 0, correction is performed based on the reference point data so that the target data has the same slope, height and position as the master 2D shape data as shown in a), b) and c) below.

a) Slope Correction

The function block corrects angular deviation caused by factors such as a difference in angle between master and target workpieces when they are measured.

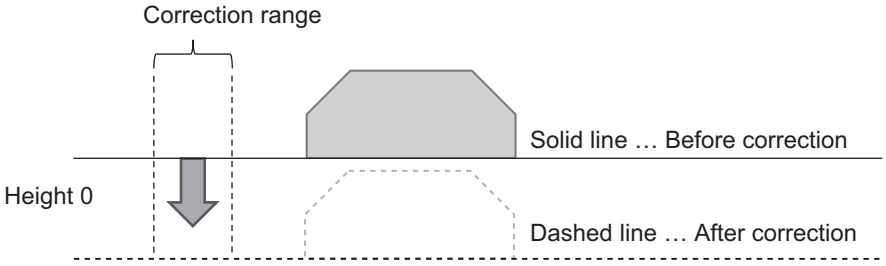


Solid line … Target 2D shape data

Dashed line … Master 2D shape data

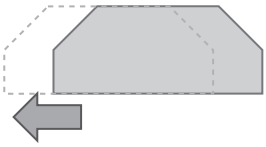
b) Height Correction

The function block corrects the height of the measurement surface within the correction range so that it becomes 0.



c) Position Correction

The function block corrects deviation caused by factors such as a difference in position between the master and target when they are measured.



Solid line … Before correction, dashed line … After correction

By using this function, deviation caused by differences in the measurement environment is corrected and you can compare the difference in shape between the master and target.

If Correction Method is set to 1, correction is performed according to *Measurement Correction* on page 182.

(c) Filtering

Refer to *Filtering* on page 182.

The following describes an overview of the entire processing.

a) The function block checks the input parameters when *Execute* changes to TRUE.

Only the parameters that are required for correction are checked.

If there is no error in the input data, the line measurement data that is input (*LineMeasurementData*) is corrected and filtered according to the correction and filtering settings and converted to the 2D shape data (*Shape2D*).

If an error is detected in the input parameters, the function block execution is aborted.

For details on the error codes, refer to *Troubleshooting* on page 223.

b) If there is no recorded data (*LineMeasurementData.DataCount* = 0), the function block execution ends without performing any operation.

The 2D shape data is not created.

c) The unit for the measurement data is always “mm”.

d) According to the parameter setting, correction, filtering and data structure conversion are performed, and the 2D shape data are created.

The details on each function are described in the following sections.

e) When you specify 0 for Correction Method (*Calibration*), assign the members of the master 2D shape data to the following input variables.

- *SlopeParams*
- *HeightParams*
- *PositionParams*
- *SmoothingFltNum*
- *MedianFltNum*
- *InputBasicData*

f) The following shows whether the input variable needs to be set or not when you specify 1 for Correction Method (*Calibration*).

- *SlopeParams*

  Required only for slope correction
- *HeightParams*

  Required
- *PositionParams*

  Not Required
- *SmoothingFltNum*

  Required
- *MedianFltNum*

  Required
- *InputBasicData*

  Not Required

g) This function block corrects and converts a maximum of 20,000 measurement data points. So, it requires a considerable time to complete processing.

If this function block exclusively uses a long processing time, other processing may not be executed or a Task Period Exceeded error may occur. To avoid this, you need to set the number of processed data points *(NumProcData)* per period.

The function block performs correction and conversion for a specified number of data points within a period. In the next period, the processing is resumed where it was interrupted.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, consider the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

h) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

Note that the execution is completed if all correction processing is finished in the period at which *Abort* changes to TRUE.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

i) If an error occurs, *Error* changes to TRUE and the function block execution is aborted. In addition, the error codes are output to *ErrorID* and *ErrorIDEx*.

For details on the error codes, refer to *Troubleshooting* on page 223.

j) If the function block is aborted or ended in an error, the line master 2D shape data may be undefined.

Conversion to 2D Shape Data

This function is the same as LineMeasure_CreateShape2D_Master.

Refer to *Conversion to 2D Shape Data* on page 184.

Slope Correction

When slope correction is selected (*sSlopeParams.Apply* = TRUE), the correction procedure is performed as follows.

If 1 is specified for Correction Method (*Calibration*), the performed function is the same as LineMeasure_CreateShape2D_Master.

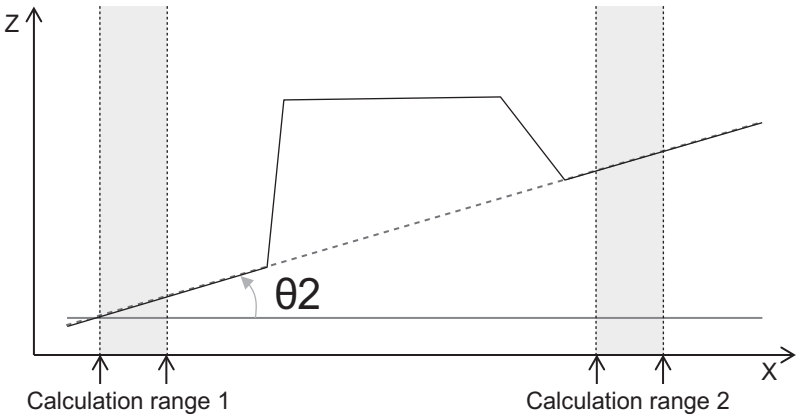
Refer to *Slope Correction* on page 185.

The following description applies to Correction Method (*Calibration*) set to 0.

(a) Assign the reference point data (*sShapeMasterData.BasicData*) of the master 2D shape data output by LineMeasure_CreateShape2D_Master to the Input Reference Data (*InputBasicData*) input variable. The reference angle (*BasicData.Angle*) in Input Reference Data is expressed as θ1.

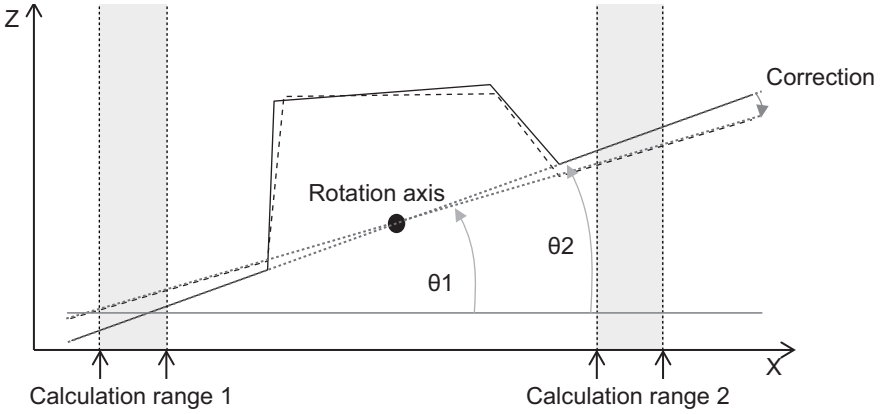
(b) When the slope of the measured object (θ2) is calculated, the straight line is created from all Z values included in Straight Line Calculation Range 1 (*sSlopeParams.Range1Low* to *sSlopeParams.Range1High*) and Straight Line Calculation Range 2 (*sSlopeParams.Range2Low* to *sSlopeParams.Range2High*).

If one or more array elements are not found in each range, a correction-impossible error occurs.



(c) If the slope θ2 of the measured object is different from the angle θ1 of the reference point data, the entire measurement data is rotated so that these angles can be the same.

The rotation axis is placed at the midpoint between *Range1High* and *Range2Low*.



Height Correction

This function is the same as LineMeasure_CreateShape2D_Master.

Refer to *Height Correction* on page 186.

Position Correction

If Correction Method (*Calibration*) is set to 1, position correction is not performed.

Refer to *Slope Correction* on page 185.

The following description applies to Correction Method (*Calibration*) set to 0.

(a) Assign the reference point data (*sShapeMasterData.BasicData*) of the master 2D shape data output by LineMeasure_CreateShape2D_Master to the Input Reference Data (*InputBasicData*) input variable.

(b) The procedure to detect Edge Detection Index is the same as the reference point data output function of LineMeasure_CreateShape2D_Master.

Refer to the description of *Edge Detection Index* on page 187 in the *Reference Point Data Output* on page 187.

Smoothing Filter

This function is the same as LineMeasure_CreateShape2D_Master.

Refer to *Smoothing Filter* on page 186.

Median Filter (X direction)

This function is the same as LineMeasure_CreateShape2D_Master.

Refer to *Median Filter (X direction)* on page 186.

Re-execution of Function Blocks

If you change *Execute* to TRUE when execution is in progress (*Busy* = TRUE), the calculation in progress is aborted and the correction calculation is restarted with new parameters.

Multi-execution of Function Blocks

It is allowed to generate multiple instances from this function block and to execute multiple correction calculations simultaneously.

However, you must set different variables for each instance when you assign them to the 2D shape data.

Timing Charts

Refer to *Timing Charts* on page 189 for LineMeasure_CreateShape2D.

Precautions for Correct Use

- Execute this function block only after the line measurement function block (LineMeasure_Cartesian) is completed normally. If the line measurement function block is aborted or ended in an error, the line measurement data is undefined. If you execute this function block with undefined line measurement data, the master 2D shape data cannot be created correctly.
- If *Calibration* = 0, execute this function block using the master 2D shape data as the input variables after the master 2D shape data creation function block (LineMeasure_CreateShap2D_Master) is ended normally. If the master 2D shape data creation function block is aborted or ended in an error, the master 2D shape data may be undefined, and the 2D shape data may not be created correctly.
- Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3CD8	0x0000 0001	Number of Recorded Data (*LineMeasurementData.DataCount*) was set to a value that exceeds the maximum number of stored measurement data.	Set a valid number of stored data.
	0x0000 0002	Resolution (*LineMeasurement-Data.Resolution*) was set to 0.	Correct the input data.
	0x0000 0003	The X values of the measurement data that was input is not stored in ascending order.	Correct the input data.
	0x0000 0004	A reference point (slope) exceeding the valid range was detected from the measurement data that was input.	Correct the input data and slope correction setting.
	0x0000 0005	Slope correction was executed based on the reference angle detected from the measurement data that was input and Reference Angle (*SlopeParams.Angle*) in the slope correction setting. As the result, the valid range was exceeded.	Correct the input data and slope correction setting.
	0x0000 0006	The reference point (edge detection position) was not detected from the measurement data that was input.	Correct the input data and position correction setting.
	0x0000 0007	The value specified for Number of Smoothing (*SmoothingFltNum*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0008	The value specified for Median Range (*MedianFltNum*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0009	The value specified for Correction Method (*Calibration*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0010	Slope Correction Parameter The value specified for Reference Angle (*Angle*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0011	Slope Correction Parameter The value specified for Straight Line Calculation Range 1 Upper Limit (*Range1High*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0012	Slope Correction Parameter The value specified for Straight Line Calculation Range 1 Lower Limit (*Range1Low*) is out of the valid range.	Specify a value which is within the valid range.

Error code	Expansion error code	Description	Corrective action
0x3CD8	0x0000 0013	Slope Correction Parameter The value specified for Straight Line Calculation Range 2 Upper Limit (*Range1High*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0014	Slope Correction Parameter The value specified for Straight Line Calculation Range 2 Lower Limit (*Range2Low*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0015	Slope Correction Parameter The values specified for Straight Line Calculation Range 1 (*Range1Low* and *Range1High*) has met the following condition: lower limit ≥ upper limit.	Specify values that can meet the following condition: lower limit < upper limit.
	0x0000 0016	Slope Correction Parameter The values specified for Straight Line Calculation Range 2 (*Range2Low* and *Range2High*) has met the following condition: lower limit ≥ upper limit.	Specify values that can meet the following condition: lower limit < upper limit.
	0x0000 0017	Slope Correction Parameter The values specified for Straight Line Calculation Range 1 and 2 have met the following condition: *Range1High* ≥ *Range2Low*.	Specify values that can meet the following condition: *Range1High* < *Range2Low*.
	0x0000 0018	Slope Correction Parameter There is no measurement data that corresponds to the Straight Line Calculation Range 1.	Set Straight Line Calculation Range 1 to values that include the measurement data.
	0x0000 0019	Slope Correction Parameter There is no measurement data that corresponds to the Straight Line Calculation Range 2.	Set Straight Line Calculation Range 2 to values that include the measurement data.
	0x0000 0020	Height Correction Parameter The value specified for Height Calculation Range Upper Limit (*RangeHigh*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0021	Height Correction Parameter The value specified for Height Calculation Range Lower Limit (*RangeLow*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0022	Height Correction Parameter The values specified for Height Calculation Range (*RangeLow* and *RangeHigh*) has met the following condition: lower limit ≥ upper limit.	Specify values that can meet the following condition: lower limit < upper limit.
	0x0000 0023	Height Correction Parameter The value specified for Height Type (*HeightType*) is out of the valid range.	Specify a value which is within the valid range.

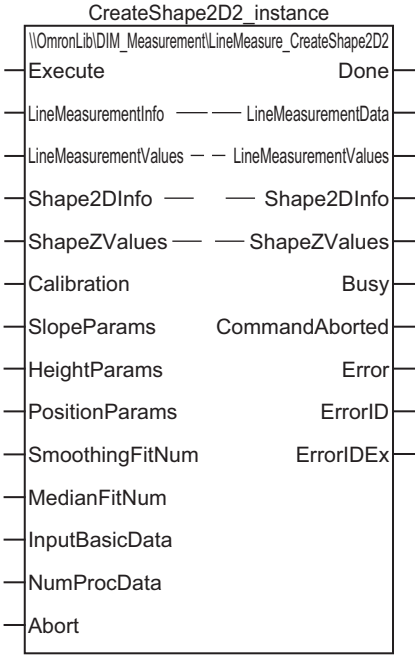
Error code	Expansion error code	Description	Corrective action
0x3CD8	0x0000 0024	Height Correction Parameter There is no measurement data that corresponds to the Height Calculation Range.	Set Height Calculation Range to values that include the measurement data.
	0x0000 0030	Position Correction Parameter The value specified for Edge Height (*EdgeHeight*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0031	Position Correction Parameter The value specified for Number of Edge Detection (*EdgeNumber*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0040	Input Reference Data The value specified for Angle (*Angle*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0041	Input Reference Data The value specified for Edge Detection Index (*EdgeIndex*) is out of the valid range.	Specify a value which is within the valid range.

Sample Programming

Refer to *Sample Programming* on page 143 for LineMeasure_Cartesian.

LineMeasure_CreateShape2D2

The LineMeasure_CreateShape2D2 function block creates 2D shape data from the line measurement data and input parameters. 2D shape data is a variable-length data array.

Function block and function name	Name	FB/ FUN	Graphic expression	ST expression
LineMeasure_CreateShape2D2	2D Shape Data Creation 2	FB		CreateShape2D2_instance(Execute, LineMeasurementInfo, LineMeasurementValues, Shape2DInfo, ShapeZValues, Calibration, SlopeParams, HeightParams, PositionParams, SmoothingFltNum, MedianFltNum, InputBasicData, NumProcData, Abort, Done, Busy, CommandAborted, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_ DIM_LineMeasurement_V2_0.slr
Namespace	OmronLib\ DIM_Measurement
Function block and function number	00229
Source code	Not published

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
Calibration*1	Correction Method	0: Correction is performed based on the correction parameter and input reference data. (during target measurement) 1. Correction is performed based on the correction parameter. (during one-shot measurement)	USINT	0 to 1	0
SlopeParams *1	Slope Correction Parameter	Set the slope correction parameter.	sSlopeParams	—	—
HeightParams *1	Height Correction Parameter	Set the height correction parameter.	sHeightParams	—	—
Position Params*1	Position Correction Parameter	Set the position correction parameter.	sPositionParams	—	—
SmoothingFlt Num*1	Number of Smoothing	0: No smoothing is performed 1: Travel averaging is performed twice 2: Travel averaging is performed 4 times 3: Travel averaging is performed 8 times 4: Travel averaging is performed 16 times 5: Travel averaging is performed 32 times 6: Travel averaging is performed 64 times 7: Travel averaging is performed 128 times 8: Travel averaging is performed 256 times	USINT	0 to 8	0
MedianFltNum*1	Median Range	0: No median processing is performed 1: Median Range is 3 2: Median Range is 5 3: Median Range is 7 4: Median Range is 9 5: Median Range is 15 6: Median Range is 31	USINT	0 to 6	0

Variable	Name	Description	Data type	Valid range	Default
InputBasic Data*1	Input Reference Data	Set the value that is output by LineMeasure_CreateShape2D_Master2.	sBasicData	—	—
NumProcData *1	Number of Processed Data Points	Specify the number of data points that are processed in a period during segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
Busy	Executing	TRUE when *Execute* is acknowledged.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 273 for details.

In-Out Variables

Variable	Name	Description	Data type	Valid range
LineMeasurementInfo	Line Measurement Information	Set the line measurement information output by LineMeasure_Cartesian2.	sLineMeasurementInfo	—
LineMeasurementValues[]*1	Line Measurement Data Array	Set the line measurement data array output by LineMeasure_Cartesian2.	ARRAY[*] OF sLineMeasurementValue	—
Shape2DInfo	2D Shape Data Information	Stores the 2D shape data information after correction is performed.	sShape2DInfo	—
ShapeZValues[] *1	2D Shape Data Array	Stores the 2D shape data array after correction is performed.	ARRAY[*] OF LREAL	—

*1. The maximum number of array elements is 20,000. The first number of array element should be 0.

Structure

Refer to the structure for *LineMeasure_CreateShape2D_Master2* on page 195.

Function

The LineMeasure_CreateShape2D2 function block provides the following functions.

(a) Conversion from the line measurement data to the 2D shape data

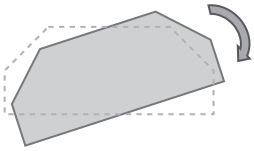
Refer to *Conversion from the line measurement data to the 2D shape data* on page 182 for details.

(b) Measurement Correction

When Correction Method (*Calibration*) is set to 0, correction is performed based on the reference point data so that the target data has the same slope, height and position as the master 2D shape data as shown in a), b) and c) below.

a) Slope Correction

The function block corrects angular deviation caused by factors such as a difference in angle between master and target workpieces when they are measured.

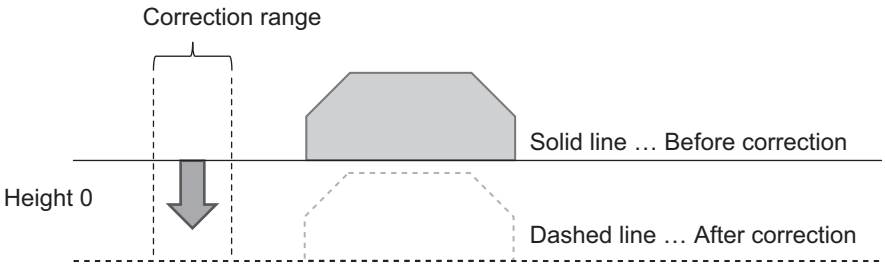


Solid line … Target 2D shape data

Dashed line … Master 2D shape data

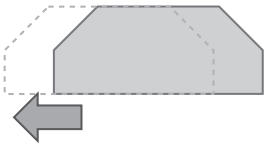
b) Height Correction

The function block corrects the height of the measurement surface within the correction range so that it becomes 0.



c) Position Correction

The function block corrects deviation caused by factors such as a difference in position between the master and target when they are measured.



Solid line … Before correction, dashed line … After correction

By using this function, deviation caused by differences in the measurement environment is corrected and you can compare the difference in shape between the master and target.

If Correction Method is set to 1, correction is performed according to *Measurement Correction* on page 201.

(c) Filtering

Refer to *Filtering* on page 202 for details.

Depending on the shape of the target or condition of the surface, the line measurement data may be unstable.In such a case, you can use the filters such as Smoothing and Median (X direction) to reduce noises.

a) Smoothing

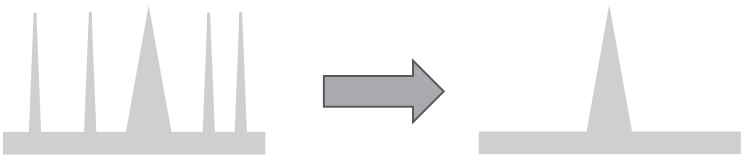
At each position in X direction, travel averaging is performed according to the specified number of smoothing. You can create the data of a smooth shape.

b) Median (X direction)

The function block sorts out *ZValue* of the line measurement data within a range specified by Median Range in which the X position is set as the center.

The center value is replaced with *ZValue* at the X position.

By applying the Median filter, you can remove spikes that are contained in the shape data.



(d) Reference Point Data Calculation

This function block outputs the edge detection index and reference angle. These data are called the reference point data. The reference point data calculated by this function block is used for the setting when executing LineMeasure_CreateShape2D2. This data is required for correcting the target 2D shape data so that deviations caused by the difference in the measurement environment between master and target can be reduced. The details on the edge detection index and reference angle are described in *Reference Point Data Output* on page 206.

By using this function, deviation caused by differences in the measurement environment is corrected and you can compare the difference in shape between the master and target.

The following describes an overview of the entire processing.

a) The function block checks the input parameters when *Execute* changes to TRUE.

Only the parameters that are required for correction are checked.

If there is no error in the input data, the input line measurement data (*LineMeasurementValues[]*) is corrected and filtered according to the correction and filtering settings and converted to the 2D shape data (*ShapeZValues[]*).

If an error is detected in the input parameters, the function block execution is aborted.

For details on the error codes, refer to *Troubleshooting* on page 234.

b) If there is no recorded data (*LineMeasurementInfo.DataCount* = 0), the function block execution ends without performing any operation. The 2D shape data is not created.

c) The unit for the measurement data is always “mm”.

d) According to the parameter setting, correction, filtering and data structure conversion are performed, and the 2D shape data is created.

The details on each function are described in the following sections.

e) When you specify 0 for Correction Method (*Calibration*), assign the members of the master 2D shape data to the following input variables.
   - *SlopeParams*
   - *HeightParams*
   - *PositionParams*
   - *SmoothingFltNum*
   - *MedianFltNum*
   - *InputBasicData*

f) The following shows whether the input variable needs to be set or not when you specify 1 for Correction Method (*Calibration*).
   - *SlopeParams*
     Required only for slope correction.
   - *HeightParams*
     Required.
   - *PositionParams*
     Not required.
   - *SmoothingFltNum*
     Required.
   - *MedianFltNum*
     Required.
   - *InputBasicData*
     Not required.

g) This function block corrects and converts a maximum of 20,000 measurement data points. So, it requires a considerable time to complete processing.

   If this function block exclusively uses a long processing time, other processing may not be executed or a Task Period Exceeded error may occur. To avoid this, you need to set the number of processed data points (*NumProcData*) per period.

   The function block performs correction and conversion for a specified number of data points within a period. In the next period, the processing is resumed where it was interrupted.

   If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

   If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

   When you decide the number of processed data points, consider the execution environment, unused time in the task period available when correction is executed, and other factors.

   If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

   If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

h) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

   Note that the execution is completed if all correction processing is finished in the period at which *Abort* changes to TRUE.

   If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

i) If an error occurs, *Error* changes to TRUE and the function block execution is aborted. In addition, the error codes are output to *ErrorID* and *ErrorIDEx*.

   For details on the error codes, refer to *Troubleshooting* on page 234.

j) If the function block is aborted or ended in an error, the line master 2D shape data may be undefined.

Conversion to 2D Shape Data

This function is the same as LineMeasure_CreateShape2D_Master2.

Refer to *Conversion to 2D Shape Data* on page 204 for details.

Slope Correction

When slope correction is selected (*sSlopeParams.Apply* = TRUE), the correction procedure is performed as follows.

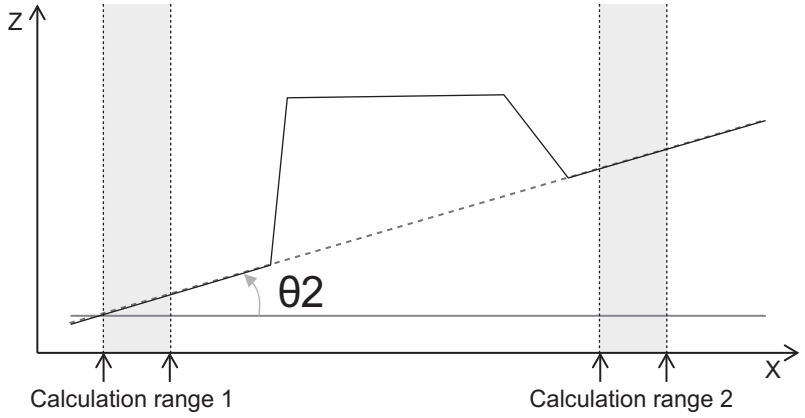
If 1 is specified for Correction Method (*Calibration*), the performed function is the same as LineMeasure_CreateShape2D_Master2.

Refer to *Slope Correction* on page 205 for details.

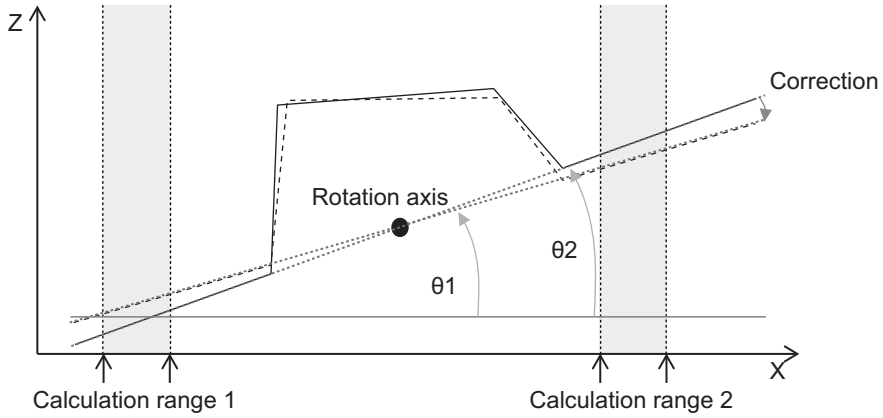
The following description applies to Correction Method (*Calibration*) set to 0.

(a) Assign the reference point data (*sShapeMasterData.BasicData*) of the master 2D shape data output by LineMeasure_CreateShape2D_Master to the Input Reference Data (*InputBasicData*) input variable. The reference angle (*BasicData.Angle*) in Input Reference Data is expressed as θ1.

(b) When the slope of the measured object (θ2) is calculated, the straight line is created from all Z values included in Straight Line Calculation Range 1 (*sSlopeParams.Range1Low* to *sSlopeParams.Range1High*) and Straight Line Calculation Range 2 (*sSlopeParams.Range2Low* to *sSlopeParams.Range2High*).
If one or more array elements are not found in each range, a correction-impossible error occurs.



(c) If the slope θ2 of the measured object is different from the angle θ1 of the reference point data, the entire measurement data is rotated so that these angles can be the same.
The rotation axis is placed at the midpoint between *Range1High* and *Range2Low*.



Height Correction

This function is the same as LineMeasure_CreateShape2D_Master2.

Refer to *Height Correction* on page 206 for details.

Position Correction

If Correction Method (*Calibration*) is set to 1, position correction is not performed.

Refer to *Slope Correction* on page 205.

The following description applies to Correction Method (*Calibration*) set to 0.

(a) Assign the reference point data (*sShapeMasterData.BasicData*) of the master 2D shape data output by LineMeasure_CreateShape2D_Master2 to the Input Reference Data (*InputBasicData*) input variable.

(b) The procedure to detect Edge Detection Index is the same as the reference point data output function of LineMeasure_CreateShape2D_Master.

Refer to the description of Edge Detection Index in *Reference Point Data Output* on page 206.

Smoothing Filter

This function is the same as LineMeasure_CreateShape2D_Master2.

Refer to *Smoothing Filter* on page 206 for details.

Median Filter (X direction)

This function is the same as LineMeasure_CreateShape2D_Master2.

Refer to *Median Filter (X direction)* on page 206 for details.

Re-execution of Function Blocks

If *Execute* is changed to TRUE again when execution is in progress (*Busy* = TRUE), the calculation in progress is aborted and the correction calculation is restarted with new parameters.

Multi-execution of Function Blocks

It is allowed to generate multiple instances from this function block and to execute multiple correction calculations simultaneously.

However, you must set different variables for each instance when you assign them to the 2D shape data.

Timing Charts

Refer to *Timing Charts* on page 209 for LineMeasure_CreateShape2D_Master2.

Precautions for Correct Use

- Execute this function block only after the line measurement function block (LineMeasure_Cartesian2) ended normally. If the line measurement function block is aborted or ended in an error, the line measurement data is undefined. If you execute this function block with undefined line measurement data, the master 2D shape data cannot be created correctly.
- If *Calibration* = 0, execute this function block using the master 2D shape data as the input variables after the master 2D shape data creation function block (LineMeasure_CreateShap2D_Master2) is ended normally. If the master 2D shape data creation function block is aborted or ended in an error, the master 2D shape data may be undefined, and the 2D shape data may not be created correctly.
- Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x0000	0x00000000	Normal end	—
0x3CD8	0x00000001	Number of Recorded Data (*LineMeasurementInfo.DataCount*) was set to a value that has exceeded the maximum number of stored measurement data.	Set a valid number of stored data.
	0x00000002	Resolution (*LineMeasurementInfo.Resolution*) was set to 0.	Correct the input data.
	0x00000003	The X values of the measurement data that was input is not stored in ascending order.	Correct the input data.
	0x00000004	A reference point (slope) exceeding the valid range was detected from the measurement data that was input.	Correct the input data and slope correction setting.
	0x00000005	Slope correction was executed based on the reference angle detected from the measurement data that was input and Reference Angle (*SlopeParams.Angle*) in the slope correction setting. As the result, the valid range was exceeded.	Correct the input data and slope correction setting.
	0x00000006	The reference point (edge detection position) was not detected from the measurement data that was input.	Correct the input data and position correction setting.
	0x00000007	The value specified for Number of Smoothing (*SmoothingFltNum*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000008	The value specified for Median Range (*MedianFltNum*) is out of the valid range.	Specify a value which is within the valid range.

Error code	Expansion error code	Description	Corrective action
0x3CD8	0x00000009	The value specified for Correction Method (*Calibration*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000010	Slope Correction Parameter The value specified for Reference Angle (*Angle*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000011	Slope Correction Parameter The value specified for Straight Line Calculation Range 1 Upper Limit (*Range1High*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000012	Slope Correction Parameter The value specified for Straight Line Calculation Range 1 Lower Limit (*Range1Low*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000013	Slope Correction Parameter The value specified for Straight Line Calculation Range 2 Upper Limit (*Range2High*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000014	Slope Correction Parameter The value specified for Straight Line Calculation Range 2 Lower Limit (*Range2Low*) is out of the valid range.	Specify a value which is within the valid range.
	0x00000015	Slope Correction Parameter The values specified for Straight Line Calculation Range 1 (*Range1Low* and *Range1High*) has met the following condition: lower limit ≥ upper limit.	Specify values that can meet the following condition: lower limit < upper limit.

Error code	Expansion error code	Description	Corrective action
0x3CD8	0x00000016	Slope Correction Parameter The values specified for Straight Line Calculation Range 2 (*Range2Low* and *Range2High*) has met the following condition: lower limit ≥ upper limit.	Specify values that can meet the following condition: lower limit < upper limit.
	0x00000017	Slope Correction Parameter The values specified for Straight Line Calculation Range 1 and 2 have met the following condition: *Range1High* ≥ *Range2Low.*	Specify values that can meet the following condition: *Range1High* < *Range2Low.*
	0x00000018	Slope Correction Parameter There is no measurement data that corresponds to the Straight Line Calculation Range 1.	Set Straight Line Calculation Range 1 to values that include the measurement data.
	0x00000019	Slope Correction Parameter There is no measurement data that corresponds to the Straight Line Calculation Range 2.	Set Straight Line Calculation Range 2 to values that include the measurement data.
	0x00000020	Height Correction Parameter The value specified for Height Calculation Range Upper Limit *(RangeHigh)* is out of the valid range.	Specify a value which is within the valid range.

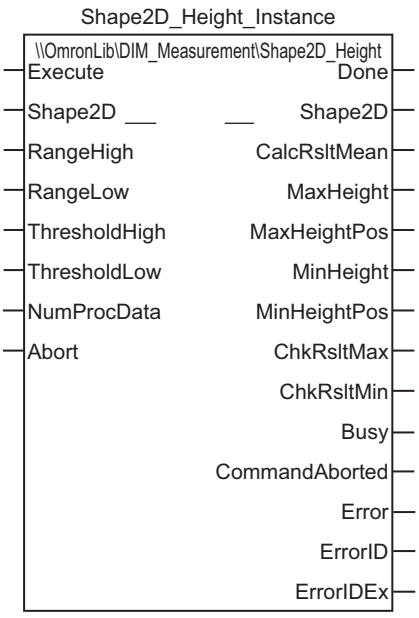
Error code	Expansion error code	Description	Corrective action
0x3CD8	0x0000 0021	Height Correction Parameter The value specified for Height Calculation Range Lower Limit (*RangeLow*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0022	Height Correction Parameter The values specified for Height Calculation Range (*RangeLow* and *RangeHigh*) has met the following condition: lower limit ≥ upper limit.	Specify values that can meet the following condition: lower limit < upper limit.
	0x0000 0023	Height Correction Parameter The value specified for Height Type (*HeightType*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0024	Height Correction Parameter There is no measurement data that corresponds to the Height Calculation Range.	Set Height Calculation Range to values that include the measurement data.
	0x0000 0030	Position Correction Parameter The value specified for Edge Height (*EdgeHeight*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0031	Position Correction Parameter The value specified for Number of Edge Detection (*EdgeNumber*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0040	Input Reference Data The value specified for Angle (*Angle*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0041	Input Reference Data The value specified for Edge Detection Index (*EdgeIndex*) is out of the valid range.	Specify a value which is within the valid range.
	0x0000 0050	The specified array for the line measurement data array (*LineMeasurementValues[]*) is out of the valid range.	The first number of array elements should be 0. Specify a value that the number of array elements is 20,000 or less.
	0x0000 0051	The specified array for the 2D shape data array (*ShapeZValues[]*) is out of the valid range.	The first number of array elements should be 0. Specify a value that the number of array elements is 20,000 or less.

Sample Programming

Refer to *Sample Programming* on page 168 for LineMeasure_Cartesian2.

Shape2D_Height

The Shape2D_Height function block measures heights in a specified measurement range of the 2D shape data.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
Shape2D_Height	2D Shape Height Measurement	FB		Shape2D_Height_Instance (Execute, Shape2D, RangeHigh, RangeLow, ThresholdHigh, ThresholdLow, NumProcData, Abort, Done, CalcRsltMean, MaxHeight, MaxHeightPos, MinHeight, MinHeightPos, ChkRsltMax, ChkRsltMin, Busy, CommandAborted, Error, ErrorID, ErrorIDEx)

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V1_1.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00141
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Product name	Model numbers	Version
Device	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
RangeHigh*1	Measurement Range Upper Limit	Specify the upper limit of the X-axis measurement range that is used for height measurement. The unit is mm.	LREAL	Negative number, positive number, or 0	0
RangeLow*1	Measurement Range Lower Limit	Specify the lower limit of the X-axis measurement range that is used for height measurement. The unit is mm.	LREAL	Negative number, positive number, or 0	0
ThresholdHigh*1	Threshold (High)	These thresholds are used to determine whether a measurement result is within the range. The unit is mm. Specify values always as: Threshold (High) ≥ Threshold (Low).	LREAL	Negative number, positive number, or 0	0
ThresholdLow*1	Threshold (Low)		LREAL	Negative number, positive number, or 0	0
NumProcData *1	Number of Processed Data Points	Specify the number of data points that are processed in a task period in the segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
CalcRsltMean	Measurement Result Mean Value	Outputs the mean value of heights in the measurement range. The unit is mm.	LREAL	Depends on data type.	—
MaxHeight	Maximum Height	Outputs the maximum height in the measurement range. The unit is mm.	LREAL	Depends on data type.	—
MaxHeightPos	Maximum Height Position	Outputs the position of the maximum height in the measurement range. The unit is mm.	LREAL	Depends on data type.	—
MinHeight	Minimum Height	Outputs the minimum height in the measurement range. The unit is mm.	LREAL	Depends on data type.	—
MinHeightPos	Minimum Height Position	Outputs the position of the minimum height in the measurement range. The unit is mm.	LREAL	Depends on data type.	—
ChkRsltMax	Maximum Value Judgment Result	TRUE when Maximum Height is equal to or less than Threshold (High).	BOOL	TRUE or FALSE	—
ChkRsltMin	Minimum Value Judgment Result	TRUE when Minimum Height is equal to or greater than Threshold (Low).	BOOL	TRUE or FALSE	—
Busy	Measuring	TRUE when the measurement is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 246.

In-Out Variables

Variable	Name	Description	Data type	Valid range
Shape2D	2D Shape Data	Specify the 2D shape data. Refer to *LineMeasure_Cartesian* on page 108 for 2D shape data creation and correction.	sShape2D	—

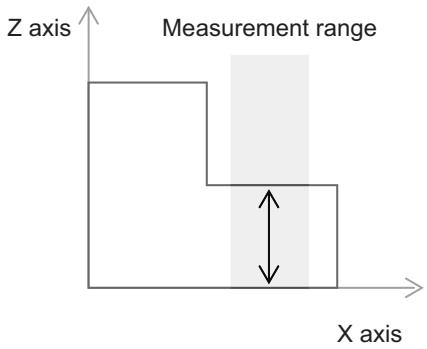
Members of Structure sShape2D

Member	Name	Data type	Valid range	Description
DataCount	Number of Data Points	UINT	0 to 20,000	The number of data points are stored.
Resolution	Resolution	UINT	Positive number	The resolution in X direction*1 The unit is μm.
ZValue	Z Measurement Data	ARRAY[0..19999] OF LREAL	Depends on data type.	The X-direction values of 2D shape data are stored. The unit is mm.

*1. Use the following expression to find the position of *sShape2D.ZValue[m]*.
Position[m] = UINT_TO_LREAL((m-1) * *sShape2D.Resolution*) / LREAL#1000

Function

The Shape2D_Height function block measures heights in a specified measurement range of the 2D shape data.



(a) The function block checks the input parameters when *Execute* changes to TRUE. If there is no error in the input data, *Busy* (Measuring) changes to TRUE. If there is an error, the function block execution is aborted and the error codes are output to *ErrorID* and *ErrorIDEx*. For details on the error codes, refer to *Troubleshooting* on page 246.

(b) The measurement range must be specified so that it contains at least one measurement data (Shape.Zvalue[X]).

If X does not meet the following condition, an error occurs.

RangeLow ≤((*Shape2D.Resolution* * (X-1)) / LREAL#1000) ≤ *RangeHigh*

X can take any value between 1 and *Shape2D.DataCount*.

(c) The function block outputs *CalcRsltMean* that contains the mean value of heights in the measurement range of the specified 2D shape data.

(d) The function block outputs *MaxHeight* and *MaxHeightPos* that contain the maximum height in the measurement range of the specified 2D shape data and the X position at which the maximum height was detected. In the same way, the function block outputs *MinHeight* and *MinHeightPos* that contain the minimum height and the X position at which the minimum height was detected.

(e) If *MaxHeight* exceeds Threshold (High), FALSE is output to *ChkRsltMax*. In the following cases, TRUE is output to *ChkRsltMax*. If *MinHeight* is smaller than Threshold (Low), FALSE is output to *ChkRsltMin*. In the following cases, TRUE is output to *ChkRsltMin*.

(f) This function block may cause a Task Period Exceeded error because it calculates the feature amount using a maximum of 20,000 2D shape data points.

To avoid this, you need to set the number of processed data points *(NumProcData)* per period.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, take into account the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

(g) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

(h) If an error occurs during the function block execution, *Error* changes to TRUE and the function block execution is aborted. For details on the error codes, refer to *Troubleshooting* on page 246.

Additional Information

The time required to execute this function block varies depending on the measurement range, resolution of the 2D shape data, or other factors that determine the amount of processed data.

Re-execution of Function Blocks

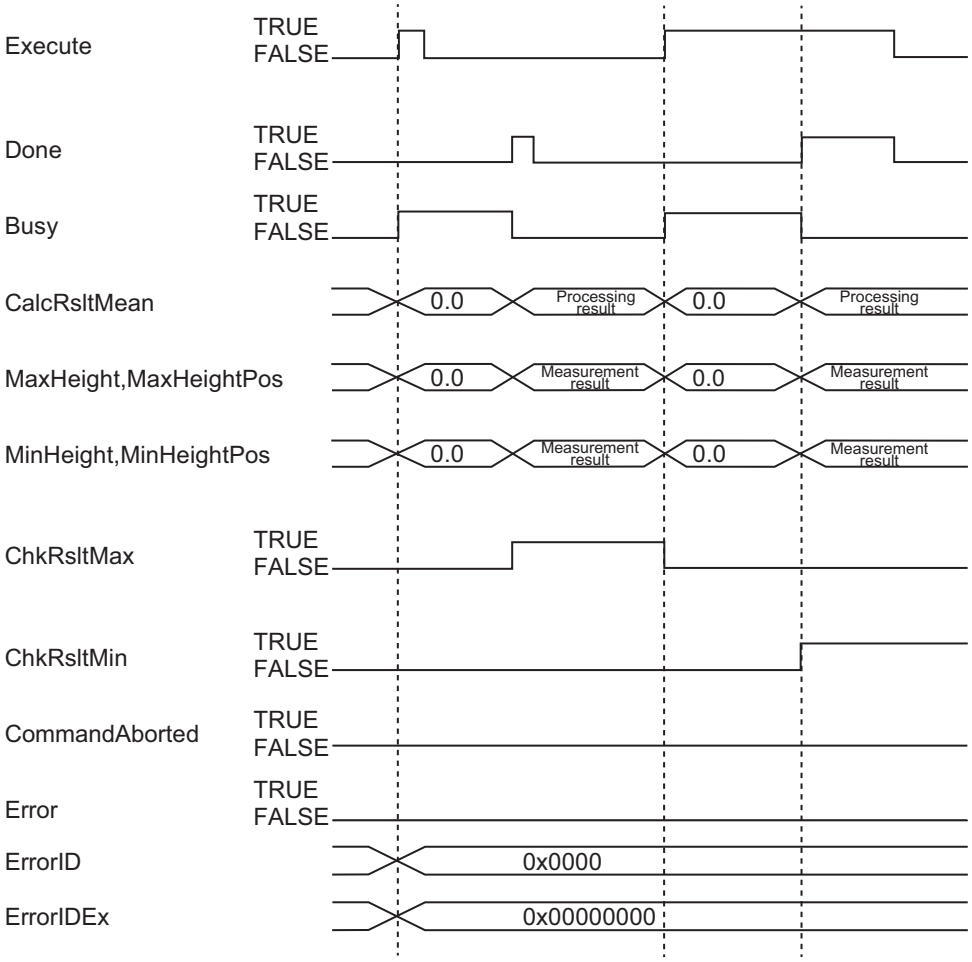
If *Execute* is changed to TRUE again while execution is in progress (*Busy* = TRUE), the execution is aborted and the processing is performed from the beginning.

Timing Charts

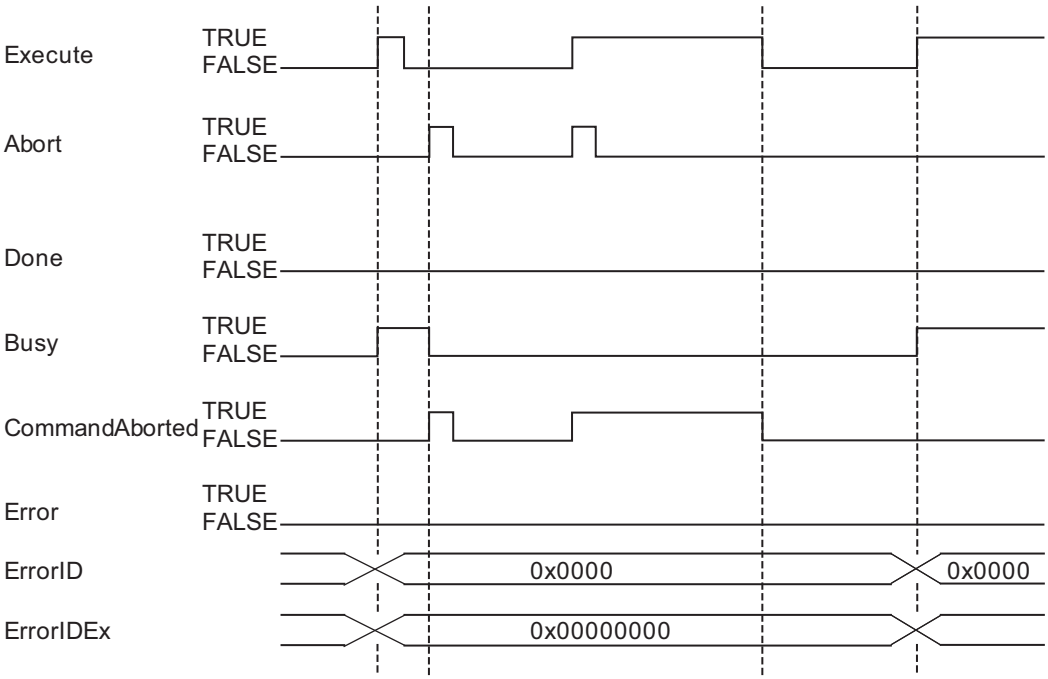
The timing charts are shown below.

● Timing Chart in a Normal State

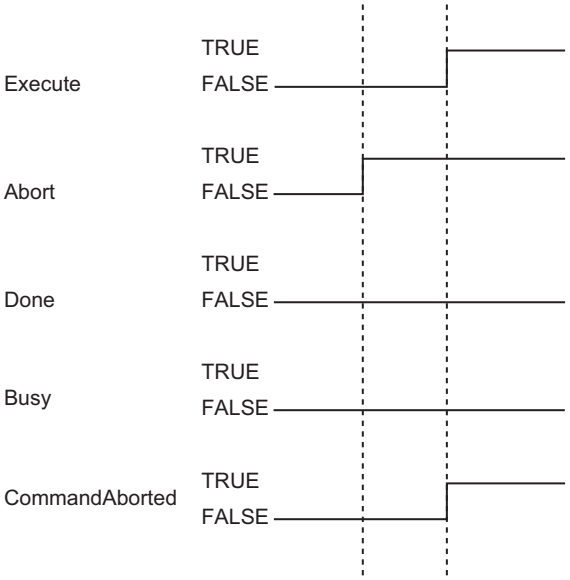
The values output to *MaxHeight*, *MaxHeightPos*, *MinHeight*, *MinHeightPos*, *ChkRsltMax*, and *ChkRsltMin* are retained until the next execution.



● Timing Chart When the Function Block Is Aborted

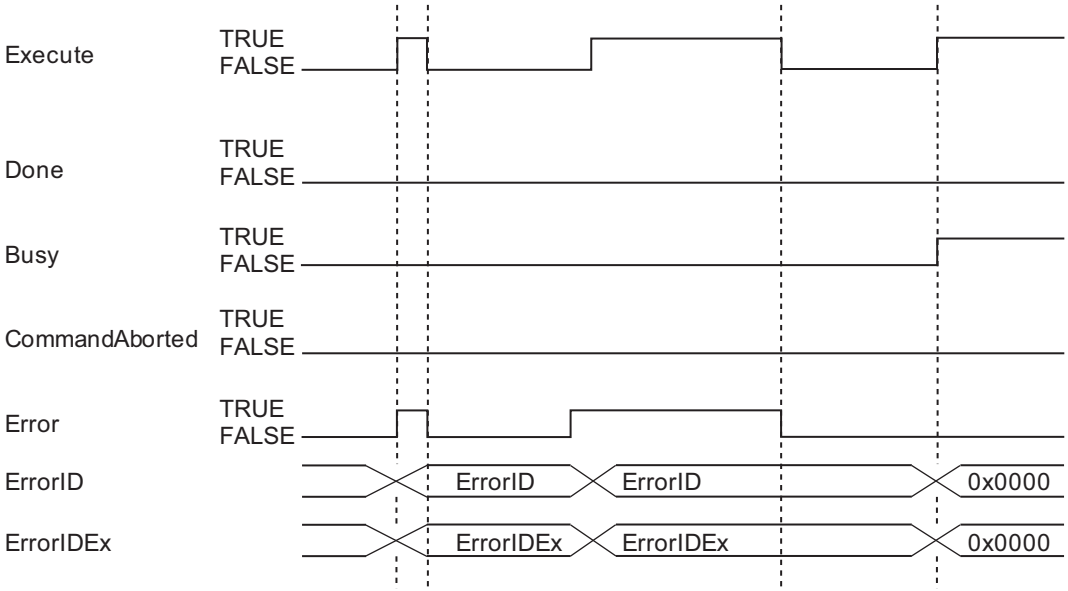


If *Execute* changes to TRUE while *Abort* is TRUE, *CommandAborted* (Interruption Completion) changes to TRUE without performing the processing.



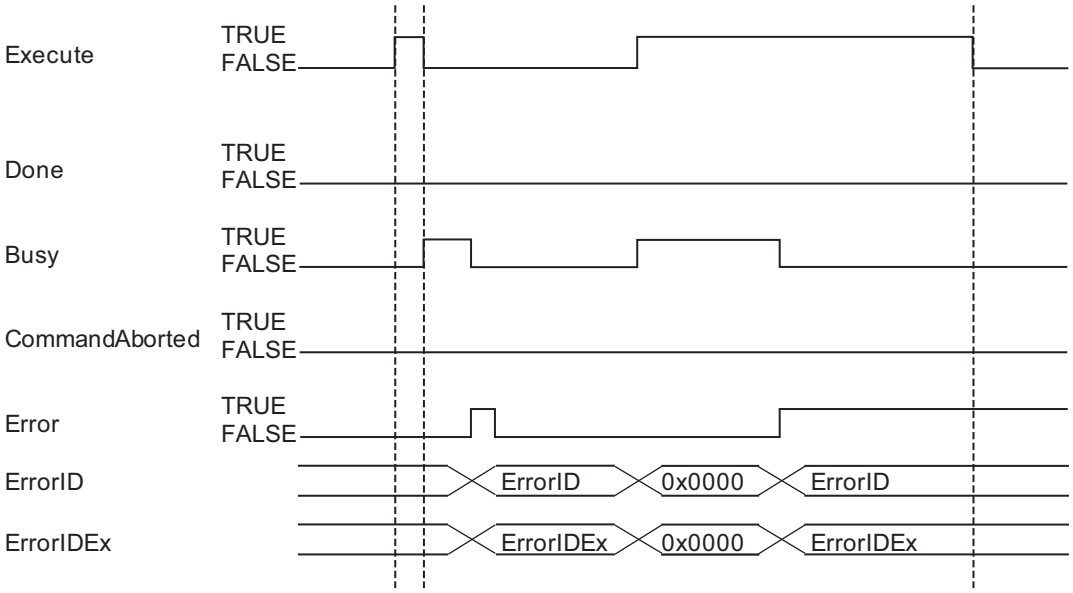
● Timing Chart When an Error Occurs (Startup Error)

If a setting value error is detected for the threshold or measurement range when the function block is executed, a startup error occurs. *Error* changes to TRUE and the values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code). When *Execute* changes to FALSE, *Error* changes to FALSE accordingly. However, *ErrorID* and *ErrorIDEx* retain the values until the next function block is executed.



● Timing Chart When an Error Occurs (Error during Function Block Execution)

If an error is detected for the 2D shape data during function block execution, *Busy* changes to FALSE, *Error* changes to TRUE, and the values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code). When *Execute* changes to FALSE, *Error* changes to FALSE accordingly. However, *ErrorID* and *ErrorIDEx* retain the values until the next function block is executed.



Precautions for Correct Use

Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3CD0	0x0000 0001	The value of Measurement Range Upper Limit is not valid.	Set a valid real number.
	0x0000 0002	The value of Measurement Range Lower Limit is not valid.	Set a valid real number.
	0x0000 0003	A value specified for the measurement range is not valid.	Specify values that can meet the following condition: Measurement Range Upper Limit ≥ Measurement Range Lower Limit.
	0x0000 0004	The value of Threshold (High) is not valid.	Set a valid real number.
	0x0000 0005	The value of Threshold (Low) is not valid.	Set a valid real number.
	0x0000 0006	Threshold (High) is smaller than Threshold (Low).	Specify values so that Threshold (High) is equal to or greater than Threshold (Low).
	0x0000 0007	There is no 2D shape data in the measurement range.	Set the measurement range or 2D shape data again.
	0x0000 0008	The number of data points *(DataCount)* of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.
	0x0000 0009	The resolution (*Resolution*) of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.

Sample Programming

Program Description

This program calculates the maximum and minimum height differences in the specified measurement range of the master and target 2D shape data. The program also determines whether the target measurement result is within the specified threshold range or not.

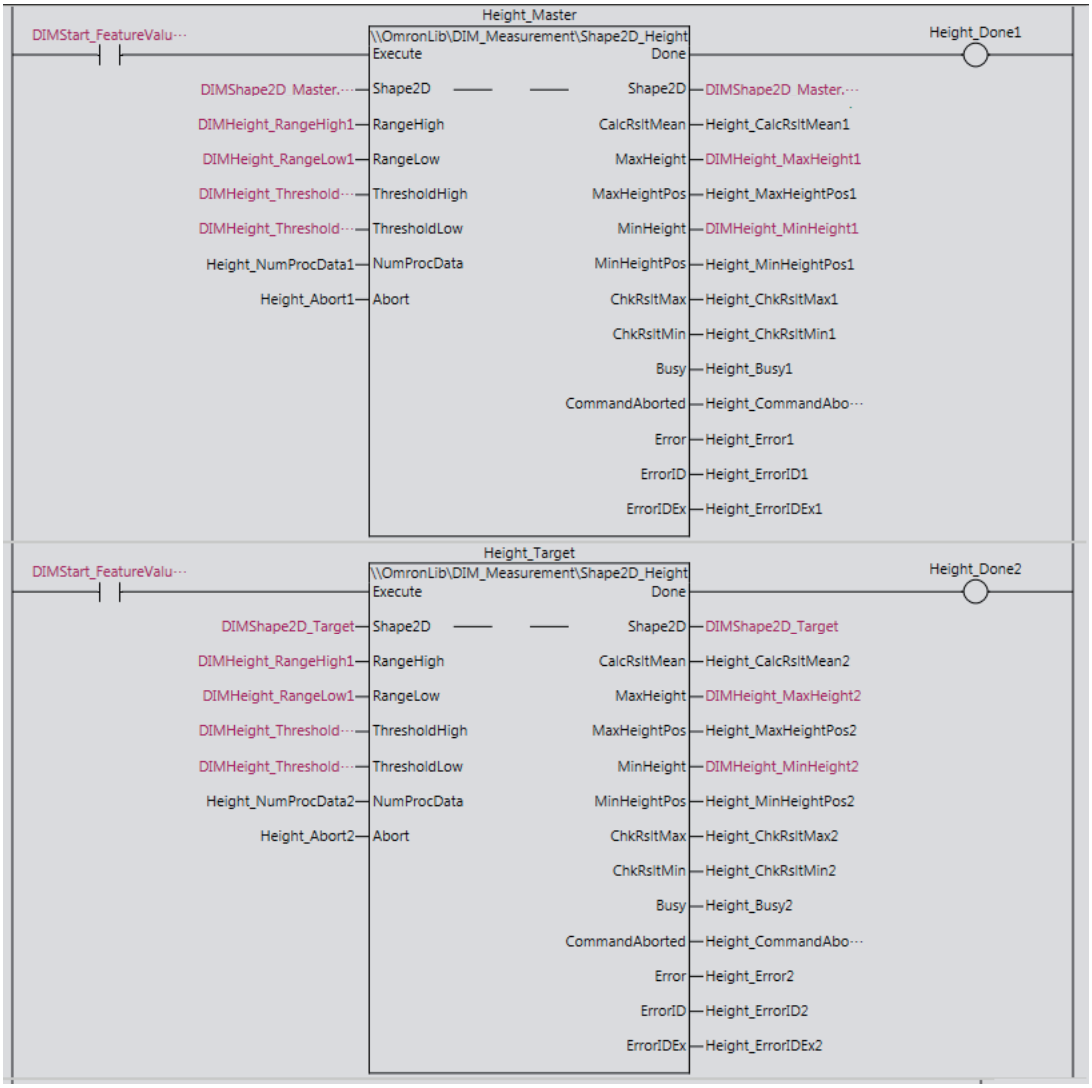
Main Variables

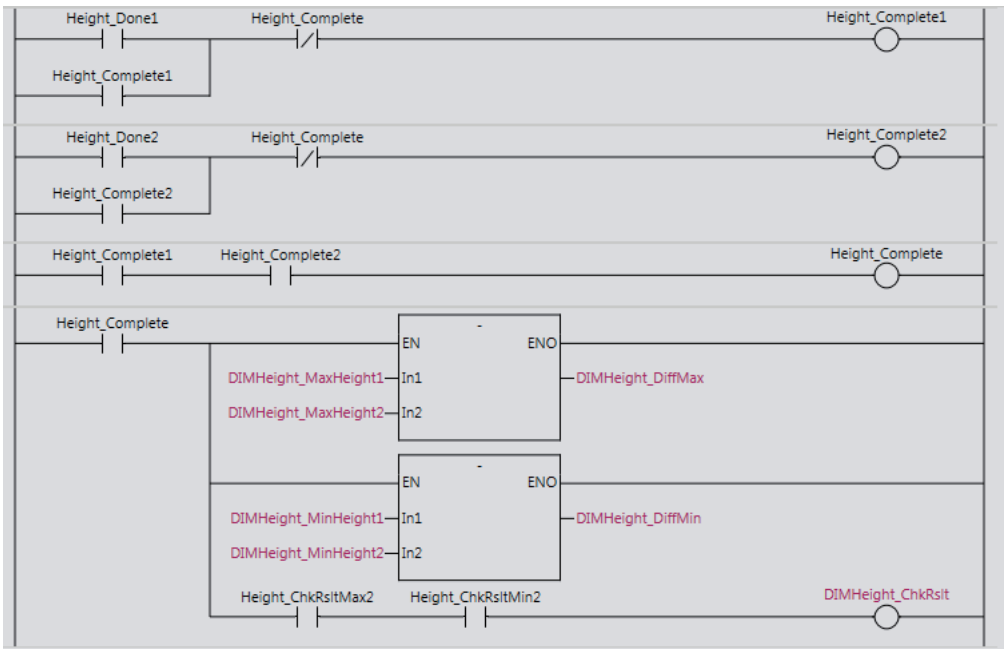
Name	Meaning	Data type	Default	Description
Height_Master	2D Shape Height Measurement FB	OmronLib\DIM_Measurement\Shape2D_Height	---	Instance of the function block that measures the height of the master 2D shape data.
DIMStart_FeatureValueHeight	Execution Trigger	BOOL	---	2D shape height measurement is started when this variable changes to TRUE. Assign this variable to the *Execute* input variable of Shape2D_Height.
DIMShape2D_Master	Master 2D Shape Data	OmronLib\DIM_Measurement\sShape2D	---	Set the master 2D shape data. Assign this variable to the *Shape2D* input variable of Shape2D_Height.
DIMHeight_RangeHigh1	Measurement Range Upper Limit	LREAL	---	Specify the upper limit of the X-axis measurement range that is used for height measurement. The unit is mm. Assign this variable to the *RangeHigh* input variable of Shape2D_Height.
DIMHeight_RangeLow1	Measurement Range Lower Limit	LREAL	---	Specify the lower limit of the X-axis measurement range that is used for height measurement. The unit is mm. Assign this variable to the *RangeLow* input variable of Shape2D_Height.
DIMHeight_ThresholdHigh1	Threshold (High)	LREAL	---	Set the threshold range. The unit is mm. Assign this variable to the *ThresholdHigh* input variable of Shape2D_Height.
DIMHeight_ThresholdLow1	Threshold (Low)	LREAL	---	Set the threshold range. The unit is mm. Assign this variable to the *ThresholdLow* input variable of Shape2D_Height.
Height_NumProcData1	Number of Processed Data Points	UINT	---	Set the number of data points that are processed in a task period during segmented processing. Assign this variable to the *NumProcData* input variable of Shape2D_Height.
Height_Abort1	Abort	BOOL	---	Aborts the processing. Assign this variable to the *Abort* input variable of Shape2D_Height.
Height_Done1	Done	BOOL	---	TRUE when the function block execution is completed. Assign this variable to the *Done* output variable of Shape2D_Height.
Height_CalcRsltMean1	Measurement Result Mean Value	LREAL	---	Outputs the mean value of heights in the measurement range. The unit is mm. Assign this variable to the *CalcRsltMean* output variable of Shape2D_Height.

Name	Meaning	Data type	Default	Description
Height_MaxHeight1	Maximum Height	LREAL	---	Outputs the maximum height within the measurement range of the master 2D shape data. The unit is mm. Assign this variable to the *MaxHeight* output variable of Shape2D_Height.
Height_MaxHeight Pos1	Maximum Height Position	LREAL	---	Outputs the position of the maximum height within the measurement range of the master 2D shape data. The unit is mm. Assign this variable to the *MaxHeightPos* output variable of Shape2D_Height.
Height_MinHeight1	Minimum Height	LREAL	---	Outputs the minimum height within the measurement range of the master 2D shape data. The unit is mm. Assign this variable to the *MinHeight* output variable of Shape2D_Height.
Height_MinHeight Pos1	Minimum Height Position	LREAL	---	Outputs the position of the minimum height within the measurement range of the master 2D shape data. The unit is mm. Assign this variable to the *MinHeightPos* output variable of Shape2D_Height.
Height_Target	2D Shape Height Measurement FB	OmronLib\ DIM_Measurement\ Shape2D_ Height	---	Instance of the function block that measures the height of the target 2D shape data.
DIMShape2D_Target	Target 2D Shape Data	OmronLib\ DIM_Measurement\ sShape2D	---	Set the target 2D shape data. Assign this variable to the *Shape2D* input variable of Shape2D_Height.
DIMHeight_Max Height2	Maximum Height	LREAL	---	Outputs the maximum height within the measurement range of the target 2D shape data. The unit is mm. Assign this variable to the *MaxHeight* output variable of Shape2D_Height.
DIMHeight_Max HeightPos2	Maximum Height Position	LREAL	---	Outputs the position of the maximum height within the measurement range of the target 2D shape data. The unit is mm. Assign this variable to the *MaxHeightPos* output variable of Shape2D_Height.
DIMHeight_Min Height2	Minimum Height	LREAL	---	Outputs the minimum height within the measurement range of the target 2D shape data. The unit is mm. Assign this variable to the *MinHeight* output variable of Shape2D_Height.
DIMHeight_Min HeightPos2	Minimum Height Position	LREAL	---	Outputs the position of the minimum height within the measurement range of the target 2D shape data. The unit is mm. Assign this variable to the *MinHeightPos* output variable of Shape2D_Height.
Height_ChkRsltMax2	Maximum Value Judgment Result	BOOL	---	TRUE when Maximum Height is equal to or less than Threshold (High). Assign this variable to the *ChkRsltMax* output variable of Shape2D_Height.
Height_ChkRsltMin2	Minimum Value Judgment Result	BOOL	---	TRUE when Minimum Height is equal to or greater than Threshold (Low). Assign this variable to the *ChkRsltMin* output variable of Shape2D_Height.

Name	Meaning	Data type	Default	Description
DIMHeight_DiffMax	Maximum Height Difference	LREAL	---	Outputs the maximum height differences that are measured from the master and target 2D shape data.
DIMHeight_DiffMin	Minimum Height Difference	LREAL	---	Outputs the minimum height differences that are measured from the master and target 2D shape data.

Ladder Diagram





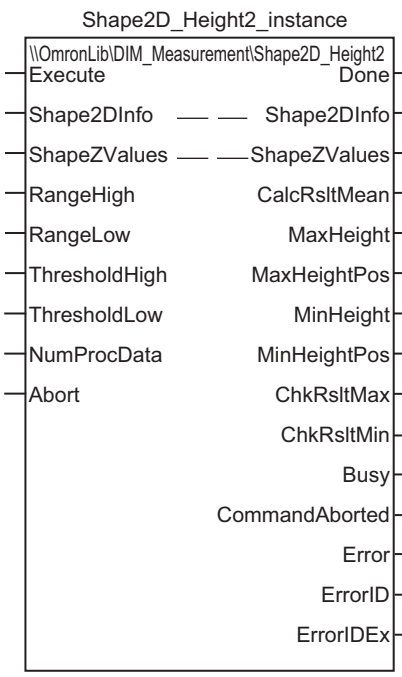
ST

```
//Master height calculation
Height_Master(
   Execute:=DIMStart_FeatureValueHeight,
   Shape2D:=DIMShape2D_Master.Shape2D,
   RangeHigh:=DIMHeight_RangeHigh1,
   RangeLow:=DIMHeight_RangeLow1,
   ThresholdHigh:=DIMHeight_ThresholdHigh1,
   ThresholdLow:=DIMHeight_ThresholdLow1,
   NumProcData:=Height_NumProcData1,
   Abort:=Height_Abort1,
   Done=>Height_Done1,
   CalcRsltMean=>Height_CalcRsltMean1,
   MaxHeight=>DIMHeight_MaxHeight1,
   MaxHeightPos=>Height_MaxHeightPos1,
   MinHeight=>DIMHeight_MinHeight1,
   MinHeightPos=>Height_MinHeightPos1,
   ChkRsltMax=>Height_ChkRsltMax1,
   ChkRsltMin=>Height_ChkRsltMin1,
   Busy=>Height_Busy1,
   CommandAborted=>Height_CommandAborted1,
   Error=>Height_Error1,
   ErrorID=>Height_ErrorID1,
   ErrorIDEx=>Height_ErrorIDEx1);

//Target height calculation
Height_Target(
   Execute:=DIMStart_FeatureValueHeight,
   Shape2D:=DIMShape2D_Target,
   RangeHigh:=DIMHeight_RangeHigh1,
   RangeLow:=DIMHeight_RangeLow1,
   ThresholdHigh:=DIMHeight_ThresholdHigh1,
   ThresholdLow:=DIMHeight_ThresholdLow1,
   NumProcData:=Height_NumProcData2,
   Abort:=Height_Abort2,
   Done=>Height_Done2,
   CalcRsltMean=>Height_CalcRsltMean2,
   MaxHeight=>DIMHeight_MaxHeight2,
   MaxHeightPos=>Height_MaxHeightPos2,
   MinHeight=>DIMHeight_MinHeight2,
   MinHeightPos=>Height_MinHeightPos2,
   ChkRsltMax=>Height_ChkRsltMax2,
   ChkRsltMin=>Height_ChkRsltMin2,
   Busy=>Height_Busy2,
   CommandAborted=>Height_CommandAborted2,
   Error=>Height_Error2,
   ErrorID=>Height_ErrorID2,
   ErrorIDEx=>Height_ErrorIDEx2);

//Master height calculation held
IF Height_Done1 = TRUE THEN
   Height_Complete1 := TRUE;
ELSE;
END_IF;

//Target height calculation held
IF Height_Done2 = TRUE THEN
   Height_Complete2 := TRUE;
ELSE;
END_IF;

//Height calculation completion processing
```

```
IF Height_Complete1 = TRUE & Height_Complete2 = TRUE THEN
   //Master/target difference calculation
   DIMHeight_DiffMax := DIMHeight_MaxHeight1-DIMHeight_MaxHeight2;
   DIMHeight_DiffMin := DIMHeight_MinHeight1-DIMHeight_MinHeight2;

   //Threshold judgment result flag processing
    IF Height_ChkRsltMax2 = TRUE & Height_ChkRsltMin2 = TRUE THEN
      DIMHeight_ChkRslt :=TRUE;
   ELSE
      DIMHeight_ChkRslt:=FALSE;
   END_IF;

   //Initialize height calculation completion flag
   Height_Complete1 := FALSE;
   Height_Complete2 := FALSE;
ELSE;
END_IF;
```

Shape2D_Height2

The Shape2D_Height2 function block measures heights in a specified measurement range of the 2D shape data for a variable-length array.

Function block and function name	Name	FB/ FUN	Graphic expression	ST expression
Shape2D_Height2	2D Shape Height Measure-ment 2	FB		Shape2D_Height2_instance(Execute, Shape2DInfo, ShapeZValues, RangeHigh, RangeLow, ThresholdHigh, ThresholdLow, NumProcData, Abort, Done, CalcRsltMean, MaxHeight, MaxHeightPos, MinHeight, MinHeightPos, ChkRsltMax, ChkRsltMin, Busy, CommandAborted, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V2_0.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00230
Source code	Not published

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
RangeHigh*1	Measurement Range Upper Limit	Specify the upper limit of the X-axis measurement range that is used for height measurement. The unit is mm.	LREAL	Negative number, positive number, or 0	0
RangeLow*1	Measurement Range Lower Limit	Specify the lower limit of the X-axis measurement range that is used for height measurement. The unit is mm.	LREAL	Negative number, positive number, or 0	0
ThresholdHigh*1	Threshold (High)	These thresholds are used to determine whether a measurement result is within the range. The unit is mm. Specify values always as: Threshold (High) ≥ Threshold (Low).	LREAL	Negative number, positive number, or 0	0
ThresholdLow*1	Threshold (Low)		LREAL	Negative number, positive number, or 0	0
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a period during segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
CalcRsltMean	Measurement Result Mean Value	Outputs the mean value of heights in the measurement range. The unit is mm.	LREAL	Depends on data type.	—
MaxHeight	Maximum Height	Outputs the maximum height in the measurement range. The unit is mm.	LREAL	Depends on data type.	—
MaxHeightPos	Maximum Height Position	Outputs the position of the maximum height in the measurement range. The unit is mm.	LREAL	Depends on data type.	—
MinHeight	Minimum Height	Outputs the minimum height in the measurement range. The unit is mm.	LREAL	Depends on data type.	—
MinHeightPos	Minimum Height Position	Outputs the position of the minimum height in the measurement range. The unit is mm.	LREAL	Depends on data type.	—
ChkRsltMax	Maximum Value Judgment Result	TRUE when Maximum Height is equal to or less than Threshold (High).	BOOL	TRUE or FALSE	—
ChkRsltMin	Minimum Value Judgment Result	TRUE when Minimum Height is equal to or greater than Threshold (Low).	BOOL	TRUE or FALSE	—
Busy	Measuring	TRUE when the measurement is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 261 for details.

In-Out Variables

Variable	Name	Description	Data type	Valid range
Shape2DInfo	2D Shape Data Information	Specify the 2D shape data information. Refer to *LineMeasure_Cartesian2* on page 151 for 2D shape data creation and correction.	sShape2DInfo	—
ShapeZValues[]*1	2D Shape Data Array	Specify an array where the 2D shape data is stored.	ARRAY[*] OF LREAL	—

*1. The maximum number of array elements is 20,000. The first number of array element should be 0.

Structure

The members of the structure sShape2DInfo are shown below.

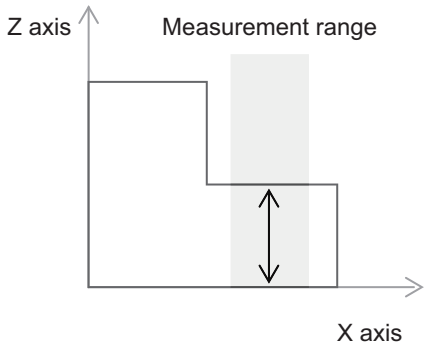
Member	Name	Data type	Valid range	Description
DataCount	Number of Data Points	UINT	0 to 20,000	The number of data points are stored.
Resolution	Resolution	UINT	Positive number	The resolution in X direction.*1 The unit is μm.

*1. Use the following expression to find the position of *ShapeZValues[m]*.

Position[m] = UINT_TO_LREAL((m-1) * *Shape2DInfo.Resolution*) / LREAL#1000

Function

The Shape2D_Height2 function block measures heights in a specified measurement range of the 2D shape data for a variable-length array.



(a) The function block checks the input parameters when *Execute* changes to TRUE. If there is no error in the input data, *Busy* (Measuring) changes to TRUE. If there is an error, the function block execution is aborted and the error codes are output to *ErrorID* and *ErrorIDEx*. For details on the error codes, refer to *Troubleshooting* on page 261.

(b) The measurement range must be specified so that it contains at least one measurement data (Shape.Zvalue[X]).

If X does not meet the following condition, an error occurs.

RangeLow ≤ ((*Shape2DInfo.Resolution* * (X-1)) / LREAL#1000) ≤ *RangeHigh*

X can take any value between 1 and *Shape2DInfo*.*DataCount*.

(c) The function block outputs *CalcRsltMean* that contains the mean value of heights in the measurement range of the specified 2D shape data.

(d) The function block outputs *MaxHeight* and *MaxHeightPos* that contain the maximum height in the measurement range of the specified 2D shape data and the X position at which the maximum height was detected. In the same way, the function block outputs *MinHeight* and *MinHeightPos* that contain the minimum height and the X position at which the minimum height was detected.

(e) If *MaxHeight* exceeds Threshold (High), FALSE is output to *ChkRsltMax*. In the following cases, TRUE is output to *ChkRsltMax*. If *MinHeight* is smaller than Threshold (Low), FALSE is output to *ChkRsltMin*. In the following cases, TRUE is output to *ChkRsltMin*.

(f) This function block may cause a Task Period Exceeded error because it calculates the feature amount using a maximum of 20,000 2D shape data points.

To avoid this, you need to set the number of processed data points *NumProcData* per period.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, take into account the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

(g) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

(h) If an error occurs during the function block execution, *Error* changes to TRUE and the function block execution is aborted. For details on the error codes, refer to *Troubleshooting* on page 261.

Additional Information

The time required to execute this function block varies depending on the measurement range, resolution of the 2D shape data, or other factors that determine the amount of processed data.

Re-execution of Function Blocks

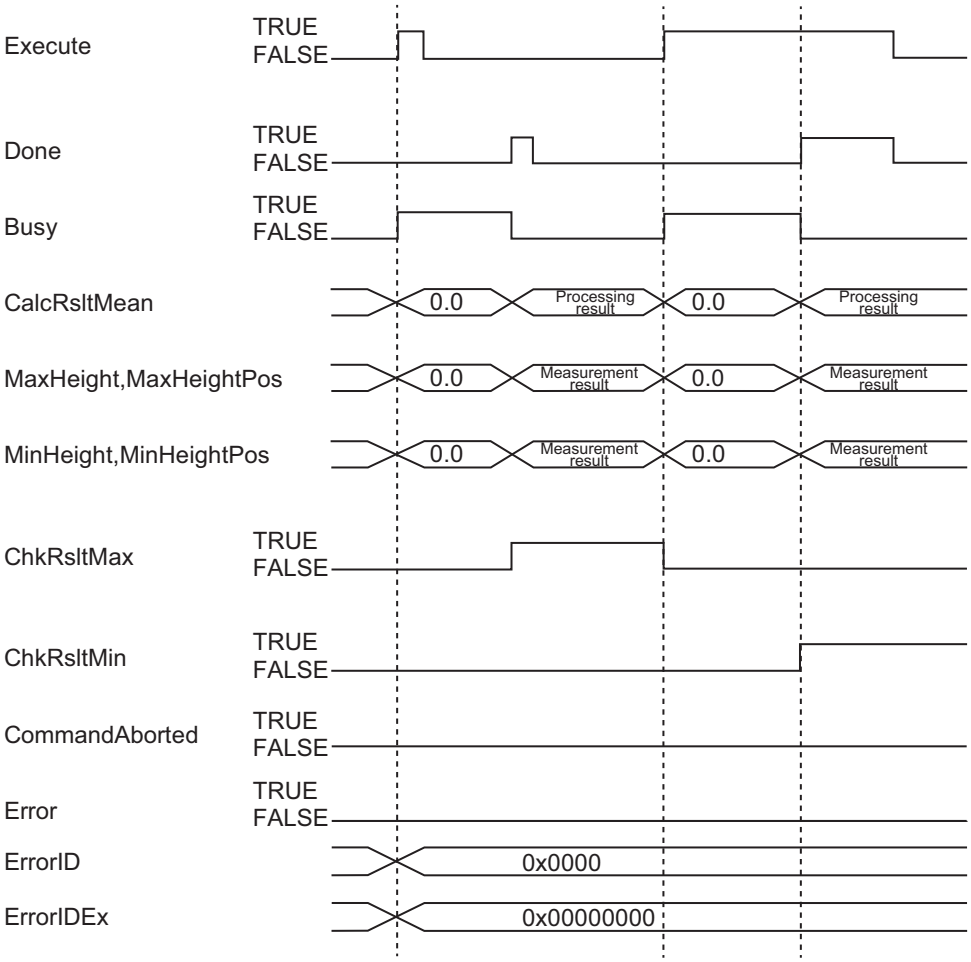
If *Execute* is changed to TRUE again while execution is in progress (*Busy* = TRUE), the execution is aborted and the processing is performed from the beginning.

Timing Charts

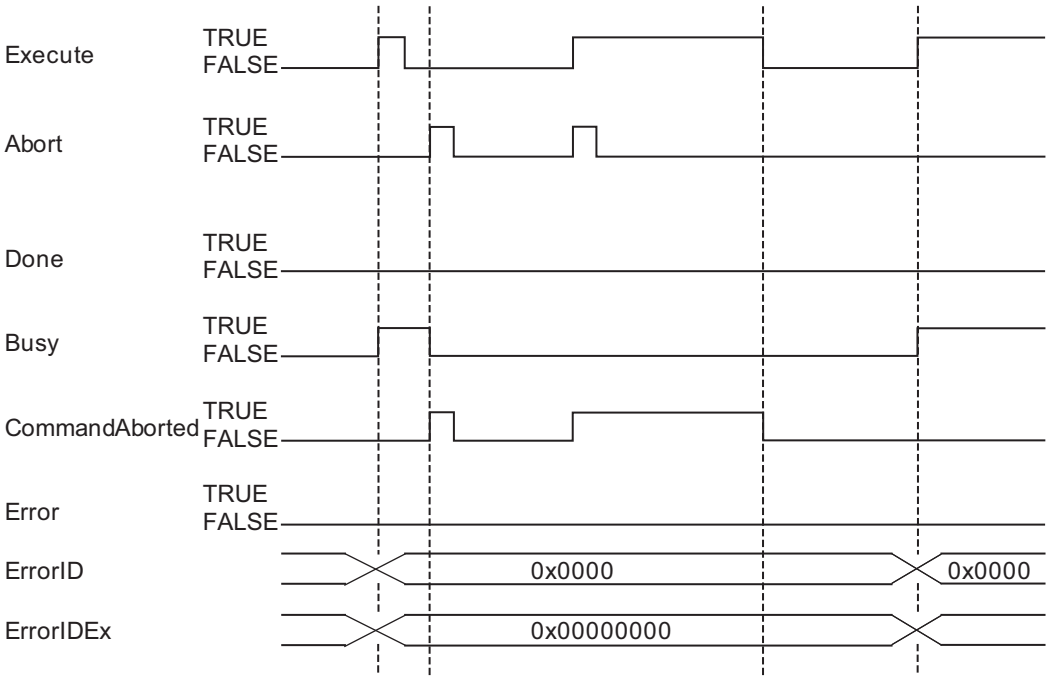
The timing charts are shown below.

● Timing Chart in a Normal State

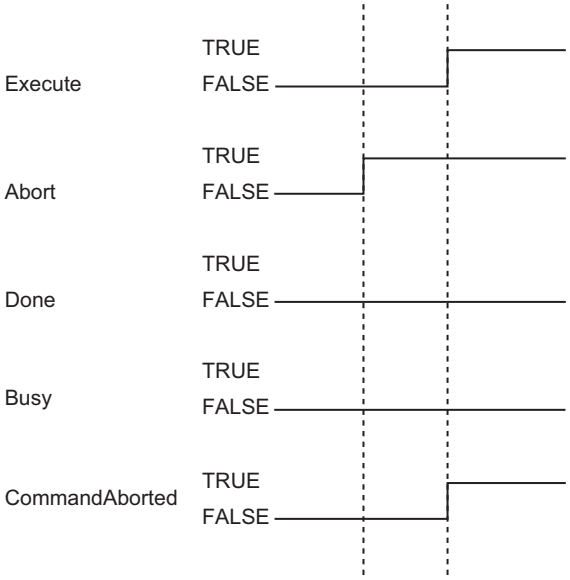
The values output to *MaxHeigh*t, *MaxHeightPos*, *MinHeight*, *MinHeightPos*, *ChkRsltMax*, and *ChkRsltMin* are retained until the next execution of the function block.



● Timing Chart When the Function Block Is Aborted

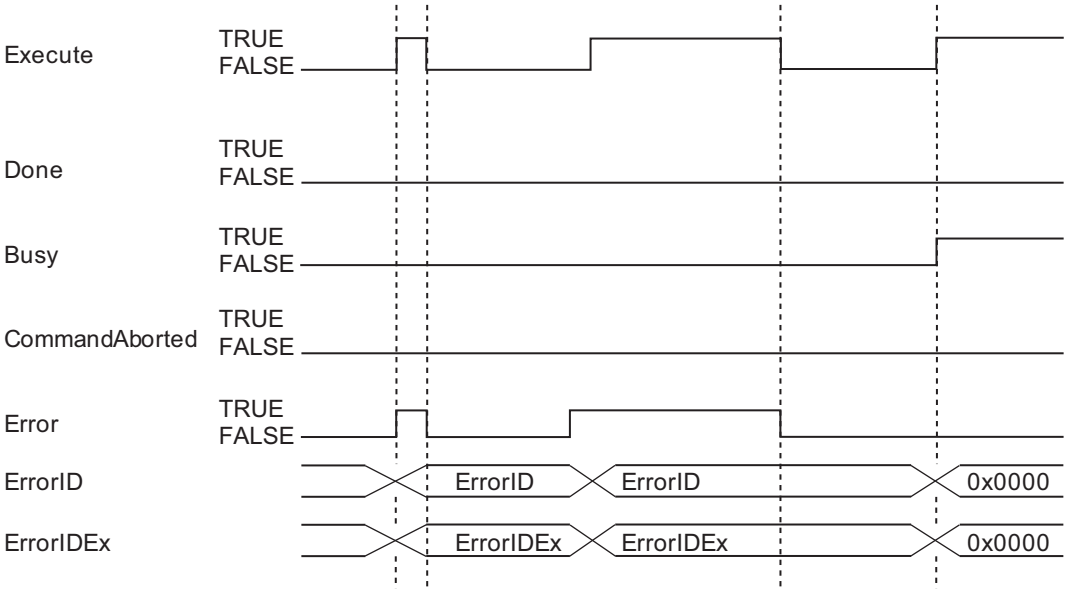


If *Execute* changes to TRUE while *Abort* is TRUE, *CommandAborted* (Interruption Completion) changes to TRUE without performing the processing.



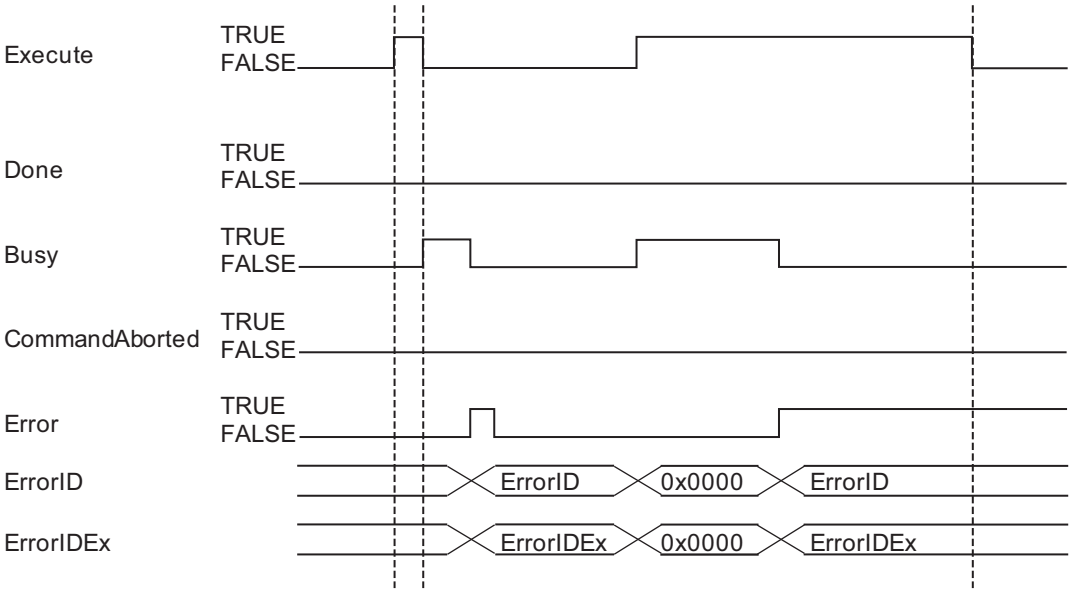
● Timing Chart When an Error Occurs (Startup Error)

If a setting value error is detected for the threshold or measurement range when the function block is executed, a startup error occurs. *Error* changes to TRUE and the values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code). When *Execute* changes to FALSE, *Erro*r changes to FALSE accordingly. However, *ErrorID* and *ErrorIDEx* retain the values until the next function block is executed.



● Timing Chart When an Error Occurs (Error during Function Block Execution)

If an error is detected for the 2D shape data during function block execution, *Bus*y changes to FALSE, *Error* changes to TRUE, and the values are output to *ErrorID* (Error Code) and *ErrorIDEx* (Expansion Error Code). When *Execute* changes to FALSE, *Erro*r changes to FALSE accordingly. However, *ErrorID* and *ErrorIDEx* retain the values until the next function block is executed.



Precautions for Correct Use

Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x0000	0x0000 0000	Normal end	—
0x3CD0	0x0000 0001	The value of Measurement Range Upper Limit is not valid.	Set a valid real number.
	0x0000 0002	The value of Measurement Range Lower Limit is not valid.	Set a valid real number.
	0x0000 0003	A value specified for the measurement range is not valid.	Specify values that can meet the following condition: Measurement Range Upper Limit ≥ Measurement Range Lower Limit.
	0x0000 0004	The value of Threshold (High) is not valid.	Set a valid real number.
	0x0000 0005	The value of Threshold (Low) is not valid.	Set a valid real number.
	0x0000 0006	Threshold (High) is smaller than Threshold (Low).	Specify values so that Threshold (High) is equal to or greater than Threshold (Low).
	0x0000 0007	There is no 2D shape data in the measurement range.	Set the measurement range or 2D shape data again.
	0x0000 0008	The number of data points (*DataCount*) of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.
	0x0000 0009	The resolution (*Resolution*) of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.
	0x0000 000A	The specified array for the 2D shape data array (*ShapeZValues*) is out of the valid range.	The first number of array elements should be 0. Specify a value that the number of array elements is 20,000 or less.

Sample Programming

Program Description

This program calculates the maximum and minimum height differences in the specified measurement range of the master and target 2D shape data. The program also determines whether the target measurement result is within the specified threshold range or not.

Main Variables

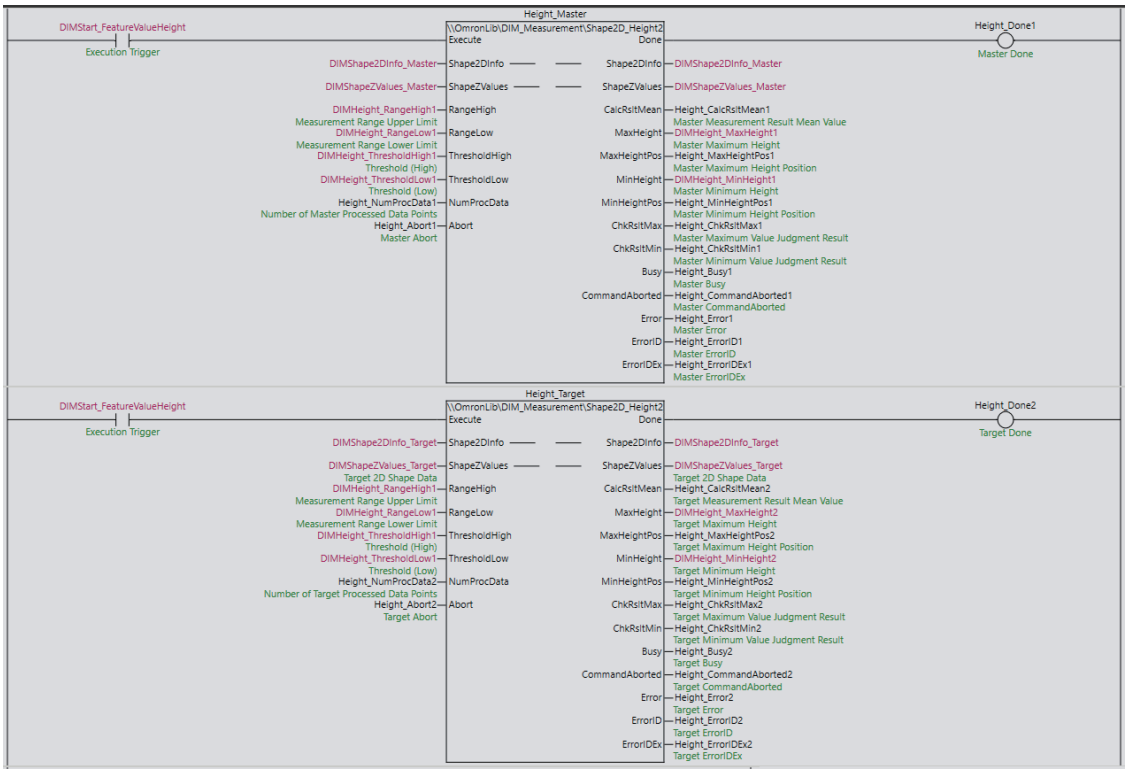
Name	Meaning	Data type	Default	Description
Height_Master	2D Shape Height Measurement FB	OmronLib\DIM_Measurement\Shape2D_Height2	—	Instance of the function block that measures the height of the master 2D shape data.
DIMStart_FeatureValueHeight	Execution Trigger	BOOL	—	2D shape height measurement is started when this variable changes to TRUE. Assign this variable to the *Execute* input variable of Shape2D_Height2.

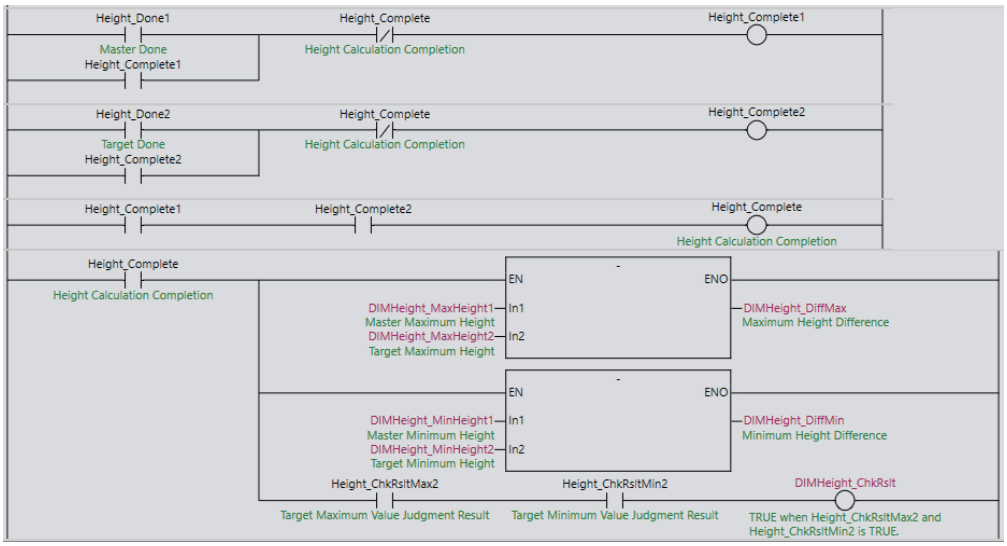
Name	Meaning	Data type	Default	Description
DIMShape2DInfo_ Master	Master 2D Shape Data Information	Omron-Lib\DIM_Measure-ment\sShape2DInfo	—	Set the master 2D shape data information. Assign this variable to the *Shape2DInfo* in-out variable of Shape2D_Height2.
DIMShapeZValues_-Master	Master 2D Shape Data Array	ARRAY[0..19999] OF LREAL	—	Set the master 2D shape data array. Assign this variable to the *ShapeZValues* in-out variable of Shape2D_Height2.
DIMHeight_Range High1	Measure-ment Range Upper Limit	LREAL	—	Specify the upper limit of the X-axis measurement range that is used for height measurement. The unit is mm. Assign this variable to the *RangeHigh* input variable of Shape2D_Height2.
DIMHeight_Range Low1	Measure-ment Range Lower Limit	LREAL	—	Specify the lower limit of the X-axis measurement range that is used for height measurement. The unit is mm. Assign this variable to the *RangeLow* input variable of Shape2D_Height2.
DIMHeight _ThresholdHigh1	Threshold (High)	LREAL	—	Set the threshold range. The unit is mm. Assign this variable to the *ThresholdHigh* input variable of Shape2D_Height2.
DIMHeight _ThresholdLow1	Threshold (Low)	LREAL	—	Set the threshold range. The unit is mm. Assign this variable to the *ThresholdLow* input variable of Shape2D_Height2.
Height_NumProc Data1	Number of Processed Data Points	UINT	—	Set the number of data points that are processed in a task period during segmented processing. Assign this variable to the *NumProcData* input variable of Shape2D_Height2.
Height_Abort1	Abort	BOOL	—	Aborts the processing. Assign this variable to the *Abort* input variable of Shape2D_Height2.
Height_Done1	Done	BOOL	—	TRUE when the function block execution is completed. Assign this variable to the *Done* output variable of Shape2D_Height2.
Height_CalcRslt Mean1	Measure-ment Result Mean Value	LREAL	—	Outputs the mean value of heights in the measurement range. The unit is mm. Assign this variable to the *CalcRsltMean* output variable of Shape2D_Height2.
Height_MaxHeight1	Maximum Height	LREAL	—	Outputs the maximum height within the measurement range of the master 2D shape data. The unit is mm. Assign this variable to the *MaxHeight* output variable of Shape2D_Height2.
Height_MaxHeight Pos1	Maximum Height Posi-tion	LREAL	—	Outputs the position of the maximum height within the measurement range of the master 2D shape data. The unit is mm. Assign this variable to the *MaxHeightPos* output variable of Shape2D_Height2.

Name	Meaning	Data type	Default	Description
Height_MinHeight1	Minimum Height	LREAL	—	Outputs the minimum height within the measurement range of the master 2D shape data. The unit is mm. Assign this variable to the *MinHeight* output variable of Shape2D_Height2.
Height_MinHeight Pos1	Minimum Height Position	LREAL	—	Outputs the position of the minimum height within the measurement range of the master 2D shape data. The unit is mm. Assign this variable to the *MinHeightPos* output variable of Shape2D_Height2.
Height_Target	2D Shape Height Measurement FB	OmronLib\DIM_Measurement\Shape2D_Height2	—	Instance of the function block that measures the height of the target 2D shape data.
DIMShape2DInfo_ Target	Target 2D Shape Data Information	OmronLib\DIM_Measurement\sShape2DInfo	—	Set the target 2D shape data information. Assign this variable to the *Shape2DInfo* in-out variable of Shape2D_Height2.
DIMShapeZValues_Target	Target 2D Shape Data Array	ARRAY[0..19999] OF LREAL	—	Set the target 2D shape data array. Assign this variable to the *ShapeZValues* in-out variable of Shape2D_Height2.
DIMHeight_Max Height2	Maximum Height	LREAL	—	Outputs the maximum height within the measurement range of the target 2D shape data. The unit is mm. Assign this variable to the *MaxHeight* output variable of Shape2D_Height2.
DIMHeight_Max HeightPos2	Maximum Height Position	LREAL	—	Outputs the position of the maximum height within the measurement range of the target 2D shape data. The unit is mm. Assign this variable to the *MaxHeightPos* output variable of Shape2D_Height2.
DIMHeight_Min Height2	Minimum Height	LREAL	—	Outputs the minimum height within the measurement range of the target 2D shape data. The unit is mm. Assign this variable to the *MinHeight* output variable of Shape2D_Height2.
DIMHeight_Min HeightPos2	Minimum Height Position	LREAL	—	Outputs the position of the minimum height within the measurement range of the target 2D shape data. The unit is mm. Assign this variable to the *MinHeightPos* output variable of Shape2D_Height2.
Height_ChkRsltMax2	Maximum Value Judgment Result	BOOL	—	TRUE when Maximum Height is equal to or less than Threshold (High). Assign this variable to the *ChkRsltMax* output variable of Shape2D_Height2.
Height_ChkRsltMin2	Minimum Value Judgment Result	BOOL	—	TRUE when Minimum Height is equal to or greater than Threshold (Low). Assign this variable to the *ChkRsltMin* output variable of Shape2D_Height2.
DIMHeight_DiffMax	Maximum Height Difference	LREAL	—	Outputs the maximum height differences that are measured for the master and target 2D shape data arrays.

Name	Meaning	Data type	Default	Description
DIMHeight_DiffMin	Minimum Height Difference	LREAL	—	Outputs the minimum height differences that are measured for the master and target 2D shape data arrays.

Ladder Diagram





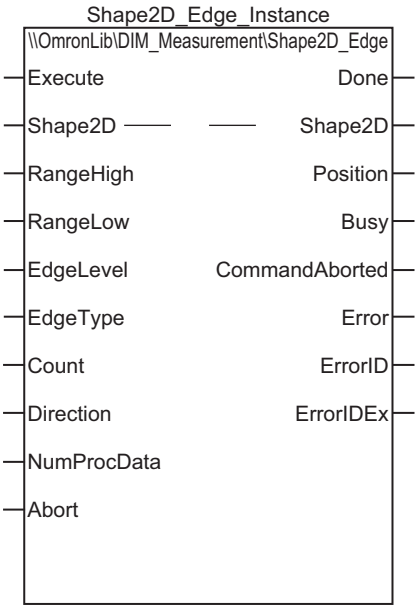
ST

```
//Master height calculation
Height_Master(
   Execute:=DIMStart_FeatureValueHeight,
   Shape2DInfo:=DIMShape2DInfo_Master,
   ShapeZValues:=DIMShapeZValues_Master,
   RangeHigh:=DIMHeight_RangeHigh1,
   RangeLow:=DIMHeight_RangeLow1,
   ThresholdHigh:=DIMHeight_ThresholdHigh1,
   ThresholdLow:=DIMHeight_ThresholdLow1,
   NumProcData:=Height_NumProcData1,
   Abort:=Height_Abort1,
   Done=>Height_Done1,
   CalcRsltMean=>Height_CalcRsltMean1,
   MaxHeight=>DIMHeight_MaxHeight1,
   MaxHeightPos=>Height_MaxHeightPos1,
   MinHeight=>DIMHeight_MinHeight1,
   MinHeightPos=>Height_MinHeightPos1,
   ChkRsltMax=>Height_ChkRsltMax1,
   ChkRsltMin=>Height_ChkRsltMin1,
   Busy=>Height_Busy1,
   CommandAborted=>Height_CommandAborted1,
   Error=>Height_Error1,
   ErrorID=>Height_ErrorID1,
   ErrorIDEx=>Height_ErrorIDEx1
   );

//Target height calculation
Height_Target(
   Execute:=DIMStart_FeatureValueHeight,
   Shape2DInfo:=DIMShape2DInfo_Target,
   ShapeZValues:=DIMShapeZValues_Target,
   RangeHigh:=DIMHeight_RangeHigh1,
   RangeLow:=DIMHeight_RangeLow1,
   ThresholdHigh:=DIMHeight_ThresholdHigh1,
   ThresholdLow:=DIMHeight_ThresholdLow1,
   NumProcData:=Height_NumProcData2,
   Abort:=Height_Abort2,
   Done=>Height_Done2,
   CalcRsltMean=>Height_CalcRsltMean2,
   MaxHeight=>DIMHeight_MaxHeight2,
   MaxHeightPos=>Height_MaxHeightPos2,
   MinHeight=>DIMHeight_MinHeight2,
   MinHeightPos=>Height_MinHeightPos2,
   ChkRsltMax=>Height_ChkRsltMax2,
   ChkRsltMin=>Height_ChkRsltMin2,
   Busy=>Height_Busy2,
   CommandAborted=>Height_CommandAborted2,
   Error=>Height_Error2,
   ErrorID=>Height_ErrorID2,
   ErrorIDEx=>Height_ErrorIDEx2
   );


//Master height calculation held
IF Height_Done1 = TRUE THEN
   Height_Complete1 := TRUE;
ELSE;
END_IF;

//Target height calculation held
IF Height_Done2 = TRUE THEN
```

```
   Height_Complete2 := TRUE;
ELSE;
END_IF;

//Height calculation completion processing
IF Height_Complete1 = TRUE & Height_Complete2 = TRUE THEN
   //Master/target difference calculation
   DIMHeight_DiffMax := DIMHeight_MaxHeight1-DIMHeight_MaxHeight2;
   DIMHeight_DiffMin := DIMHeight_MinHeight1-DIMHeight_MinHeight2;

   //Threshold judgment result flag processing
    IF Height_ChkRsltMax2 = TRUE & Height_ChkRsltMin2 = TRUE THEN
        DIMHeight_ChkRslt :=TRUE;
   ELSE
        DIMHeight_ChkRslt:=FALSE;
   END_IF;

   //Initialize height calculation completion flag
   Height_Complete1 := FALSE;
   Height_Complete2 := FALSE;
ELSE;
END_IF;
```

Shape2D_Edge

The Shape2D_Edge function block measures the position (X coordinate) at which a height in the specified measurement range of the 2D shape data passes the edge level.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
Shape2D_Edge	2D Shape Edge Position Measurement	FB		Shape2D_Edge_Instance (Execute, Shape2D, RangeHigh, RangeLow, EdgeLevel, EdgeType, Count, Direction, NumProcData, Abort, Done, Position, Busy, CommandAborted, Error, ErrorID, ErrorIDEx)

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V1_1.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00142
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Product name	Model numbers	Version
Device	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
RangeHigh*1	Measurement Range Upper Limit	Specify the upper limit of the X-axis measurement range that is used for edge detection. The unit is mm.	LREAL	Negative number, positive number, or 0	0
RangeLow*1	Measurement Range Lower Limit	Specify the lower limit of the X-axis measurement range that is used for edge detection. The unit is mm.	LREAL	Negative number, positive number, or 0	0
EdgeLevel*1	Edge Level	Set the edge level for height. The unit is mm.	LREAL	Negative number, positive number, or 0	0
EdgeType*1	Edge Type	TRUE: Rising edge detection FALSE: Falling edge detection	BOOL	TRUE or FALSE	FALSE
Count*1	Count	Specify what number detected edge after starting measurement is output as the measurement result.	UINT	Positive number	1
Direction*1	Measurement Direction	FALSE: Measurement starts from the lower limit to the upper limit of the measurement range. TRUE: Measurement starts from the upper limit to the lower limit of the measurement range.	BOOL	TRUE or FALSE	FALSE
NumProcData *1	Number of Processed Data Points	Specify the number of data points that are processed in a task period in the segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
Position	Position	Outputs the position (X coordinate) at which an edge is detected. The unit is mm.	LREAL	Depends on data type.	—
Busy	Measuring	TRUE when the measurement is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 273.

In-Out Variables

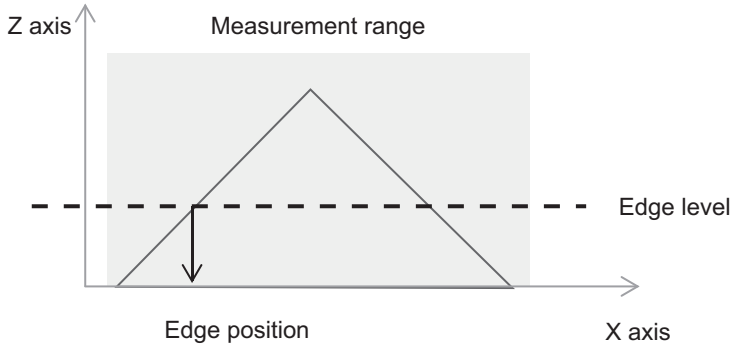
Variable	Name	Description	Data type	Valid range
Shape2D	2D Shape Data	Specify the 2D shape data. Refer to *LineMeasure_Cartesian* on page 108 for 2D shape data creation and correction.	sShape2D	—

Members of Structure sShape2D

Refer to *Members of Structure sShape2D* on page 241 of Shape2D_Height.

Function

The Shape2D_Edge function block measures the position (X coordinate) at which a height in the specified measurement range of the 2D shape data passes the edge level.



(a) The function block checks the input parameters when *Execute* changes to TRUE. If there is no error in the input data, *Busy* (Measuring) changes to TRUE. If there is an error, the function block execution is aborted and the error codes are output to *ErrorID* and *ErrorIDEx*. For details on the error codes, refer to *Troubleshooting* on page 273.

(b) The measurement range must be specified so that it contains at least one measurement data (Shape.Zvalue[X]).

If X does not meet the following condition, an error occurs.

RangeLow ≤ ((*Shape2D.Resolution* * (X-1)) / LREAL#1000) ≤ *RangeHigh*

X can take any value between 1 and *Shape2D.DataCount*.

(c) This function block measures the position at which a height (Z coordinate of the shape data) in the specified measurement range of the 2D shape data passes the edge level. The measured position is output to *Position*. The measurement result differs depending on the settings of Edge Type, Measurement Direction and Edge Count.

- Edge Type (*EdgeType*)

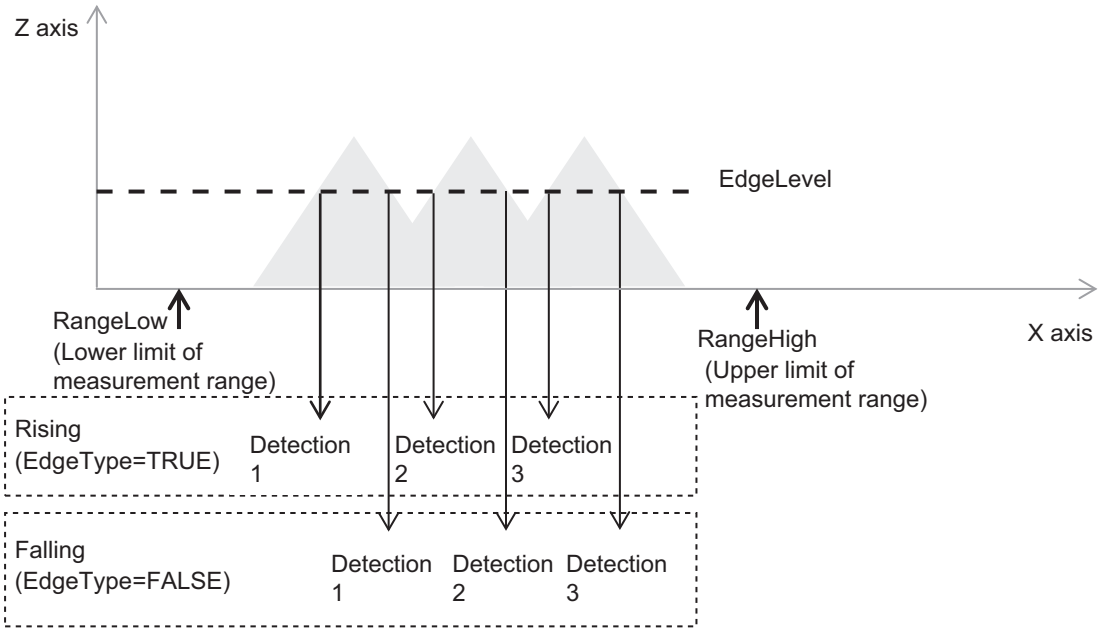
  Select in which direction (rising or falling) the height passes the edge level.
- Measurement Direction (*Direction*)

  Select Measurement Range Lower Limit or Measurement Range Upper Limit to start measurement.
- Edge Count (*Count*)

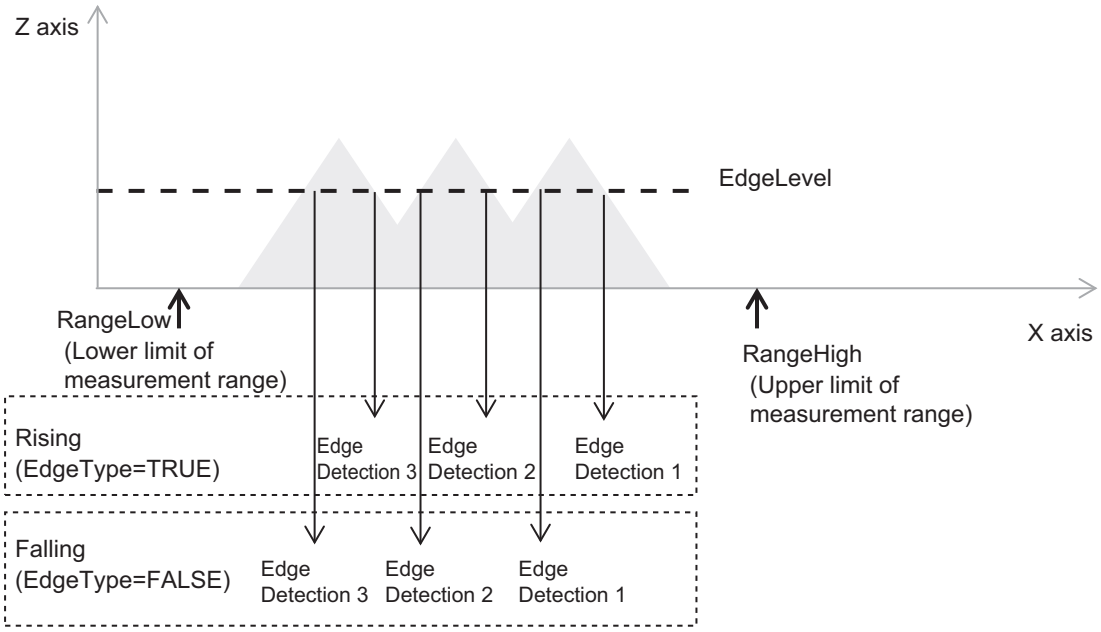
  Specify what number detected edge is output as the measurement result.

The following is an example.

Example 1. When measurement starts from the measurement range lower limit (*Direction* = FALSE), the edge detection positions are as follows depending on Edge Type and Edge Count.



Example 2. When measurement starts from the measurement range upper limit (*Direction* = TRUE), the edge detection positions are as follows depending on Edge Type and Edge Count.



(d) This function block may cause a Task Period Exceeded error because it calculates the feature amount using a maximum of 20,000 2D shape data points.

To avoid this, you need to set the number of processed data points *(NumProcData)* per period.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, take into account the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

(e) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

(f) If an error occurs during the function block execution, *Error* changes to TRUE and the function block execution is aborted. For details on the error codes, refer to *Troubleshooting* on page 273.

Additional Information

The time required to execute this function block varies depending on the measurement range, resolution of the 2D shape data, or other factors that determine the amount of processed data.

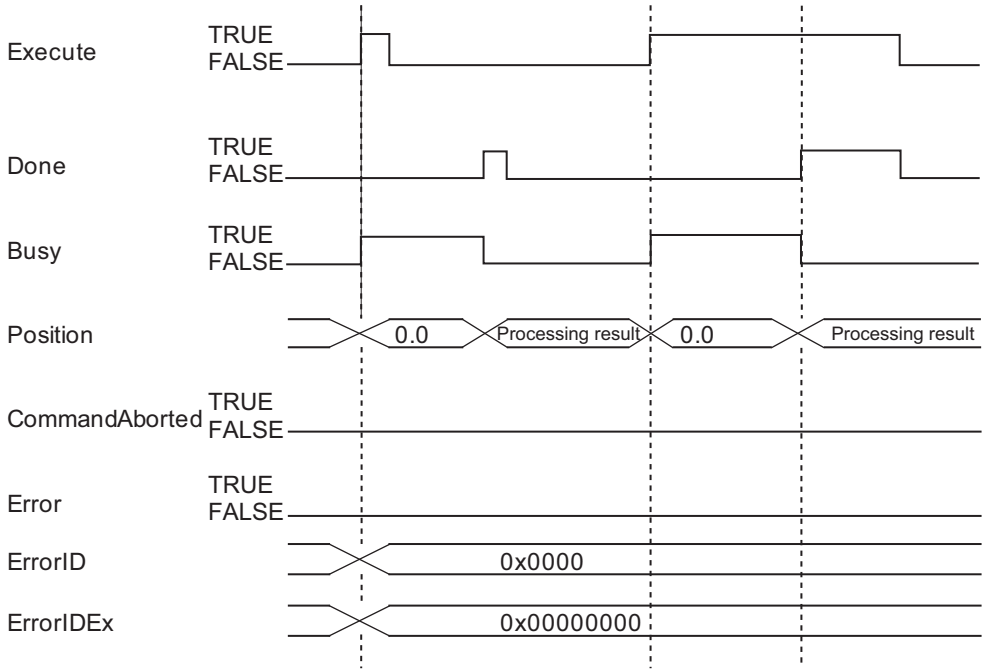
Re-execution of Function Blocks

If *Execute* is changed to TRUE again while execution is in progress (*Busy* = TRUE), the execution is aborted and the processing is performed from the beginning.

Timing Charts

The timing charts are shown below.

● Timing Chart in a Normal State



● Timing Chart When the Function Block Is Aborted

Refer to *Timing Chart When the Function Block Is Aborted* on page 244 of Shape2D_Height.

● Timing Chart When an Error Occurs

Refer to *Timing Chart When an Error Occurs (Startup Error)* on page 245 of Shape2D_Height.

Precautions for Correct Use

Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3CD1	0x0000 0001	The value of Measurement Range Upper Limit is not valid.	Set a valid real number.
	0x0000 0002	The value of Measurement Range Lower Limit is not valid.	Set a valid real number.
	0x0000 0003	A value specified for the measurement range is not valid.	Specify values that can meet the following condition: Measurement Range Upper Limit ≥ Measurement Range Lower Limit.
	0x0000 0004	The value of the edge level is not valid.	Set a valid real number.
	0x0000 0005	The value specified for the edge count is not valid.	Specify a positive number for the edge count.
	0x0000 0006	There is no 2D shape data in the measurement range.	Set the measurement range or 2D shape data again.
	0x0000 0007	The edge position was not detected.	Set appropriate values for the edge level and edge count.
	0x0000 0008	The number of data points *(DataCount)* of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.
	0x0000 0009	The resolution (*Resolution*) of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.

Sample Programming

Program Description

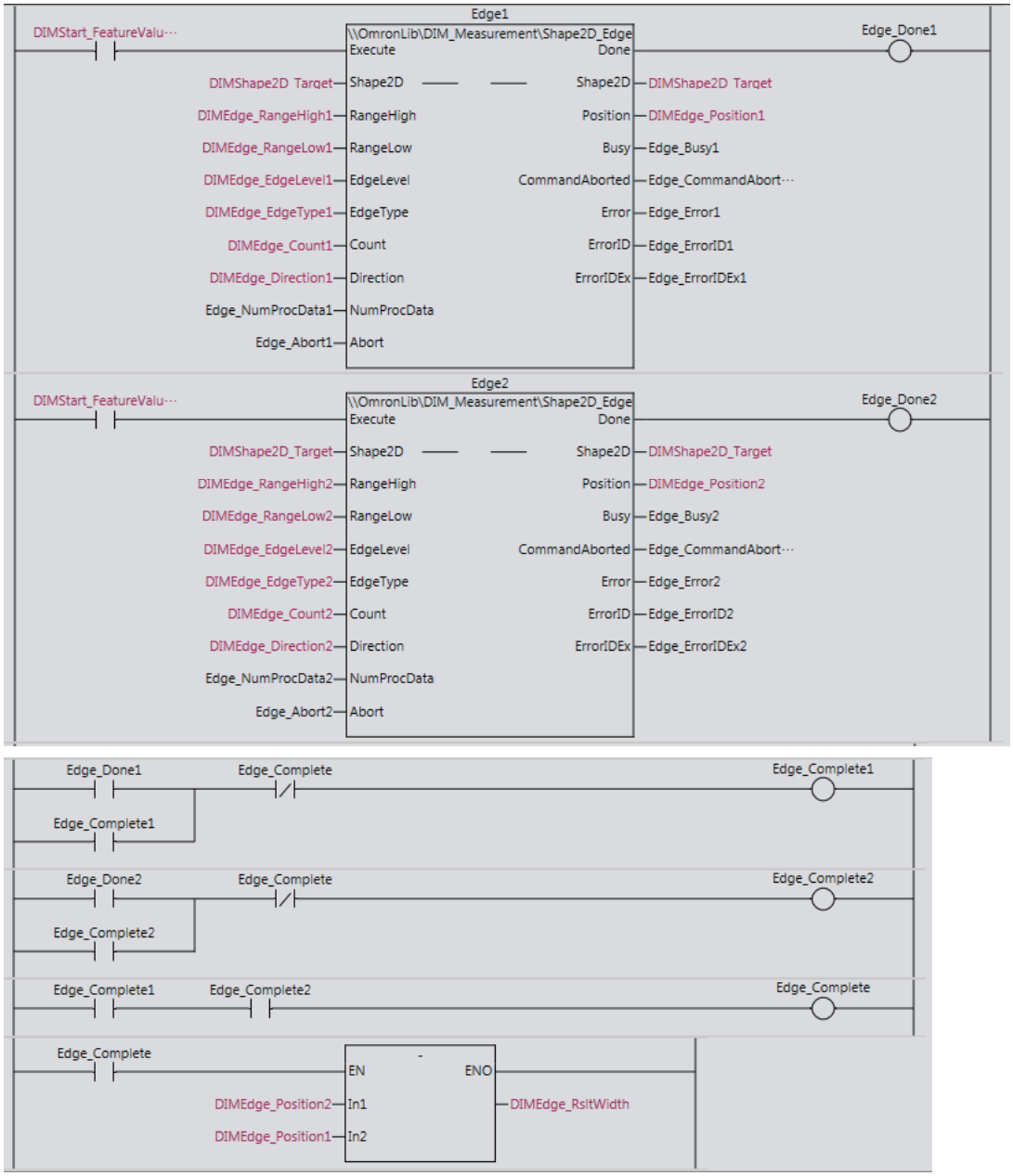
This program detects edges in a specified measurement range of the 2D shape data and calculates distances between edges. The program also determines whether the target measurement result is within the specified threshold range or not.

Main Variables

Name	Meaning	Data type	Default	Description
DIMStart_Feature ValueEdge	Execution Trigger	BOOL	---	2D shape height measurement is started when this variable changes to TRUE. Assign this variable to the *Execute* input variable of Shape2D_Edge.
Edge1(2)	2D Shape Edge Position Measurement FB	OmronLib\ DIM_Measurement\ Shape2D_ Edge	---	Instance of the function block that measures the first or the second edge position of the 2D shape data.
DIMShape2D_Target	2D Shape Data	OmronLib\ DIM_Measurement\ sShape2D	---	Set the 2D shape data. Assign this variable to the *Shape2D* input variable of Shape2D_Edge.
DIMEdge_Range High1(2)	Measurement Range Upper Limit	LREAL	---	Specify the upper limit of the X-axis measurement range that is used for the first or the second edge detection. The unit is mm. Assign this variable to the *RangeHigh* input variable of Shape2D_Edge.
DIMEdge_Range Low1(2)	Measurement Range Lower Limit	LREAL	---	Specify the lower limit of the X-axis measurement range that is used for the first or the second edge detection. The unit is mm. Assign this variable to the *RangeLow* input variable of Shape2D_Edge.
DIMEdge_Edge Level1(2)	Edge Level	LREAL	---	Set the edge level of the height for the first or the second edge detection. The unit is mm. Assign this variable to the *EdgeLevel* input variable of Shape2D_Edge.
DIMEdge_EdgeType1 (2)	Edge Type	BOOL	---	Set the trigger condition for the first or the second edge detection. Assign this variable to the *EdgeType* input variable of Shape2D_Edge.
DIMEdge_Count1(2)	Edge Count	UINT	---	Specify what number detected edge after starting measurement is output as the measurement result for the first or the second edge detection. Assign this variable to the *Count* input variable of Shape2D_Edge.
DIMEdge _Direction1(2)	Measurement Direction	BOOL	---	Set the measurement direction for the first or the second edge detection. Assign this variable to the *Direction* input variable of Shape2D_Edge.

Name	Meaning	Data type	Default	Description
DIMEdge _Position1(2)	Position	LREAL	---	Outputs the position (X coordinate) at which the first edge is detected. The unit is mm. Assign this variable to the *Position* output variable of Shape2D_Edge.
Edge_Complete	Edge Detection Completed	BOOL	---	Changes to TRUE when the function block completes the detection of two edges.
DIMEdge_RsltWidth	Width	LREAL	---	Outputs the difference between the first and second edge positions as the width.

Ladder Diagram

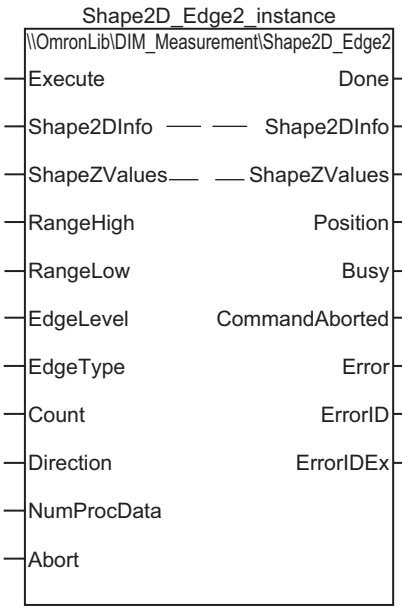


ST

```
//Edge 1 calculation
Edge1(
   Execute:=DIMStart_FeatureValueEdge,
   Shape2D:=DIMShape2D_Target,
   RangeHigh:=DIMEdge_RangeHigh1,
   RangeLow:=DIMEdge_RangeLow1,
   EdgeLevel:=DIMEdge_EdgeLevel1,
   EdgeType:=DIMEdge_EdgeType1,
   Count:=DIMEdge_Count1,
   Direction:=DIMEdge_Direction1,
   NumProcData:=Edge_NumProcData1,
   Abort:=Edge_Abort1,
   Done=>Edge_Done1,
   Position=>DIMEdge_Position1,
   Busy=>Edge_Busy1,
   CommandAborted=>Edge_CommandAborted1,
   Error=>Edge_Error1,
   ErrorID=>Edge_ErrorID1,
   ErrorIDEx=>Edge_ErrorIDEx1);

//Edge 2 calculation
Edge2(
   Execute:=DIMStart_FeatureValueEdge,
   Shape2D:=DIMShape2D_Target,
   RangeHigh:=DIMEdge_RangeHigh2,
   RangeLow:=DIMEdge_RangeLow2,
   EdgeLevel:=DIMEdge_EdgeLevel2,
   EdgeType:=DIMEdge_EdgeType2,
   Count:=DIMEdge_Count2,
   Direction:=DIMEdge_Direction2,
   NumProcData:=Edge_NumProcData2,
   Abort:=Edge_Abort2,
   Done=>Edge_Done2,
   Position=>DIMEdge_Position2,
   Busy=>Edge_Busy2,
   CommandAborted=>Edge_CommandAborted2,
   Error=>Edge_Error2,
   ErrorID=>Edge_ErrorID2,
   ErrorIDEx=>Edge_ErrorIDEx2);

//Edge 1 calculation completion held
IF Edge_Done1 = TRUE THEN
   Edge_Complete1 := TRUE;
ELSE;
END_IF;

//Edge 2 calculation completion held
IF Edge_Done2 = TRUE THEN
   Edge_Complete2 := TRUE;
ELSE;
END_IF;

IF Edge_Complete1 = TRUE & Edge_Complete2 = TRUE THEN
   //Result distance calculation
   DIMEdge_RsltWidth := DIMEdge_Position2 - DIMEdge_Position1;

   //Initialize edge calculation completion flag
   Edge_Complete1 := FALSE;
   Edge_Complete2 := FALSE;
ELSE;
END_IF;
```

Shape2D_Edge2

The Shape2D_Edge2 function block measures the position (X coordinate) at which a height in the specified measurement range of the 2D shape data for a variable-length array passes the edge level.

Function block and function name	Name	FB/ FUN	Graphic expression	ST expression
Shape2D_Edge2	2D Shape Edge Position Measurement 2	FB		Shape2D_Edge2_instance(Execute, Shape2DInfo, ShapeZValues, RangeHigh, RangeLow, EdgeLevel, EdgeType, Count, Direction, NumProcData, Abort, Done, Position, Busy, CommandAborted, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V2_0.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00231
Source code	Not published

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
RangeHigh*1	Measurement Range Upper Limit	Specify the upper limit of the X-axis measurement range that is used for edge detection. The unit is mm.	LREAL	Negative number, positive number, or 0	0
RangeLow*1	Measurement Range Lower Limit	Specify the lower limit of the X-axis measurement range that is used for edge detection. The unit is mm.	LREAL	Negative number, positive number, or 0	0
EdgeLevel*1	Edge level	Set the edge level for height. The unit is mm.	LREAL	Negative number, positive number, or 0	0
EdgeType*1	Edge Type	TRUE: Rising edge detection FALSE: Falling edge detection	BOOL	TRUE or FALSE	FALSE
Count*1	Count	Specify what number detected edge after starting measurement is output as the measurement result.	UINT	Positive number	1
Direction*1	Measurement Direction	FALSE: Measurement starts from the lower limit to the upper limit of the measurement range. TRUE: Measurement starts from the upper limit to the lower limit of the measurement range.	BOOL	TRUE or FALSE	FALSE
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a period during segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
Position	Position	Outputs the position (X coordinate) at which an edge is detected. The unit is mm.	LREAL	Depends on data type.	—
Busy	Measuring	TRUE when the measurement is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 284 for details.

In-Out Variables

Variable	Name	Description	Data type	Valid range
Shape2DInfo	2D Shape Data Information	Specify the 2D shape data information. Refer to *LineMeasure_Cartesian2* on page 151 for 2D shape data creation and correction.	sShape2DInfo	—
ShapeZValues[]*1	2D Shape Data Array	Specify an array where the 2D shape data is stored.	ARRAY[*] OF LREAL	—

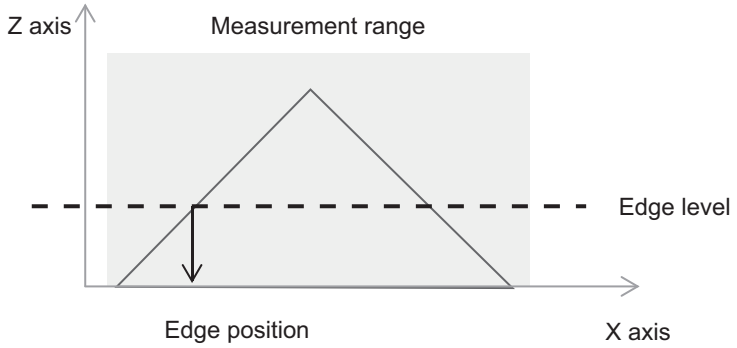
*1. The maximum number of array elements is 20,000. The first number of array element should be 0.

Members of Structure sShape2DInfo

Refer to *Structure* on page 256 of Shape2D_Height2.

Function

The Shape2D_Edge2 function block measures the position (X coordinate) at which a height in the specified measurement range of the 2D shape data for a variable-length array passes the edge level.



(a) The function block checks the input parameters when *Execute* changes to TRUE. If there is no error in the input data, *Busy* (Measuring) changes to TRUE. If there is an error, the function block execution is aborted and the error codes are output to *ErrorID* and *ErrorIDEx*. For details on the error codes, refer to *Troubleshooting* on page 284.

(b) The measurement range must be specified so that it contains at least one measurement data (*Shape.Zvalue[X]*).

If X does not meet the following condition, an error occurs.

RangeLow ≤ ((*Shape2DInfo*.*Resolution* * (X-1)) / LREAL#1000) ≤ *RangeHigh*

X can take any value between 1 and *Shape2DInfo*.*DataCount*.

(c) This function block measures the position at which a height (Z coordinate of the shape data) in the specified measurement range of the 2D shape data passes the edge level. The measured position is output to *Position*. The measurement result differs depending on the settings of Edge Type, Measurement Direction and Edge Count.

- Edge Type (*EdgeType*)

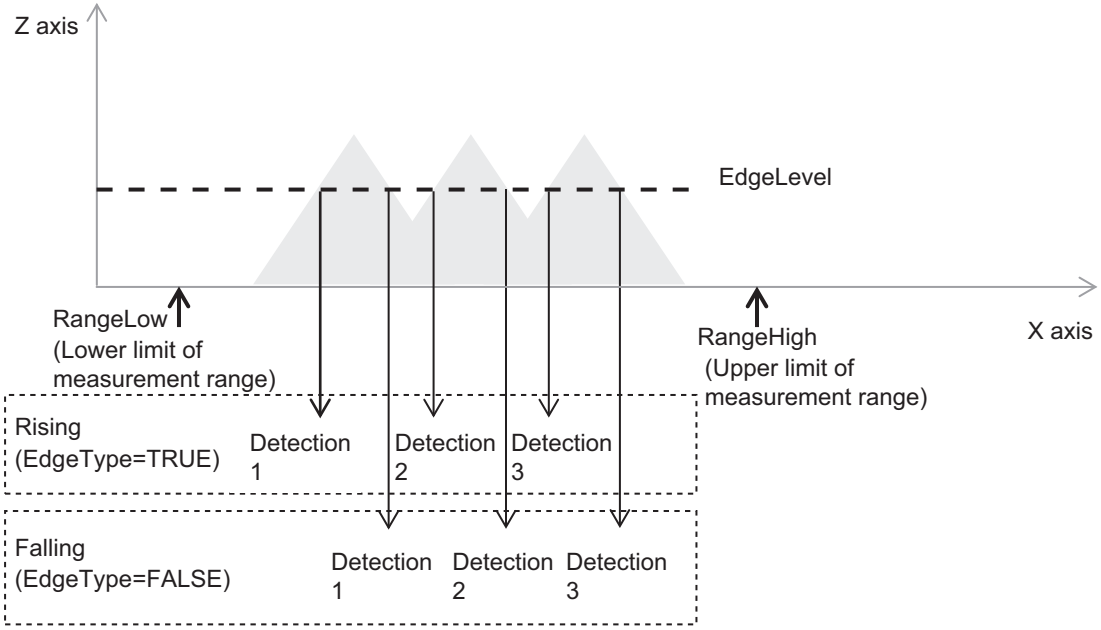
  Select in which direction (rising or falling) the height passes the edge level.
- Measurement Direction (*Direction*)

  Select Measurement Range Lower Limit or Measurement Range Upper Limit to start measurement.
- Edge Count (*Count*)

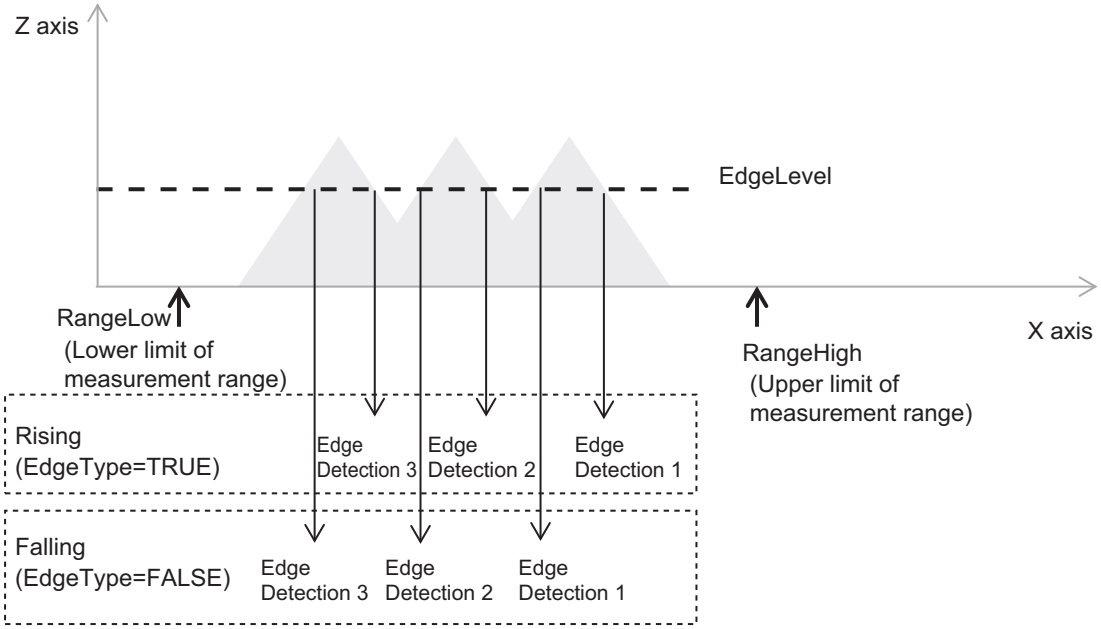
  Specify what number detected edge is output as the measurement result.

The following is an example.

Example 1. When measurement starts from the measurement range lower limit (*Direction* = FALSE), the edge detection positions will change as shown below depending on Edge Type and Edge Count.



Example 2. When measurement starts from the measurement range upper limit (*Direction* = TRUE), the edge detection positions will change as shown below depending on Edge Type and Edge Count.



(d) This function block may cause a Task Period Exceeded error because it calculates the feature amount using a maximum of 20,000 2D shape data points.

To avoid this, you need to set the number of processed data points *NumProcData* per period in the task that is executing.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, take into account the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

(e) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

(f) If an error occurs during the function block execution, *Error* changes to TRUE and the function block execution is aborted. For details on the error codes, refer to *Troubleshooting* on page 284.

Additional Information

The time required to execute this function block varies depending on the measurement range, resolution of the 2D shape data, or other factors that determine the amount of processed data.

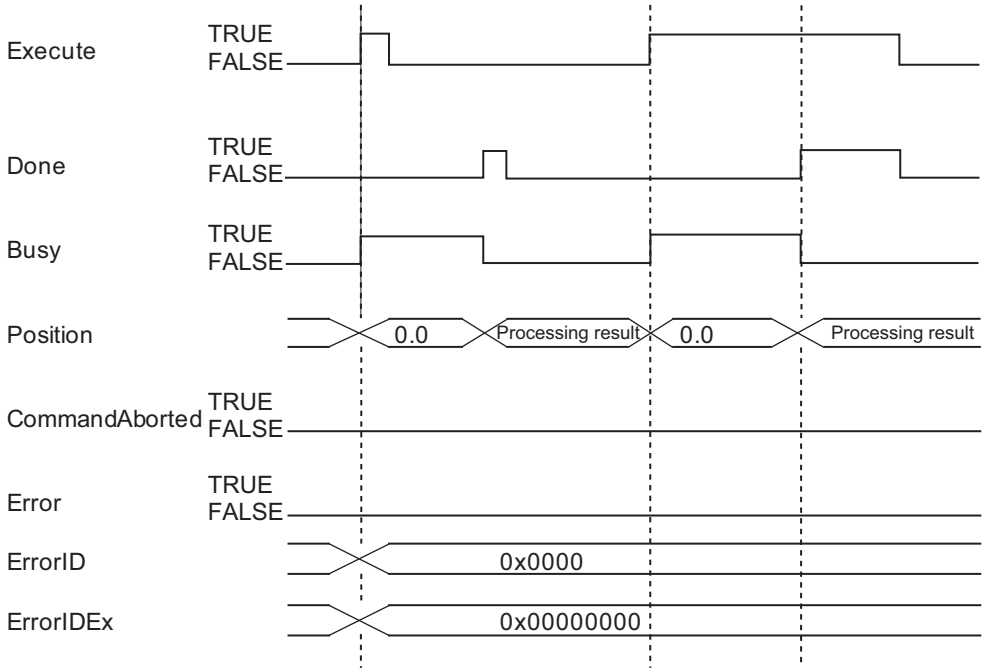
Re-execution of Function Blocks

If *Execute* is changed to TRUE again while execution is in progress (*Busy* = TRUE), the execution is aborted and the processing is performed from the beginning.

Timing Charts

The timing charts are shown below.

● Timing Chart in a Normal End State



● Timing Chart When the Function Block Is Aborted

Refer to *Timing Chart When the Function Block Is Aborted* on page 259 of Shape2D_Height2.

● Timing Chart When an Error Occurs (Startup Error)

Refer to *Timing Chart When an Error Occurs (Startup Error)* on page 260 of Shape2D_Height2.

● Timing Chart When an Error Occurs (Error during Function Block Execution)

Refer to *Timing Chart When an Error Occurs (Error during Function Block Execution)* on page 260 of Shape2D_Height2.

Precautions for Correct Use

Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x0000	0x00000000	Normal end	—
0x3CD1	0x00000001	The value of Measurement Range Upper Limit is not valid.	Set a valid real number.
	0x00000002	The value of Measurement Range Lower Limit is not valid.	Set a valid real number.
	0x00000003	A value specified for the measurement range is not valid.	Specify values that can meet the following condition: Measurement Range Upper Limit ≥ Measurement Range Lower Limit.
	0x00000004	The value of the edge level is not valid.	Set a valid real number.
	0x00000005	The value specified for the edge count is not valid.	Specify a positive number for the edge count.
	0x00000006	There is no 2D shape data in the measurement range.	Set the measurement range or 2D shape data again.
	0x00000007	The edge position was not detected.	Set appropriate values for the edge level and edge count.
	0x00000008	The number of data points (*DataCount*) of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.
	0x00000009	The resolution (*Resolution*) of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.
	0x0000000A	The specified array for the 2D shape data array (*ShapeZValues*) is out of the valid range.	The first number of array elements should be 0. Specify a value that the number of array elements is 20,000 or less.

Sample Programming

Program Description

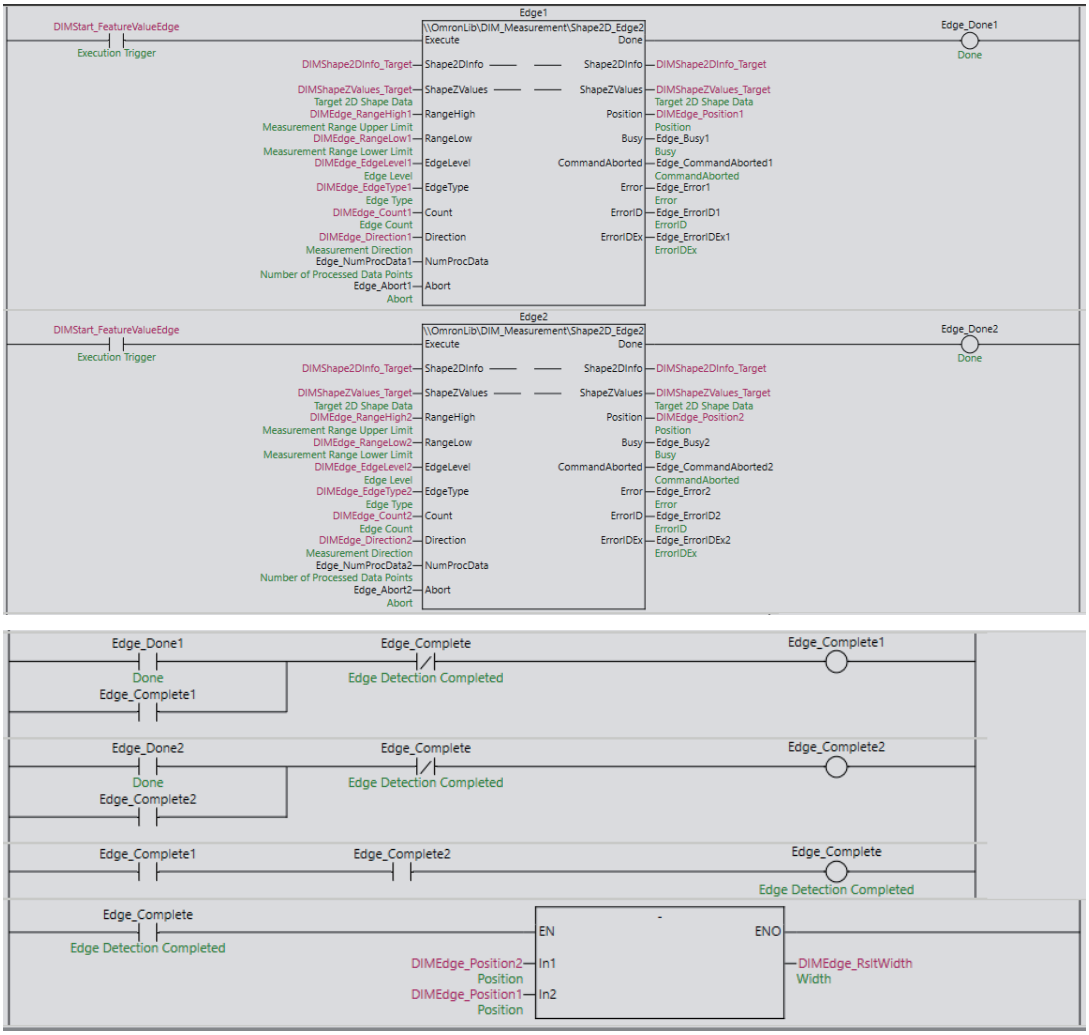
This program detects edges in a specified measurement range of the 2D shape data and calculates distances between edges. The program also determines whether the target measurement result is within the specified threshold range or not.

Main Variables

Name	Meaning	Data type	Default	Description
Edge1, Edge2	2D Shape Edge Position Measurement FB	OmronLib\DIM_Measurement\Shape2D_Edge2	—	An instance of the function block that measures the edge position of the 2D shape data.
DIMStart_FeatureValueEdge	Execution Trigger	BOOL	—	The 2D shape edge measurement is started when this variable changes to TRUE. Assign this variable to the *Execute* input variable of Shape2D_Edge2.

Name	Meaning	Data type	Default	Description
DIMShape2DInfo_Target	2D Shape Data Information	OmronLib\DIM_Measurement\sShape2DInfo	—	Set the 2D shape data information. Assign this variable to the *Shape2DInfo* in-out variable of Shape2D_Edge2.
DIMShapeZValues_Target	2D Shape Data Array	ARRAY[0..19999] OF LREAL	—	Set the 2D shape data array. Assign this variable to the *ShapeZValues* in-out variable of Shape2D_Edge2.
DIMEdge_RangeHigh1, DIMEdge_RangeHigh2	Measurement Range Upper Limit	LREAL	—	Specify the upper limit of the X-axis measurement range that is used for edge detection. The unit is mm. Assign this variable to the *RangeHigh* input variable of Shape2D_Edge2.
DIMEdge_RangeLow1, DIMEdge_RangeLow2	Measurement Range Lower Limit	LREAL	—	Specify the lower limit of the X-axis measurement range that is used for edge detection. The unit is mm. Assign this variable to the *RangeLow* input variable of Shape2D_Edge2.
DIMEdge_EdgeLevel1, DIMEdge_EdgeLevel2	Edge Level	LREAL	—	Set the edge level of the height for edge detection. The unit is mm. Assign this variable to the *EdgeLevel* input variable of Shape2D_Edge2.
DIMEdge_EdgeType1, DIMEdge_EdgeType2	Edge Type	BOOL	—	Set the trigger condition for edge detection. Assign this variable to the *EdgeType* input variable of Shape2D_Edge2.
DIMEdge_Count1, DIMEdge_Count2	Edge Count	UINT	—	Specify how many times of edge detection from starting the measurement is output as the measurement result for the edge detection. Assign this variable to the *Count* input variable of Shape2D_Edge2.
DIMEdge_Direction1, DIMEdge_Direction2	Measurement Direction	BOOL	—	Set the measurement direction for edge detection. Assign this variable to the *Direction* input variable of Shape2D_Edge2.
DIMEdge_Position1, DIMEdge_Position2	Position	LREAL	—	Outputs the position (X coordinate) where an edge is detected. The unit is mm. Assign this variable to the *Position* output variable of Shape2D_Edge2.
Edge_Complete	Edge Detection Completed	BOOL	—	Changes to TRUE when the function block completes the detection of two edges.
DIMEdge_RsltWidth	Width	LREAL	—	Outputs the difference between the first and second edge positions as the width.

Ladder Diagram



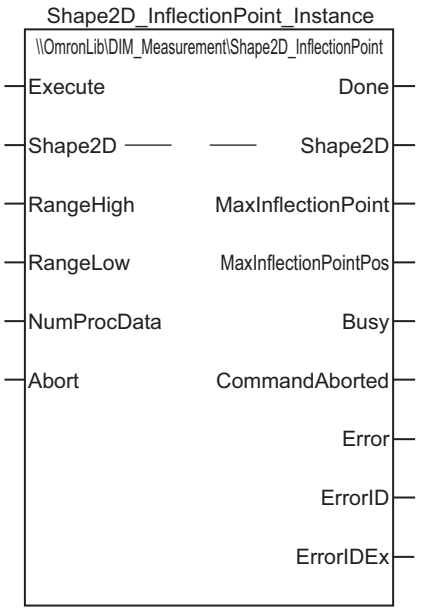
ST

```
//Edge 1 calculation
Edge1(
   Execute:=DIMStart_FeatureValueEdge,
   Shape2DInfo:=DIMShape2DInfo_Target,
   ShapeZValues:=DIMShapeZValues_Target,
   RangeHigh:=DIMEdge_RangeHigh1,
   RangeLow:=DIMEdge_RangeLow1,
   EdgeLevel:=DIMEdge_EdgeLevel1,
   EdgeType:=DIMEdge_EdgeType1,
   Count:=DIMEdge_Count1,
   Direction:=DIMEdge_Direction1,
   NumProcData:=Edge_NumProcData1,
   Abort:=Edge_Abort1,
   Done=>Edge_Done1,
   Position=>DIMEdge_Position1,
   Busy=>Edge_Busy1,
   CommandAborted=>Edge_CommandAborted1,
   Error=>Edge_Error1,
   ErrorID=>Edge_ErrorID1,
   ErrorIDEx=>Edge_ErrorIDEx1);
```

```
// Edge 2 calculation
Edge2(
   Execute:=DIMStart_FeatureValueEdge,
   Shape2DInfo:=DIMShape2DInfo_Target,
   ShapeZValues:=DIMShapeZValues_Target,
   RangeHigh:=DIMEdge_RangeHigh2,
   RangeLow:=DIMEdge_RangeLow2,
   EdgeLevel:=DIMEdge_EdgeLevel2,
   EdgeType:=DIMEdge_EdgeType2,
   Count:=DIMEdge_Count2,
   Direction:=DIMEdge_Direction2,
   NumProcData:=Edge_NumProcData2,
   Abort:=Edge_Abort2,
   Done=>Edge_Done2,
   Position=>DIMEdge_Position2,
   Busy=>Edge_Busy2,
   CommandAborted=>Edge_CommandAborted2,
   Error=>Edge_Error2,
   ErrorID=>Edge_ErrorID2,
   ErrorIDEx=>Edge_ErrorIDEx2);

//Edge 1 calculation completion held
IF Edge_Done1 = TRUE THEN
   Edge_Complete1 := TRUE;
ELSE;
END_IF;

// Edge 2 calculation completion held
IF Edge_Done2 = TRUE THEN
   Edge_Complete2 := TRUE;
ELSE;
END_IF;

IF Edge_Complete1 = TRUE & Edge_Complete2 = TRUE THEN
   //Result distance calculation
   DIMEdge_RsltWidth := DIMEdge_Position2 - DIMEdge_Position1;

   //Initialize edge calculation completion flag
   Edge_Complete1 := FALSE;
   Edge_Complete2 := FALSE;
ELSE;
END_IF;
```

Shape2D_InflectionPoint

The Shape2D_InflectionPoint function block measures the position at which the shape line is bended (inflection point) in the specified measurement range of the 2D shape data. If there are multiple inflection points in the measurement range, the position of the inflection point with the largest amount (sensitivity) of bend is output.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
Shape2D_ InfrectionPoint	2D Shape Inflection Point Measurement	FB		Shape2D_InflectionPoint_Instance (Execute, Shape2D, RangeHigh, RangeLow, NumProcData, Abort, Done, MaxInflectionPoint, MaxInflectionPointPos, Busy, CommandAborted, Error, ErrorID, ErrorIDEx)

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V1_1.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00143
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Product name	Model numbers	Version
Device	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
RangeHigh*1	Measurement Range Upper Limit	Specify the upper limit of the X-axis measurement range that is used for inflection point detection. The unit is mm.	LREAL	Negative number, positive number, or 0	0
RangeLow*1	Measurement Range Lower Limit	Specify the lower limit of the X-axis measurement range that is used for inflection point detection. The unit is mm.	LREAL	Negative number, positive number, or 0	0
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a task period in the segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
MaxInflection Point	Maximum Inflection Point Sensitivity	Outputs the maximum inflection point sensitivity.	LREAL	Depends on data type.	—
MaxInflection PointPos	Maximum Inflection Point Position	Outputs the position (X coordinate) of the maximum inflection point. The unit is mm.	LREAL	Depends on data type.	—
Busy	Measuring	TRUE when the measurement is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 293.

In-Out Variables

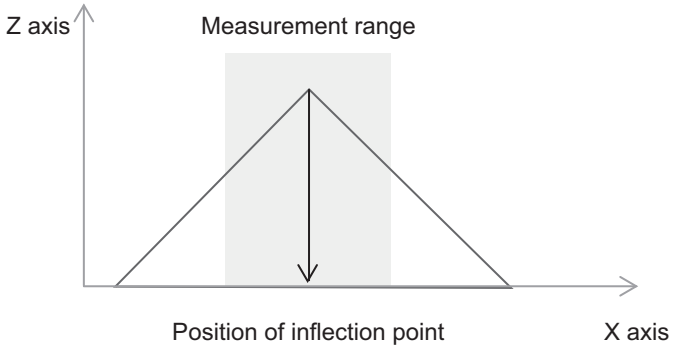
Variable	Name	Description	Data type	Valid range
Shape2D	2D Shape Data	Specify the 2D shape data. Refer to *LineMeasure_Cartesian* on page 108 for 2D shape data creation and correction.	sShape2D	—

Members of Structure sShape2D

Refer to *Members of Structure sShape2D* on page 241 of Shape2D_Height.

Function

The Shape2D_InflectionPoint function block measures the position at which the shape line is bended (inflection point) in the specified measurement range of the 2D shape data. If there are multiple inflection points in the measurement range, the position of the inflection point with the largest amount (sensitivity) of bend is output.



(a) The function block checks the input parameters when *Execute* changes to TRUE. If there is no error in the input data, *Busy* (Measuring) changes to TRUE. If there is an error, the function block execution is aborted and the error codes are output to *ErrorID* and *ErrorIDEx*. For details on the error codes, refer to *Troubleshooting* on page 293.

(b) The measurement range must be specified so that it contains at least one measurement data (Shape.Zvalue[X]).

If X does not meet the following condition, an error occurs.

RangeLow ≤ ((*Shape2D.Resolution* * (X-1)) / LREAL#1000) ≤ *RangeHigh*

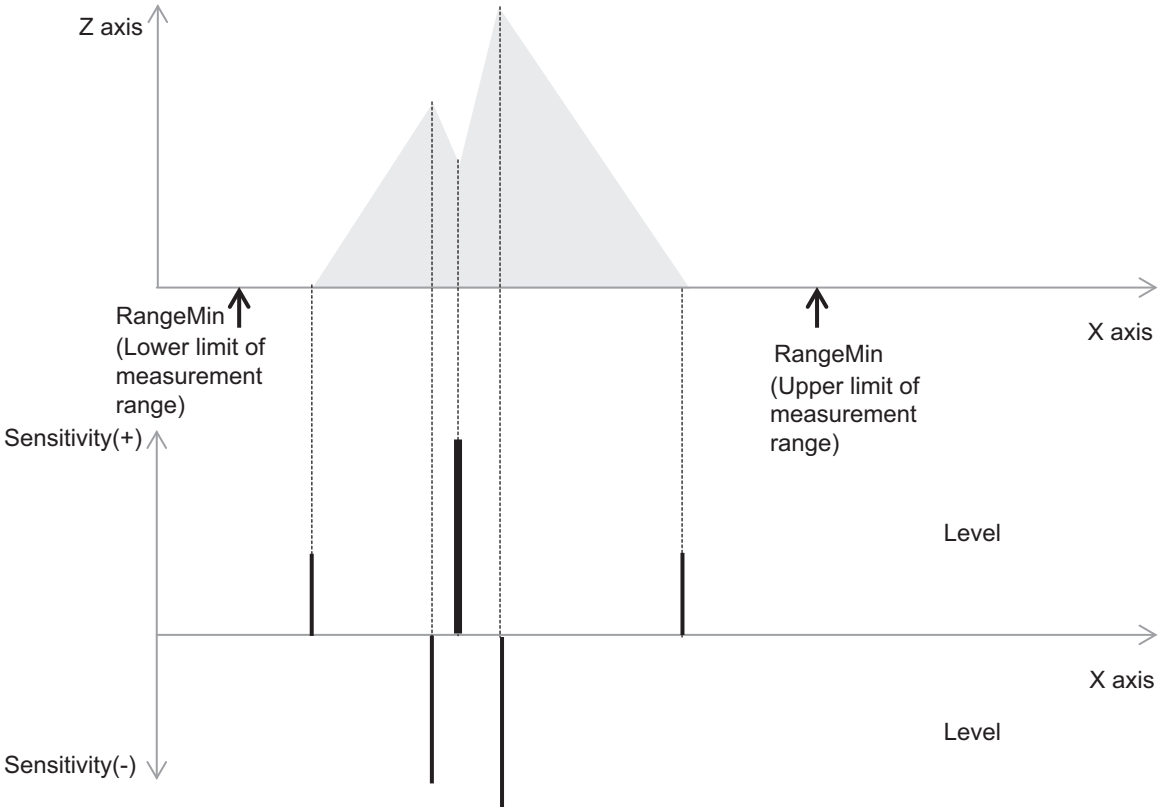
X can take any value between 1 and *Shape2D.DataCount*.

(c) Of the inflection points within the measurement range, the inflection point that is measured with the maximum sensitivity is output.

The sensitivities are compared as absolute values.

If there are multiple inflection points that were measured with the maximum sensitivity, the inflection point with the lowest position is output.

In the following example, the shape shown below is used for inflection point detection.



In the above figure, there are five inflection points.

Of these inflection points, the third from the left (bold line) is measured with the maximum sensitivity. So this point is recognized as the inflection point, and its sensitivity and position are output.

(d) This function block may cause a Task Period Exceeded error because it calculates the feature amount using a maximum of 20,000 2D shape data points.

To avoid this, you need to set the number of processed data points *(NumProcData)* per period.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, take into account the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

(e) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

(f) If an error occurs during the function block execution, *Error* changes to TRUE and the function block execution is aborted. For details on the error codes, refer to *Troubleshooting* on page 293.

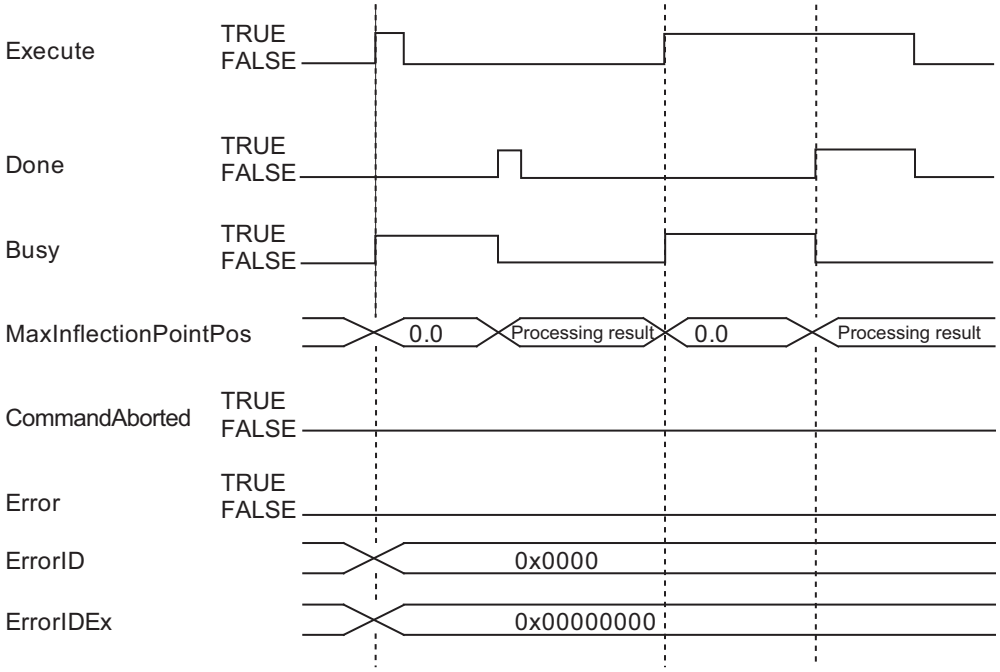
Re-execution of Function Blocks

If *Execute* is changed to TRUE again while execution is in progress (*Busy* = TRUE), the execution is aborted and the processing is performed from the beginning.

Timing Charts

The timing charts are shown below.

● Timing Chart in a Normal State



● Timing Chart When the Function Block Is Aborted

Refer to *Timing Chart When the Function Block Is Aborted* on page 244 of Shape2D_Height.

● Timing Chart When an Error Occurs

Refer to *Timing Chart When an Error Occurs (Startup Error)* on page 245 of Shape2D_Height.

Precautions for Correct Use

Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3CD2	0x0000 0001	The value of Measurement Range Upper Limit is not valid.	Set a valid real number.
	0x0000 0002	The value of Measurement Range Lower Limit is not valid.	Set a valid real number.
	0x0000 0003	A value specified for the measurement range is not valid.	Specify values that can meet the following condition: Measurement Range Upper Limit ≥ Measurement Range Lower Limit.
	0x0000 0004	There is no shape data in the measurement range.	Set the measurement range or 2D shape data again.
	0x0000 0005	The position of the inflection point was not detected.	Set appropriate values for the inflection point level, sign, and measurement direction again.
	0x0000 0006	The number of data points *(DataCount)* of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.
	0x0000 0007	The resolution (*Resolution*) of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.

Sample Programming

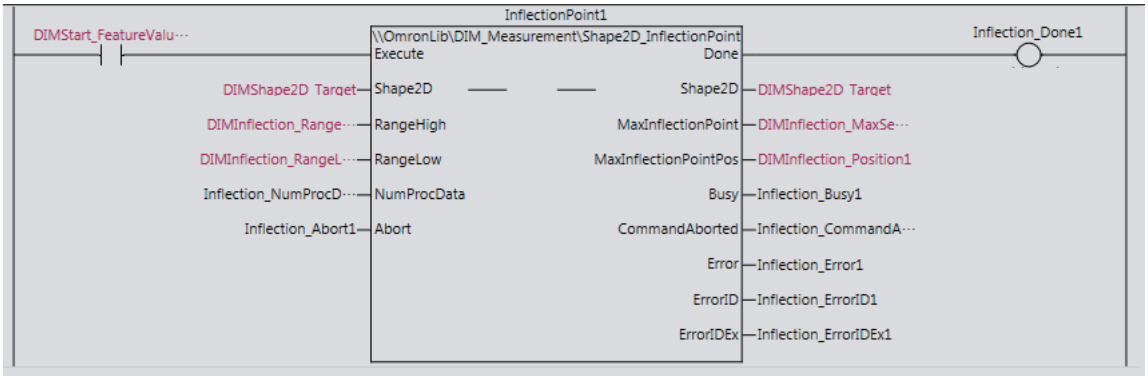
Program Description

This program calculates the position at which the inflection point of the shape line is detected in the specified measurement range of the 2D shape data.

Main Variables

Name	Meaning	Data type	Default	Description
DIMStart_Feature ValueInflectionPoint	Execution Trigger	BOOL	---	2D shape height measurement is started when this variable changes to TRUE. Assign this variable to the *Execute* input variable of Shape2D_InflectionPoint.
InflectionPoint1	2D Shape Inflection Point Measurement FB	Omron-Lib\DIM_Measure-ment\Shape2D_InflectionPoint	---	Instance of the function block that measures the inflection point of the 2D shape data.
DIMShape2D_Target	Target 2D Shape Data	Omron-Lib\DIM_Measure-ment\sShape2D	---	Set the Target 2D shape data. Assign this variable to the *Shape2D* input variable of Shape2D_InflectionPoint.
DIMInflection_Range High1	Measure-ment Range Upper Limit	LREAL	---	Specify the upper limit of the X-axis measurement range that is used for inflection point detection. The unit is mm. Assign this variable to the *RangeHigh* input variable of Shape2D_InflectionPoint.
DIMInflection_Range Low1	Measure-ment Range Lower Limit	LREAL	---	Specify the lower limit of the X-axis measurement range that is used for inflection point detection. The unit is mm. Assign this variable to the *RangeLow* input variable of Shape2D_InflectionPoint.
Inflection_NumProc Data1	Number of Processed Data Points	UINT	---	Set the number of data points that are processed in a task period during segmented processing. Assign this variable to the *NumProcData* input variable of Shape2D_InflectionPoint.
DIMInflection_Max Senstvt1	Maximum Inflection Point Sensitivity	LREAL	---	Outputs the maximum inflection point sensitivity. Assign this variable to the *MaxInflectionPoint* output variable of Shape2D_InflectionPoint.
DIMInflection _Position1	Maximum Inflection Point Position	LREAL	---	Outputs the position (X coordinate) of the maximum inflection point. The unit is mm. Assign this variable to the *MaxInflectionPointPos* output variable of Shape2D_InflectionPoint.

Ladder Diagram

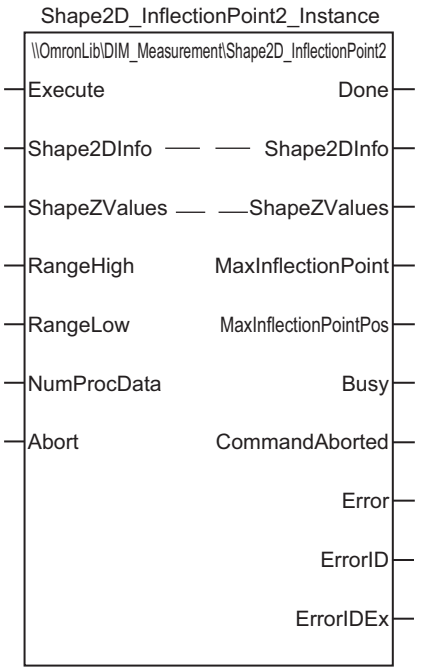


ST

```
//Inflection point calculation
InflectionPoint1(
   Execute:=DIMStart_FeatureValueInflectionPoint,
   Shape2D:=DIMShape2D_Target,
   RangeHigh:=DIMInflection_RangeHigh1,
   RangeLow:=DIMInflection_RangeLow1,
   NumProcData:=Inflection_NumProcData1,
   Abort:=Inflection_Abort1,
   Done=>Inflection_Done1,
   MaxInflectionPoint=>DIMInflection_MaxSenstvt1,
   MaxInflectionPointPos=>DIMInflection_Position1,
   Busy=>Inflection_Busy1,
   CommandAborted=>Inflection_CommandAborted1,
   Error=>Inflection_Error1,
   ErrorID=>Inflection_ErrorID1,
   ErrorIDEx=>Inflection_ErrorIDEx1);
```

Shape2D_InflectionPoint2

The Shape2D_InflectionPoint2 function block measures the position at which the shape line is bended (inflection point) in the specified measurement range of the 2D shape data for a variable-length array. If there are multiple inflection points in the measurement range, the position of the inflection point with the largest amount (sensitivity) of bend is output.

Function block and function name	Name	FB/ FUN	Graphic expression	ST expression
Shape2D_InflectionPoint2	2D Shape Inflection Point Measurement 2	FB		Shape2D_InflectionPoint2_Instance(Execute, Shape2DInfo, ShapeZValues, RangeHigh, RangeLow, NumProcData, Abort, Done, MaxInflectionPoint, MaxInflectionPointPos, Busy, CommandAborted, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V2_0.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00232
Source code	Not published

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
RangeHigh*1	Measurement Range Upper Limit	Specify the upper limit of the X-axis measurement range that is used for inflection point detection. The unit is mm.	LREAL	Negative number, positive number, or 0	0
RangeLow*1	Measurement Range Lower Limit	Specify the lower limit of the X-axis measurement range that is used for inflection point detection. The unit is mm.	LREAL	Negative number, positive number, or 0	0
NumProcData *1	Number of Processed Data Points	Specify the number of data points that are processed in a period during segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
MaxInflection-Point	Maximum Inflection Point Sensitivity	Outputs the maximum inflection point sensitivity.	LREAL	Depends on data type.	—
MaxInflection-PointPos	Maximum Inflection Point Position	Outputs the position (X coordinate) of the maximum inflection point. The unit is mm.	LREAL	Depends on data type.	—
Busy	Measuring	TRUE when the measurement is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 301 for details.

In-Out Variables

Variable	Name	Description	Data type	Valid range
Shape2DInfo	2D Shape Data Information	Specify the 2D shape data information. Refer to *LineMeasure_Cartesian2* on page 151 for 2D shape data creation and correction.	sShape2DInfo	—
ShapeZValues[]*1	2D Shape Data Array	Specify an array where the 2D shape data is stored.	ARRAY[*] OF LREAL	—

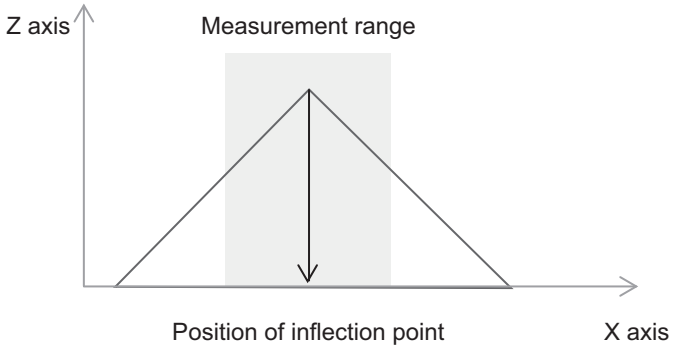
*1. The maximum number of array elements is 20,000. The first number of array element should be 0.

Members of Structure sShape2DInfo

Refer to *Structure* on page 256 of Shape2D_Height2.

Function

The Shape2D_InflectionPoint2 function block measures the position at which the shape line is bent (inflection point) in the specified measurement range of the 2D shape data for a variable-length array. If there are multiple inflection points in the measurement range, the position of the inflection point with the largest amount (sensitivity) of bend is output.



(a) The function block checks the input parameters when *Execute* changes to TRUE. If there is no error in the input data, *Busy* (Measuring) changes to TRUE. If there is an error, the function block execution is aborted and the error codes are output to *ErrorID* and *ErrorIDEx*. For details on the error codes, refer to *Troubleshooting* on page 301.

(b) The measurement range must be specified so that it contains at least one measurement data (*Shape.Zvalue[X]*).

If X does not meet the following condition, an error occurs.

RangeLow ≤ ((*Shape2DInfo.Resolution* * (X-1)) / LREAL#1000) ≤ *RangeHigh*

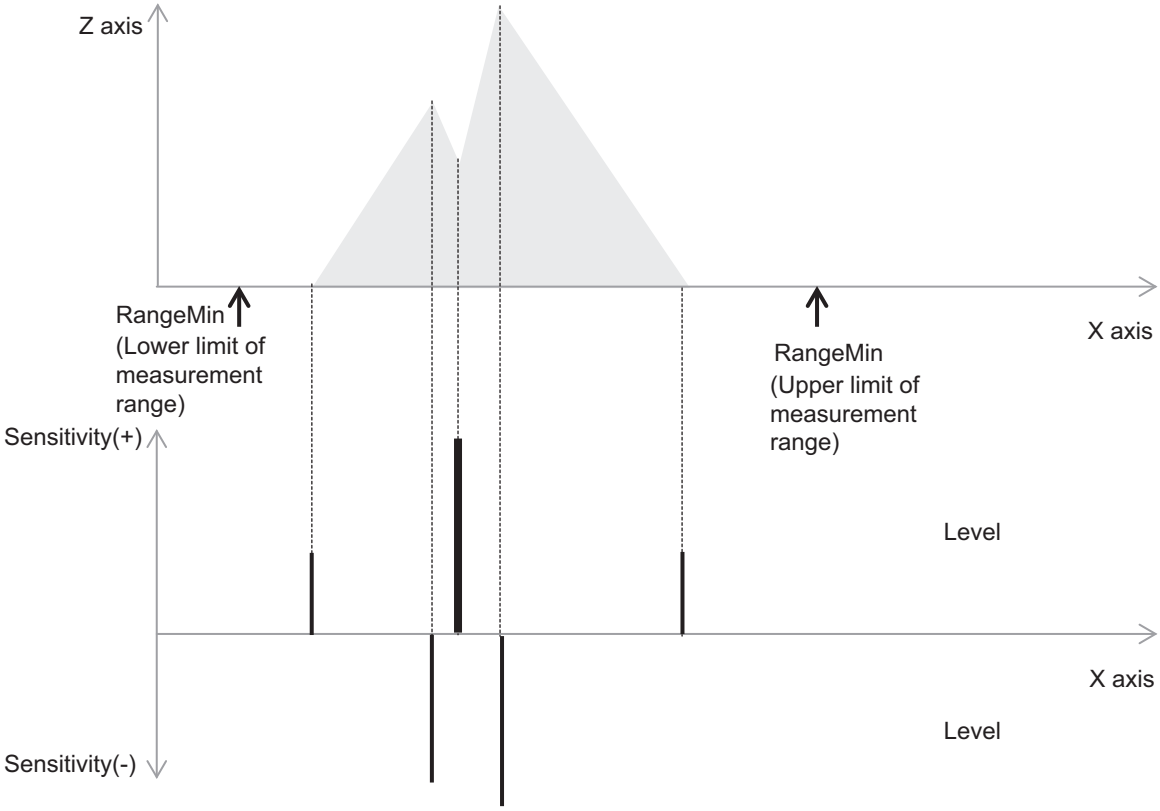
X can take any value between 1 and *Shape2DInfo.DataCount*.

(c) Of the inflection points within the measurement range, the inflection point that is measured with the maximum sensitivity is output.

The sensitivities are compared as absolute values.

If there are multiple inflection points that were measured with the maximum sensitivity, the inflection point with the lowest position is output.

In the following example, the shape shown below is used for inflection point detection.



In the above figure, there are five inflection points.

Of these inflection points, the third from the left (bold line) is measured with the maximum sensitivity. So this point is recognized as the inflection point, and its sensitivity and position are output.

(d) This function block may cause a Task Period Exceeded error because it calculates the feature amount using a maximum of 20,000 2D shape data points.

To avoid this, you need to set the number of processed data points *NumProcData* per period in the task that is executing.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, take into account the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

(e) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *Command-Aborted* changes to TRUE.

(f) If an error occurs during the function block execution, *Error* changes to TRUE and the function block execution is aborted. For details on the error codes, refer to *Troubleshooting* on page 301.

Additional Information

The time required to execute this function block varies depending on the measurement range, resolution of the 2D shape data, or other factors that determine the amount of processed data.

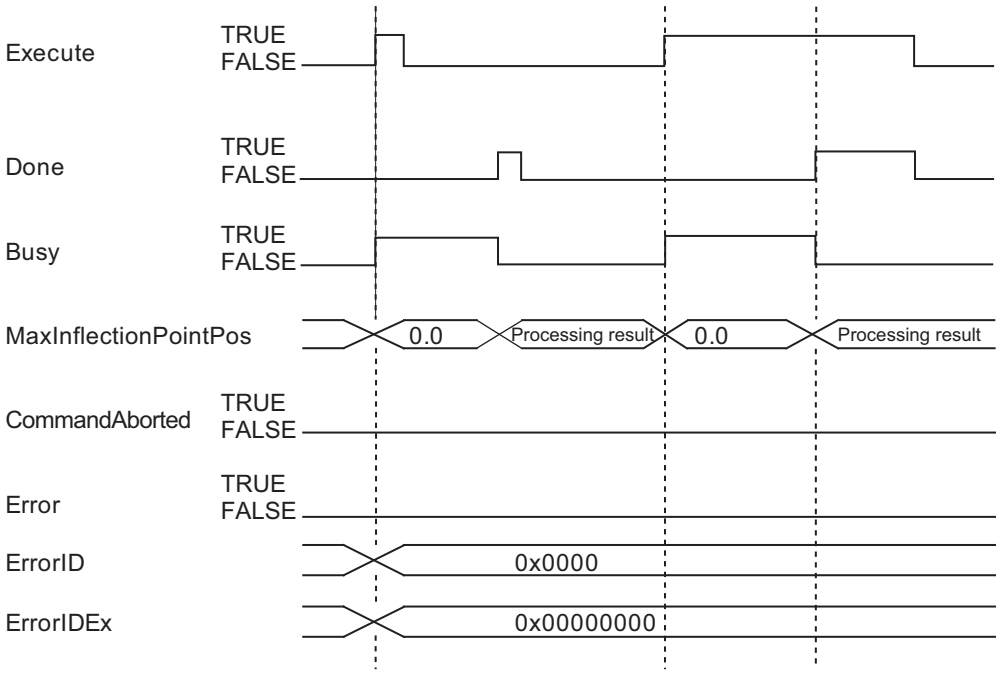
Re-execution of Function Blocks

If *Execute* is changed to TRUE again while execution is in progress (*Busy* = TRUE), the execution is aborted and the processing is performed from the beginning.

Timing Charts

The timing charts are shown below.

● Timing Chart in a Normal End State



● Timing Chart When the Function Block Is Aborted

Refer to *Timing Chart When the Function Block Is Aborted* on page 259 of Shape2D_Height2.

● Timing Chart When an Error Occurs (Startup Error)

Refer to *Timing Chart When an Error Occurs (Startup Error)* on page 260 of Shape2D_Height2.

● Timing Chart When an Error Occurs (Error during Function Block Execution)

Refer to *Timing Chart When an Error Occurs (Error during Function Block Execution)* on page 260 of Shape2D_Height2.

Precautions for Correct Use

Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x0000	0x00000000	Normal end	—
0x3CD2	0x00000001	The value of Measurement Range Upper Limit is not valid.	Set a valid real number.
	0x00000002	The value of Measurement Range Lower Limit is not valid.	Set a valid real number.
	0x00000003	A value specified for the measurement range is not valid.	Specify values that can meet the following condition: Measurement Range Upper Limit ≥ Measurement Range Lower Limit.
	0x00000004	There is no 2D shape data in the measurement range.	Set the measurement range or 2D shape data again.
	0x00000005	The position of the inflection point was not detected.	Check whether the 2D shape data is created correctly.
	0x00000006	The number of data points (*DataCount*) of the 2D shape data information is not valid.	Check whether the 2D shape data is created correctly.
	0x00000007	The resolution (*Resolution*) of the 2D shape data information is not valid.	Check whether the 2D shape data is created correctly.
	0x00000008	The specified array for the 2D shape data array (*ShapeZValues*) is out of the valid range.	The first number of array elements should be 0. Specify a value that the number of array elements is 20,000 or less.

Sample Programming

Program Description

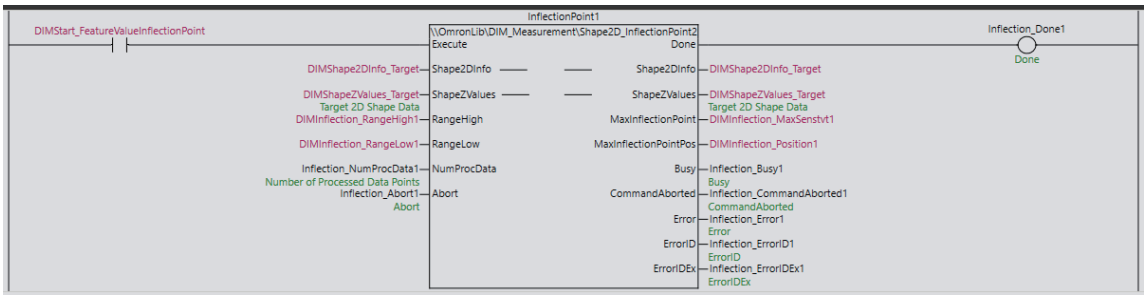
This program calculates the position at which the inflection point of the shape line is detected in the specified measurement range of the 2D shape data.

Main Variables

Name	Meaning	Data type	Default	Description
InflectionPoint1	2D Shape Inflection Point Measurement FB	OmronLib\DIM_Measurement\Shape2D_InflectionPoint2	—	Instance of the function block that measures the inflection point of the 2D shape data.

Name	Meaning	Data type	Default	Description
DIMStart_Feature ValueInflectionPoint	Execution Trigger	BOOL	—	The inflection point measurement for 2D shape is started when this variable changes to TRUE. Assign this variable to the *Execute* input variable of Shape2D_InflectionPoint2.
DIMShape2DInfo_ Target	2D Shape Data Information	Omron-Lib\DIM_Measurement\sShape2DInfo	—	Set the 2D shape data information. Assign this variable to the *Shape2DInfo* in-out variable of Shape2D_InflectionPoint2.
DIMShapeZValues_Target	2D Shape Data Array	ARRAY[0..19999] OF LREAL	—	Set the 2D shape data array. Assign this variable to the *ShapeZValues* in-out variable of Shape2D_InflectionPoint2.
DIMInflection_RangeHigh1	Measurement Range Upper Limit	LREAL	—	Specify the upper limit of the X-axis measurement range that is used for inflection point detection. The unit is mm. Assign this variable to the *RangeHigh* input variable of Shape2D_InflectionPoint2.
DIMInflection_RangeLow1	Measurement Range Lower Limit	LREAL	—	Specify the lower limit of the X-axis measurement range that is used for inflection point detection. The unit is mm. Assign this variable to the *RangeLow* input variable of Shape2D_InflectionPoint2.
Inflection_NumProc Data1	Number of Processed Data Points	UINT	—	Set the number of data points that are processed in a task period during segmented processing. Assign this variable to the *NumProcData* input variable of Shape2D_InflectionPoint2.
DIMInflection_Max Senstvt1	Maximum Inflection Point Sensitivity	LREAL	—	Outputs the maximum inflection point sensitivity. Assign this variable to the *MaxInflectionPoint* output variable of Shape2D_InflectionPoint2.
DIMInflection_ Position1	Maximum Inflection Point Position	LREAL	—	Outputs the position (X coordinate) of the maximum inflection point. The unit is mm. Assign this variable to the *MaxInflectionPointPos* output variable of Shape2D_InflectionPoint2.

Ladder Diagram



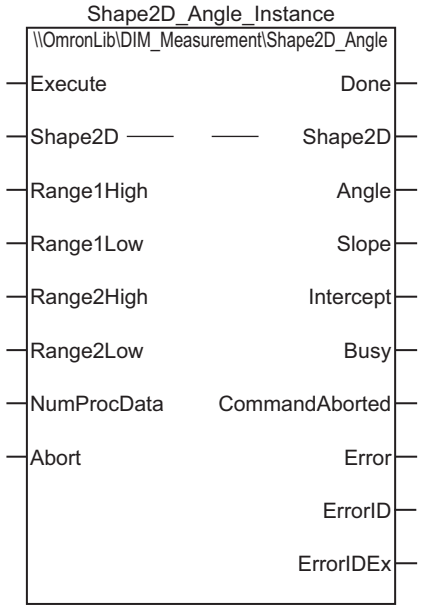
ST

```
//Inflection point calculation
InflectionPoint1(
   Execute:=DIMStart_FeatureValueInflectionPoint,
   Shape2DInfo:=DIMShape2DInfo_Target,
   ShapeZValues:=DIMShapeZValues_Target,
   RangeHigh:=DIMInflection_RangeHigh1,
   RangeLow:=DIMInflection_RangeLow1,
   NumProcData:=Inflection_NumProcData1,
   Abort:=Inflection_Abort1,
   Done=>Inflection_Done1,
   MaxInflectionPoint=>DIMInflection_MaxSenstvt1,
   MaxInflectionPointPos=>DIMInflection_Position1,
   Busy=>Inflection_Busy1,
   CommandAborted=>Inflection_CommandAborted1,
   Error=>Inflection_Error1,
   ErrorID=>Inflection_ErrorID1,
   ErrorIDEx=>Inflection_ErrorIDEx1);
```

Shape2D_Angle

The Shape2D_Angle function block draws a straight line between heights within in the two calculation measurement ranges of the 2D shape data. Then, the function block calculates the angle θ of the straight line to the horizontal surface.

This function block also outputs slope a and intercept b of the straight line relative to the horizontal axis X and vertical axis Z (height).

Function block name	Name	FB/ FUN	Graphic expression	ST expression
Shape2D_Angle	2D Shape Angle Measurement	FB		Shape2D_Angle_Instance (Execute, Shape2D, Range1High, Range1Low, Range2High, Range2Low, NumProcData, Abort, Done, Angle, Slope, Intercept, Busy, CommandAborted, Error, ErrorID, ErrorIDEx)

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V1_1.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00144
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Product name	Model numbers	Version
Device	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
Range1High*1	Measurement Range 1 Upper Limit	Specify the upper limit of the X-axis measurement range 1 that is used for straight line calculation. The unit is mm.	LREAL	Negative number, positive number, or 0	0
Range1Low*1	Measurement Range 1 Lower Limit	Specify the lower limit of the X-axis measurement range 1 that is used for straight line calculation. The unit is mm.	LREAL	Negative number, positive number, or 0	0
Range2High*1	Measurement Range 2 Upper Limit	Specify the upper limit of the X-axis measurement range 2 that is used for straight line calculation. The unit is mm.	LREAL	Negative number, positive number, or 0	0
Range2Low*1	Measurement Range 2 Lower Limit	Specify the lower limit of the X-axis measurement range 2 that is used for straight line calculation. The unit is mm.	LREAL	Negative number, positive number, or 0	0
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a task period in the segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
Angle	Angle	Outputs the calculated angle of the straight line. The unit is θ.	LREAL	-90° to 90°*1	—
Slope	Slope	Outputs the calculated slope of the straight line.	LREAL	Depends on data type.	—
Intercept	Intercept	Outputs the calculated Z-intercept of the straight line.	LREAL	Depends on data type.	—
Busy	Measuring	TRUE when the measurement is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*2	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*2	—

*1. Measurement is impossible when the slope and intercept cannot be calculated.
Refer to *Function* on page 307 for details.

*2. Refer to *Troubleshooting* on page 309.

In-Out Variables

Variable	Name	Description	Data type	Valid range
Shape2D	2D Shape Data	Specify the 2D shape data. Refer to *LineMeasure_Cartesian* on page 108 for 2D shape data creation and correction.	sShape2D	—

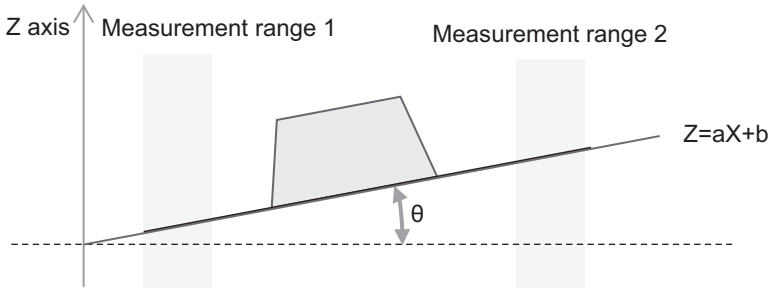
Members of Structure sShape2D

Refer to *Members of Structure sShape2D* on page 241 of Shape2D_Height.

Function

The Shape2D_Angle function block draws a straight line between heights within the two calculation measurement ranges of the 2D shape data. Then, it calculates the angle θ of the straight line to the horizontal surface.

This function block also outputs slope a and intercept b of the straight line relative to the horizontal axis X and vertical axis Z (height).



(a) The function block checks the input parameters when *Execute* changes to TRUE. If there is no error in the input data, *Busy* (Measuring) changes to TRUE. If there is an error, the function block execution is aborted and the error codes are output to *ErrorID* and *ErrorIDEx*. For details on the error codes, refer to *Troubleshooting* on page 309.

(b) The measurement range must be specified so that it contains at least one measurement data (Shape.Zvalue[X]).

If X does not meet the following condition, an error occurs.

RangeLow ≤((*Shape2D.Resolution* * (X-1)) / LREAL#1000) ≤ *RangeHigh*

X can take any value between 1 and *Shape2D.DataCount*.

(c) The function block draws a straight line using all heights within the specified measurement range 1 and measurement range 2.

(d) When a straight line is drawn successfully, the function block calculates and outputs the angle of the straight line relative to the horizontal line. The output values are the slope and Z-intercept of the straight line.

If the angle in the measurement range is close to 90° or -90° and the slope and intercept cannot be calculated, a measurement error occurs.

(e) This function block may cause a Task Period Exceeded error because it calculates the feature amount using a maximum of 20,000 2D shape data points.

To avoid this, you need to set the number of processed data points *(NumProcData)* per period.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, take into account the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

(f) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

(g) If an error occurs during the function block execution, *Error* changes to TRUE and the function block execution is aborted. For details on the error codes, refer to *Troubleshooting* on page 309.

Additional Information

The time required to execute this function block varies depending on the measurement range, resolution of the 2D shape data, or other factors that determine the amount of processed data.

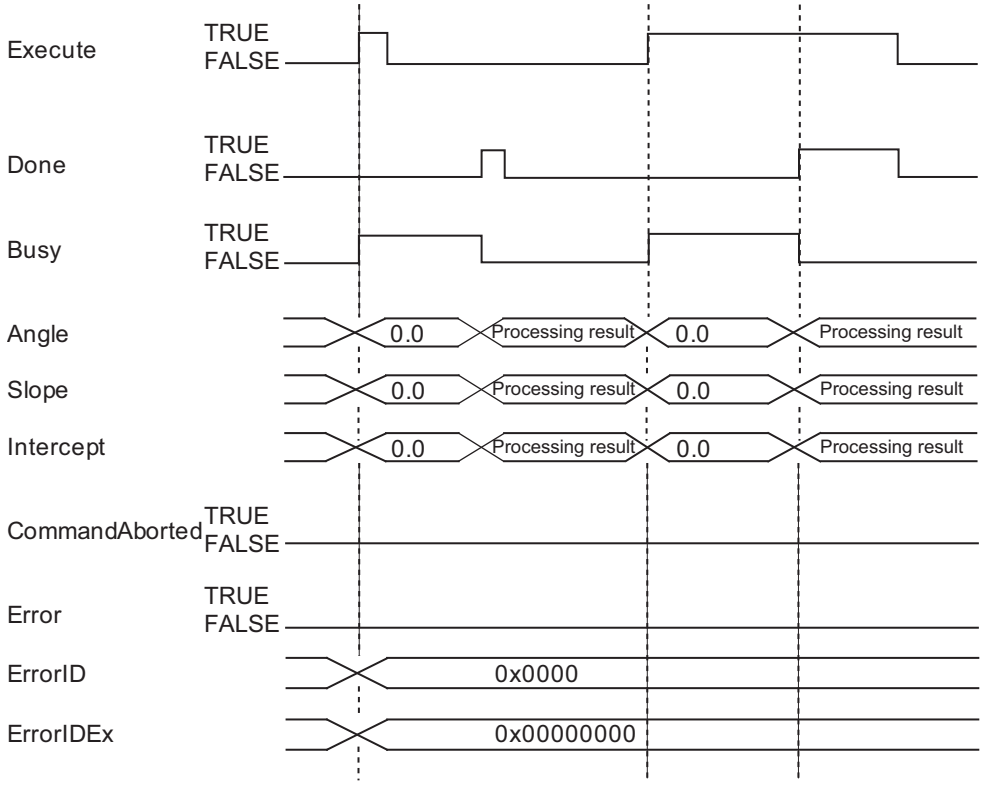
Re-execution of Function Blocks

If *Execute* is changed to TRUE again while execution is in progress (*Busy* = TRUE), the execution is aborted and the processing is performed from the beginning.

Timing Charts

The timing charts are shown below.

Timing Chart in a Normal State



Timing Chart When the Function Block Is Aborted

Refer to *Timing Chart When the Function Block Is Aborted* on page 244 of Shape2D_Height.

Timing Chart When an Error Occurs

Refer to *Timing Chart When an Error Occurs (Startup Error)* on page 245 of Shape2D_Height.

Precautions for Correct Use

Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3CD3	0x0000 0001	The value of Measurement Range 1 Upper Limit is not valid.	Set a valid real number.
	0x0000 0002	The value of Measurement Range 1 Lower Limit is not valid.	Set a valid real number.
	0x0000 0003	A value specified for the measurement range 1 is not valid.	Specify values that can meet the following condition: Measurement Range 1 Upper Limit ≥ Measurement Range 1 Lower Limit.
	0x0000 0004	The value of Measurement Range 2 Upper Limit is not valid.	Set a valid real number.
	0x0000 0005	The value of Measurement Range 2 Lower Limit is not valid.	Set a valid real number.
	0x0000 0006	A value specified for the measurement range 2 is not valid.	Specify values that can meet the following condition: Measurement Range 2 Upper Limit ≥ Measurement Range 1 Lower Limit.
	0x0000 0007	The measurement range 1 and measurement range 2 are overlapped or inverted.	Specify values that can meet the following condition: measurement range 1 > measurement range 2.
	0x0000 0008	There is no shape data in the measurement range 1.	Set the measurement range 1 or 2D shape data again.
	0x0000 0009	There is no shape data in the measurement range 2.	Set the measurement range 2 or 2D shape data again.
	0x0000 000A	The slope or intercept of the straight line cannot be calculated.	Set the measurement range 1, measurement range 2 or 2D shape data again.
	0x0000 000B	The number of data points *(DataCount)* of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.
	0x0000 000C	The resolution (*Resolution*) of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.

Sample Programming

Program Description

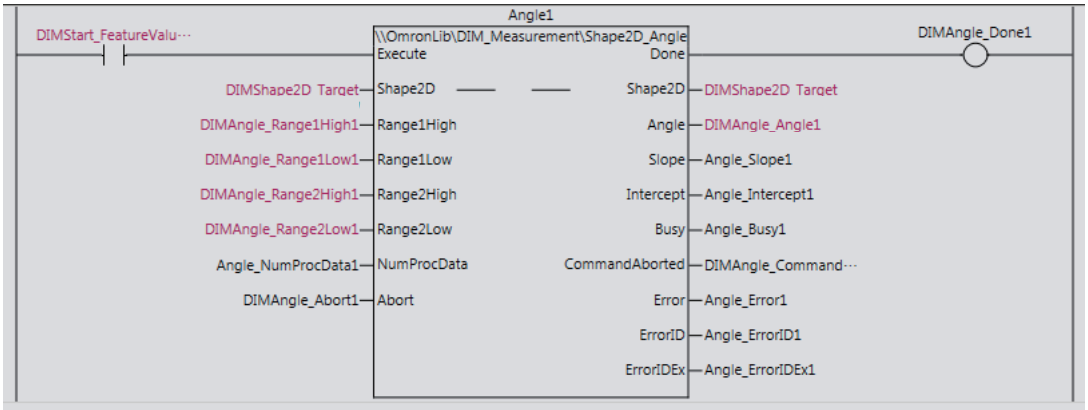
This program draws a straight line between heights within in the calculation measurement ranges of the 2D shape data. Then, the program calculates the angle, slope and intercept.

Main Variables

Name	Meaning	Data type	Default	Description
DIMStart_Feature ValueAngle	Execution Trigger	BOOL	---	2D shape height measurement is started when this variable changes to TRUE. Assign this variable to the *Execute* input variable of Shape2D_Angle.
Angle1	2D Shape Angle Measurement FB	OmronLib\ DIM_Measurement\ Shape2D_ Angle	---	Instance of the function block that measures the angle of the 2D shape data.
DIMShape2D_Target	Target 2D Shape Data	OmronLib\ DIM_Measurement\ sShape2D	---	Set the target 2D shape data. Assign this variable to the *Shape2D* input variable of Shape2D_Angle.
DIMAngle_Range1 High1	Measurement Range 1 Upper Limit	LREAL	---	Specify the upper limit of the X-axis measurement range 1 that is used for straight line calculation. The unit is mm. Assign this variable to the *Range1High* input variable of Shape2D_Angle.
DIMAngle_Range1 Low1	Measurement Range 1 Lower Limit	LREAL	---	Specify the lower limit of the X-axis measurement range 1 that is used for straight line calculation. The unit is mm. Assign this variable to the *Range1Low* input variable of Shape2D_Angle.
DIMAngle_Range2 High1	Measurement Range 2 Upper Limit	LREAL	---	Specify the upper limit of the X-axis measurement range 2 that is used for straight line calculation. The unit is mm. Assign this variable to the *Range2High* input variable of Shape2D_Angle.
DIMAngle_Range2 Low1	Measurement Range 2 Lower Limit	LREAL	---	Specify the lower limit of the X-axis measurement range 2 that is used for straight line calculation. The unit is mm. Assign this variable to the *Range2Low* input variable of Shape2D_Angle.
Angle_NumProcData1	Number of Processed Data Points	UINT	---	Set the number of data points that are processed in a task period during segmented processing. Assign this variable to the *NumProcData* input variable of Shape2D_Angle.
DIMAngle_Angle1	Angle	LREAL	---	Outputs the calculated angle of the straight line. The unit is θ. Assign this variable to the *Angle* output variable of Shape2D_Angle.
Angle_Slope1	Slope	LREAL	---	Outputs the calculated slope of the straight line. Assign this variable to the *Slope* output variable of Shape2D_Angle.

Name	Meaning	Data type	Default	Description
Angle_Intercept1	Intercept	LREAL	---	Outputs the calculated Z-intercept of the straight line. Assign this variable to the *Intercept* output variable of Shape2D_Angle.

Ladder Diagram



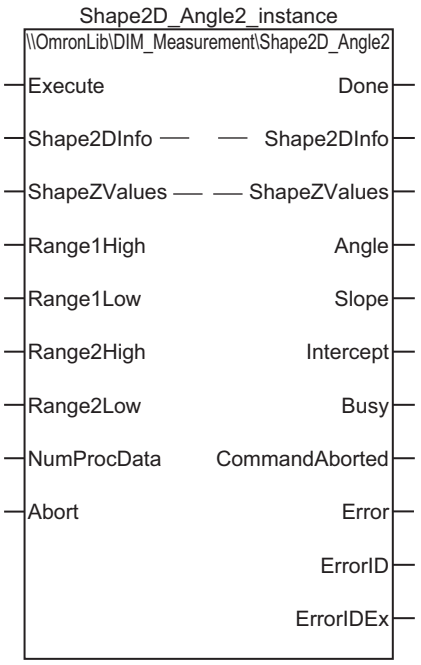
ST

```
//Angle, slope and intercept calculation
Angle1(
   Execute:=DIMStart_FeatureValueAngle,
   Shape2D:=DIMShape2D_Target,
   Range1High:=DIMAngle_Range1High1,
   Range1Low:=DIMAngle_Range1Low1,
   Range2High:=DIMAngle_Range2High1,
   Range2Low:=DIMAngle_Range2Low1,
   NumProcData:=Angle_NumProcData1,
   Abort:=Angle_Abort1,
   Done=>Angle_Done1,
   Angle=>DIMAngle_Angle1,
   Slope=>Angle_Slope1,
   Intercept=>Angle_Intercept1,
   Busy=>Angle_Busy1,
   CommandAborted=>Angle_CommandAborted1,
   Error=>Angle_Error1,
   ErrorID=>Angle_ErrorID1,
   ErrorIDEx=>Angle_ErrorIDEx1);
```

Shape2D_Angle2

The Shape2D_Angle2 function block draws a straight line between heights within the two calculation measurement ranges of the 2D shape data for a variable-length array. Then, it calculates the angle θ of the straight line to the horizontal surface.

This function block also outputs slope a and intercept b of the straight line relative to the horizontal axis X and vertical axis Z (height).

Function block and function name	Name	FB/ FUN	Graphic expression	ST expression
Shape2D_Angle2	2D Shape Angle Measurement 2	FB		Shape2D_Angle2_instance(Execute, Shape2DInfo, ShapeZValues, Range1High, Range1Low, Range2High, Range2Low, NumProcData, Abort, Done, Angle, Slope, Intercept, Busy, CommandAborted, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V2_0.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00233
Source code	Not published

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
Range1High*1	Measurement Range 1 Upper Limit	Specify the upper limit of the X-axis measurement range 1 that is used for straight line detection. The unit is mm.	LREAL	Negative number, positive number, or 0	0
Range1Low*1	Measurement Range 1 Lower Limit	Specify the lower limit of the X-axis measurement range 1 that is used for straight line detection. The unit is mm.	LREAL	Negative number, positive number, or 0	0
Range2High*1	Measurement Range 2 Upper Limit	Specify the upper limit of the X-axis measurement range 2 that is used for straight line detection. The unit is mm.	LREAL	Negative number, positive number, or 0	0
Range2Low*1	Measurement Range 2 Lower Limit	Specify the lower limit of the X-axis measurement range 2 that is used for straight line detection. The unit is mm.	LREAL	Negative number, positive number, or 0	0
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a period during segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
Angle	Angle	Outputs the calculated angle of the straight line. The unit is °.	LREAL	-90° to 90°*1	—
Slope	Slope	Outputs the calculated slope of the straight line.	LREAL	Depends on data type.	—
Intercept	Intercept	Outputs the calculated Z-intercept of the straight line.	LREAL	Depends on data type.	—
Busy	Measuring	TRUE when the measurement is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*2	—

Variable	Name	Description	Data type	Valid range	Default
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*2	—

*1. Measurement is impossible when the slope and intercept cannot be calculated.
Refer to *Function* on page 315 for details.

*2. Refer to *Troubleshooting* on page 317 for details.

In-Out Variables

Variable	Name	Description	Data type	Valid range
Shape2DInfo	2D Shape Data Information	Specify the 2D shape data information. Refer to *LineMeasure_Cartesian2* on page 151 for 2D shape data creation and correction.	sShape2DInfo	—
ShapeZValues[]*1	2D Shape Data Array	Specify an array where the 2D shape data is stored.	ARRAY[*] OF LREAL	—

*1. The maximum number of array elements is 20,000. The first number of array element should be 0.

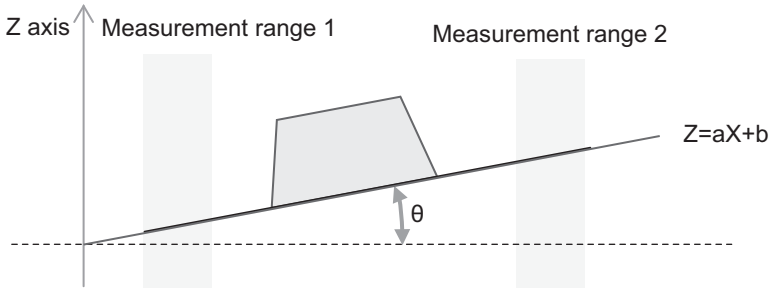
Members of Structure sShape2DInfo

Refer to *Structure* on page 256 of Shape2D_Height2.

Function

The Shape2D_Angle2 function block draws a straight line between heights within the two calculation measurement ranges of the 2D shape data for a variable-length array. Then, it calculates the angle θ of the straight line to the horizontal surface.

This function block also outputs slope a and intercept b of the straight line relative to the horizontal axis X and vertical axis Z (height).



(a) The function block checks the input parameters when *Execute* changes to TRUE. If there is no error in the input data, *Busy* (Measuring) changes to TRUE. If there is an error, the function block execution is aborted and the error codes are output to *ErrorID* and *ErrorIDEx*. For details on the error codes, refer to *Troubleshooting* on page 317.

(b) The measurement range must be specified so that it contains at least one measurement data (*Shape.Zvalue[X]*).

If X does not meet the following condition, an error occurs.

RangeLow ≤ ((*Shape2DInfo.Resolution* * (X-1)) / LREAL#1000) ≤ *RangeHigh*

X can take any value between 1 and *Shape2D.DataCount*.

(c) The function block draws a straight line using all heights within the specified measurement range 1 and measurement range 2.

(d) When a straight line is drawn successfully, the function block calculates and outputs the angle of the straight line relative to the horizontal line. The output values are the slope and Z-intercept of the straight line.

If the angle in the measurement range is close to 90° or -90° and the slope and intercept cannot be calculated, a measurement error occurs.

(e) This function block may cause a Task Period Exceeded error because it calculates the feature amount using a maximum of 20,000 2D shape data points.

To avoid this, you need to set the number of processed data points *NumProcData* per period in the task that is executing.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, take into account the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

(f) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

(g) If an error occurs during the function block execution, *Error* changes to TRUE and the function block execution is aborted. For details on the error codes, refer to *Troubleshooting* on page 317.

Additional Information

The time required to execute this function block varies depending on the measurement range, resolution of the 2D shape data, or other factors that determine the amount of processed data.

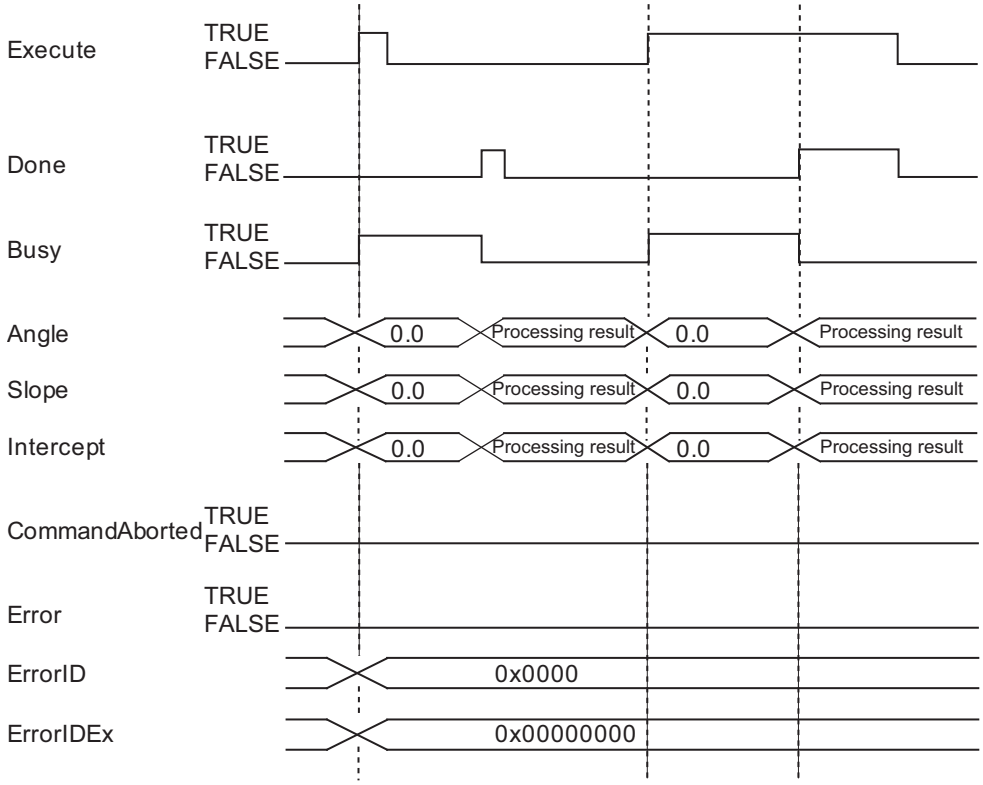
Re-execution of Function Blocks

If *Execute* is changed to TRUE again while execution is in progress (*Busy* = TRUE), the execution is aborted and the processing is performed from the beginning.

Timing Charts

The timing charts are shown below.

● Timing Chart in a Normal End State



● Timing Chart When the Function Block Is Aborted

Refer to *Timing Chart When the Function Block Is Aborted* on page 259 of Shape2D_Height2.

● Timing Chart When an Error Occurs (Startup Error)

Refer to *Timing Chart When an Error Occurs (Startup Error)* on page 260 of Shape2D_Height2.

● Timing Chart When an Error Occurs (Error during Function Block Execution)

Refer to *Timing Chart When an Error Occurs (Error during Function Block Execution)* on page 260 of Shape2D_Height2.

Precautions for Correct Use

Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x0000	0x00000000	Normal end	—
0x3CD3	0x00000001	The value of Measurement Range 1 Upper Limit is not valid.	Set a valid real number.
	0x00000002	The value of Measurement Range 1 Lower Limit is not valid.	Set a valid real number.
	0x00000003	A value specified for the measurement range 1 is not valid.	Specify values that can meet the following condition: Measurement Range 1 Upper Limit ≥ Measurement Range 1 Lower Limit.
	0x00000004	The value of Measurement Range 2 Upper Limit is not valid.	Set a valid real number.
	0x00000005	The value of Measurement Range 2 Lower Limit is not valid.	Set a valid real number.
	0x00000006	A value specified for Measurement Range 2 is not valid.	Specify values that can meet the following condition: Measurement Range 2 Upper Limit ≥ Measurement Range 2 Lower Limit.
	0x00000007	The measurement range 1 and measurement range 2 are overlapped or inverted.	Specify values that can meet the following condition: measurement range 1 > measurement range 2.
	0x00000008	There is no 2D shape data in Measurement Range 1.	Set the measurement range 1 or 2D shape data again.
	0x00000009	There is no 2D shape data in Measurement Range 2.	Set Measurement Range 2 or 2D shape data again.
	0x0000000A	The slope or intercept of the straight line cannot be calculated.	Set the measurement range 1, measurement range 2 or 2D shape data again.
	0x0000000B	The number of data points (*DataCount*) of the 2D shape data information is not valid.	Check whether the 2D shape data is created correctly.
	0x0000000C	The resolution (*Resolution*) of the 2D shape data information is not valid.	Check whether the 2D shape data is created correctly.
	0x0000000D	The specified array for the 2D shape data array (*ShapeZValues*) is out of the valid range.	The first number of array elements should be 0. Specify a value that the number of array elements is 20,000 or less.

Sample Programming

Program Description

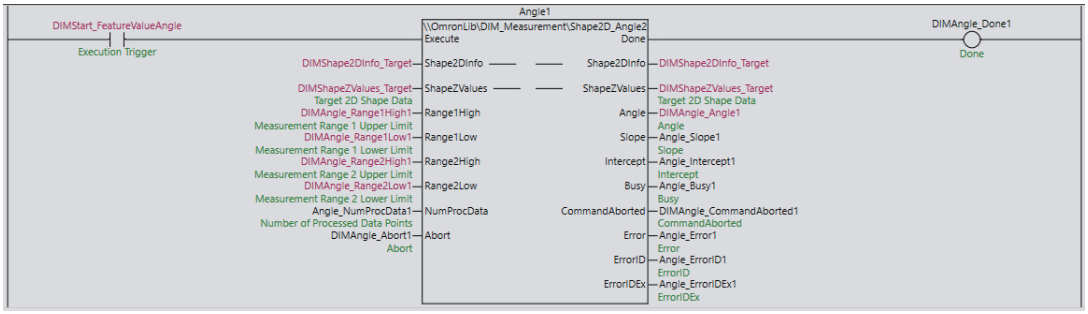
This program draws a straight line between heights within in the calculation measurement ranges of the 2D shape data. Then, the program calculates the angle, slope and intercept.

Main Variables

Name	Meaning	Data type	Default	Description
Angle1	Angle Measurement FB	OmronLib\DIM_Measurement\Shape2D_Angle2	—	Instance of the function block that measures the angle of the 2D shape data.
DIMStart_FeatureValueAngle	Execution Trigger	BOOL	—	The 2D shape angle measurement is started when this variable changes to TRUE. Assign this variable to the *Execute* input variable of Shape2D_Angle2.
DIMShape2DInfo_Target	2D Shape Data Information	OmronLib\DIM_Measurement\sShape2DInfo	—	Set the 2D shape data information. Assign this variable to the *Shape2DInfo* in-out variable of Shape2D_Angle2.
DIMShapeZValues_Target	2D Shape Data Array	ARRAY[0..19999] OF LREAL	—	Set the 2D shape data array. Assign this variable to the *ShapeZValues* in-out variable of Shape2D_Angle2.
DIMAngle_Range1High1	Measurement Range 1 Upper Limit	LREAL	—	Specify the upper limit of the X-axis measurement range 1 that is used for straight line calculation. The unit is mm. Assign this variable to the *Range1High* input variable of Shape2D_Angle2.
DIMAngle_Range1Low1	Measurement Range 1 Lower Limit	LREAL	—	Specify the lower limit of the X-axis measurement range 1 that is used for straight line calculation. The unit is mm. Assign this variable to the *Range1Low* input variable of Shape2D_Angle2.
DIMAngle_Range2High1	Measurement Range 2 Upper Limit	LREAL	—	Specify the upper limit of the X-axis measurement range 2 that is used for straight line calculation. The unit is mm. Assign this variable to the *Range2High* input variable of Shape2D_Angle2.
DIMAngle_Range2Low1	Measurement Range 2 Lower Limit	LREAL	—	Specify the lower limit of the X-axis measurement range 2 that is used for straight line calculation. The unit is mm. Assign this variable to the *Range2Low* input variable of Shape2D_Angle2.
Angle_NumProcData1	Number of Processed Data Points	UINT	—	Set the number of data points that are processed in a task period during segmented processing. Assign this variable to the *NumProcData* input variable of Shape2D_Angle2.

Name	Meaning	Data type	Default	Description
DIMAngle_Angle1	Angle	LREAL	—	Outputs the calculated angle of the straight line. The unit is °. Assign this variable to the *Angle* output variable of Shape2D_Angle2.
Angle_Slope1	Slope	LREAL	—	Outputs the calculated slope of the straight line. Assign this variable to the *Slope* output variable of Shape2D_Angle2.
Angle_Intercept1	Intercept	LREAL	—	Outputs the calculated Z-intercept of the straight line. Assign this variable to the *Intercept* output variable of Shape2D_Angle2.

Ladder Diagram



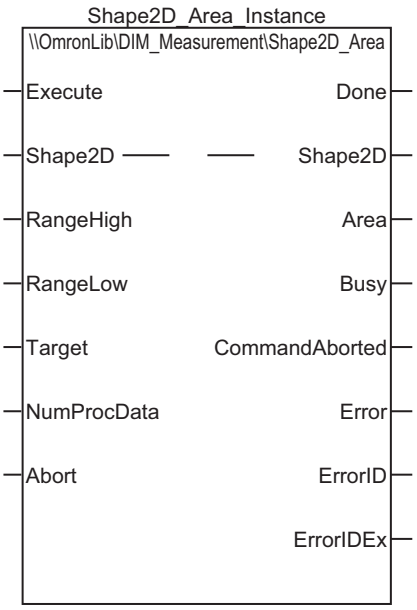
ST

```
//Angle, slope and intercept calculation
Angle1(
   Execute:=DIMStart_FeatureValueAngle,
   Shape2DInfo:=DIMShape2DInfo_Target,
   ShapeZValues:=DIMShapeZValues_Target,
   Range1High:=DIMAngle_Range1High1,
   Range1Low:=DIMAngle_Range1Low1,
   Range2High:=DIMAngle_Range2High1,
   Range2Low:=DIMAngle_Range2Low1,
   NumProcData:=Angle_NumProcData1,
   Abort:=Angle_Abort1,
   Done=>Angle_Done1,
   Angle=>DIMAngle_Angle1,
   Slope=>Angle_Slope1,
   Intercept=>Angle_Intercept1,
   Busy=>Angle_Busy1,
   CommandAborted=>Angle_CommandAborted1,
   Error=>Angle_Error1,
   ErrorID=>Angle_ErrorID1,
   ErrorIDEx=>Angle_ErrorIDEx1);
```

Shape2D_Area

The Shape2D_Area function block calculates the area in the specified range. A range of the 2D shape data must be specified as the integral range.

The function block determines the base using shape data within the integral range, and calculates the area (sectional area) of a region bounded by the base and waveform of the 2D shape data.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
Shape2D_Area	2D Shape Sectional Area Measurement	FB		Shape2D_Area_Instance (Execute, Shape2D, RangeHigh, RangeLow, Target, NumProcData, Abort, Done, Area, Busy, CommandAborted, Error, ErrorID, ErrorIDEx)

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V1_1.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00145
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Product name	Model numbers	Version
Device	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
RangeHigh*1	Integral Range Upper Limit	Specify the upper limit of the X-axis range that is used for integral calculation. The unit is mm.	LREAL	Negative number, positive number, or 0	0
RangeLow*1	Integral Range Lower Limit	Specify the lower limit of the X-axis range that is used for integral calculation. The unit is mm.	LREAL	Negative number, positive number, or 0	0
Target*1	Target of Integration	Specify in which direction (above or below base) the height is recognized as the target of integration. FALSE: Height above base TRUE: Height below base	BOOL	TRUE or FALSE	FALSE
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a task period in the segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
Area	Sectional Area	Outputs the calculated sectional area. The unit is mm^2.	LREAL	Depends on data type.	—
Busy	Measuring	TRUE when the measurement is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 326.

In-Out Variables

Variable	Name	Description	Data type	Valid range
Shape2D	2D Shape Data	Specify the 2D shape data. Refer to *LineMeasure_Cartesian* on page 108 for 2D shape data creation and correction.	sShape2D	—

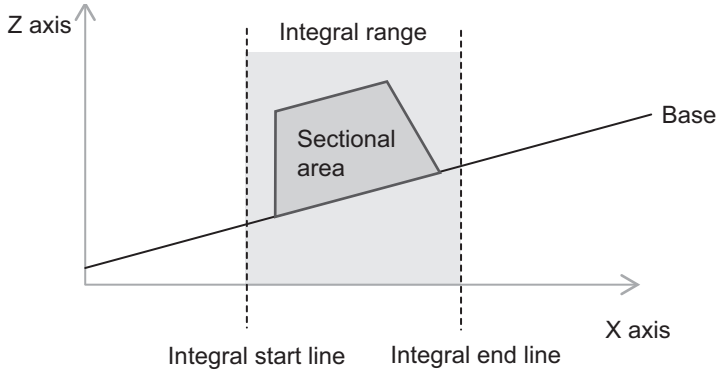
Members of Structure sShape2D

Refer to *Members of Structure sShape2D* on page 241 of Shape2D_Height.

Function

The Shape2D_Area function block calculates the area in the specified range. A range of the 2D shape data must be specified as the integral range.

The function block determines the base using shape data within the integral range, and calculates the area (sectional area) of a region bounded by the base and waveform of the 2D shape data.



The function block checks the input parameters when *Execute* changes to TRUE. If there is no error in the input data, *Busy* (Measuring) changes to TRUE. If there is an error, the function block execution is aborted and the error codes are output to *ErrorID* and *ErrorIDEx*. For details on the error codes, refer to *Troubleshooting* on page 326.

The integral range must be specified so that it contains at least one measurement data (Shape.Zvalue[X]).

If X does not meet the following condition, an error occurs.

RangeLow ≤((*Shape2D.Resolution* * (X-1)) / LREAL#1000) ≤ *RangeHigh*

X can take any value between 1 and *Shape2D.DataCount*.

In the integral range, the height of each point relative to the base is calculated, and the sum of those heights are used to calculate the area.

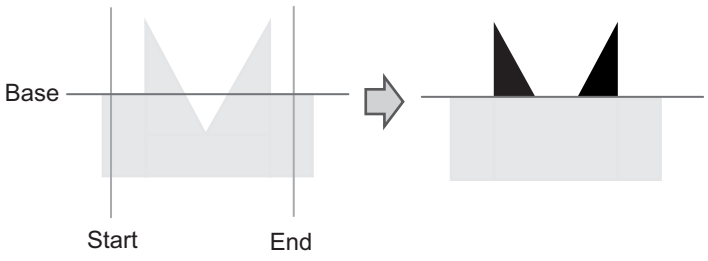
Integration is performed only for heights in the direction specified with *Target* (Target of Integration).

The heights in the opposite direction to *Target* are regarded as the base height.

The following is an example.

If *Target* is FALSE, the concave region is regarded as the base, as shown in the following figure (on the right).

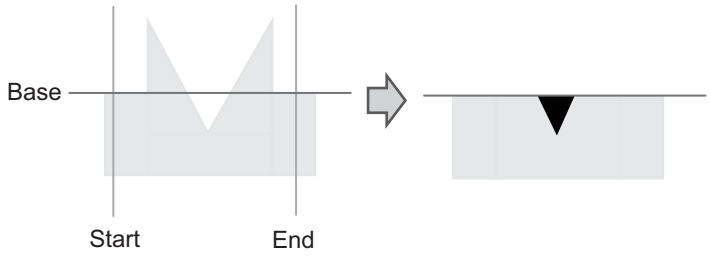
The area of the convex region (black color) is calculated.



If *Target* is TRUE, the convex region is regarded as the base, as shown in the following figure (on the right).

The area of the concave region (black color) is calculated.

The area is output as a negative value.



If there is no height of integration in the integral range, 0 is output to *Area*.

This function block may cause a Task Period Exceeded error because it calculates the feature amount using a maximum of 20,000 2D shape data points.

To avoid this, you need to set the number of processed data points *(NumProcData)* per period.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, take into account the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

If an error occurs during the function block execution, *Error* changes to TRUE and the function block execution is aborted. For details on the error codes, refer to *Troubleshooting* on page 326.

Additional Information

The time required to execute this function block varies depending on the measurement range, resolution of the 2D shape data, or other factors that determine the amount of processed data.

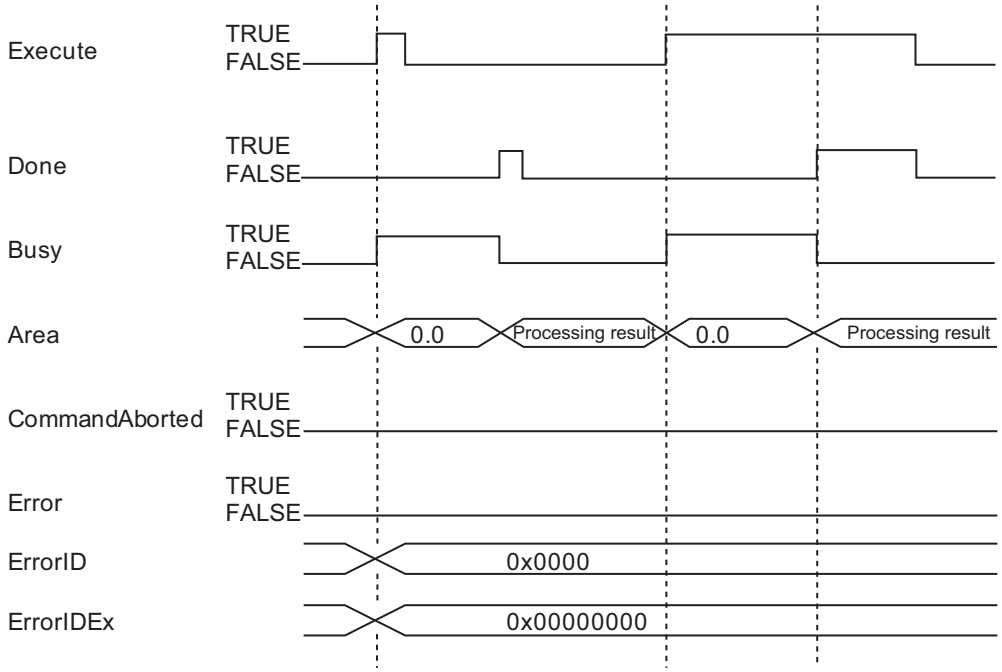
Re-execution of Function Blocks

If *Execute* is changed to TRUE again while execution is in progress (*Busy* = TRUE), the execution is aborted and the processing is performed from the beginning.

Timing Charts

The timing charts are shown below.

● Timing Chart in a Normal State



● Timing Chart When the Function Block Is Aborted

Refer to *Timing Chart When the Function Block Is Aborted* on page 244 of Shape2D_Height.

● Timing Chart When an Error Occurs

Refer to *Timing Chart When an Error Occurs (Startup Error)* on page 245 of Shape2D_Height.

Precautions for Correct Use

Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3CD4	0x0000 0001	The value of Integral Range Upper Limit is not valid.	Set a valid real number.
	0x0000 0002	The value of Integral Range Lower Limit is not valid.	Set a valid real number.
	0x0000 0003	A value specified for the integral range is not valid.	Specify values that can meet the following condition: Integral Range Upper Limit ≥ Integral Range Lower Limit.
	0x0000 0004	There is no shape data in the integral range.	Set the integral range or 2D shape data again.
	0x0000 0005	The number of data points *(DataCount)* of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.
	0x0000 0006	The resolution (*Resolution*) of the 2D shape data is not valid.	Check whether the 2D shape data is created correctly.

Sample Programming

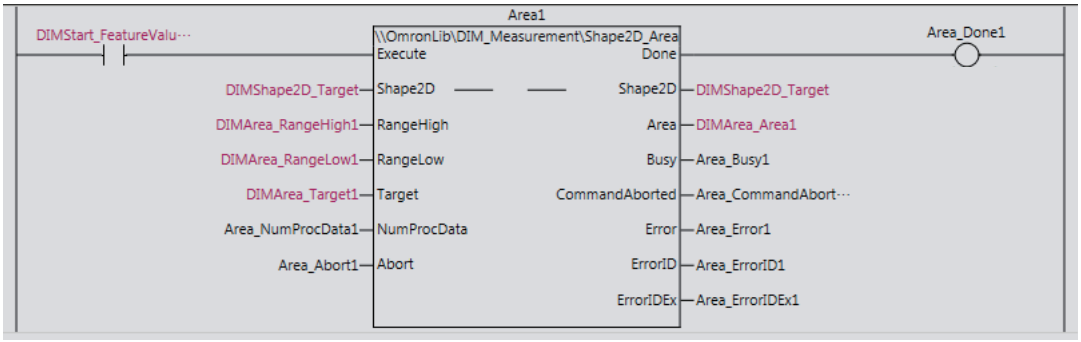
Program Description

This program calculates the area in the specified measurement range of the 2D shape data.

Main Variables

Name	Meaning	Data type	Default	Description
DIMStart_Feature ValueArea	Execution Trigger	BOOL	---	2D shape height measurement is started when this variable changes to TRUE. Assign this variable to the *Execute* input variable of Shape2D_Area.
Area1	2D Shape Sectional Area Measurement	OmronLib\ DIM_Measurement\ Shape2D_Area	---	Instance of the function block that measures the sectional area of the 2D shape data.
DIMShape2D_Target	Target 2D Shape Data	OmronLib\ DIM_Measurement\ sShape2D	---	Set the target 2D shape data. Assign this variable to the *Shape2D* input variable of Shape2D_Area.
DIMArea_RangeHigh1	Integral Range Upper Limit	LREAL	---	Specify the upper limit of the X-axis range that is used for integral calculation. The unit is mm. Assign this variable to the *RangeHigh* input variable of Shape2D_Area.
DIMArea_RangeLow1	Integral Range Lower Limit	LREAL	---	Specify the lower limit of the X-axis range that is used for integral calculation. The unit is mm. Assign this variable to the *RangeLow* input variable of Shape2D_Area.
DIMArea_Target1	Target of Integration	BOOL	---	Specify in which direction (above or below base) the height is recognized as the target of integration. FALSE: Height above base TRUE: Height below base Assign this variable to the *Target* input variable of Shape2D_Area.
Area_NumProcData1	Number of Processed Data Points	UINT	---	Set the number of data points that are processed in a task period during segmented processing. Assign this variable to the *NumProcData* input variable of Shape2D_Area.
DIMArea_Area1	Sectional Area	LREAL	---	Outputs the calculated sectional area. The unit is $mm^{2.}$ Assign this variable to the *Area* output variable of Shape2D_Area.

Ladder Diagram



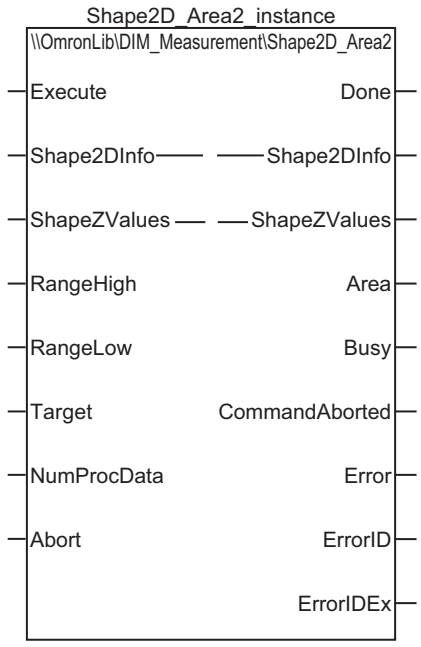
ST

```
//Sectional area calculation
Area1(
   Execute:=DIMStart_FeatureValueArea,
   Shape2D:=DIMShape2D_Target,
   RangeHigh:=DIMArea_RangeHigh1,
   RangeLow:=DIMArea_RangeLow1,
   Target:=DIMArea_Target1,
   NumProcData:=Area_NumProcData1,
   Abort:=Area_Abort1,
   Done=>Area_Done1,
   Area=>DIMArea_Area1,
   Busy=>Area_Busy1,
   CommandAborted=>Area_CommandAborted1,
   Error=>Area_Error1,
   ErrorID=>Area_ErrorID1,
   ErrorIDEx=>Area_ErrorIDEx1);
```

Shape2D_Area2

The Shape2D_Area2 function block calculates the area in the specified range of the variable-length array data. A range of the 2D shape data must be specified as the integral range.

The function block determines the base using shape data within the integral range, and calculates the area (sectional area) of a region bounded by the base and waveform of the 2D shape data.

Function block and function name	Name	FB/ FUN	Graphic expression	ST expression
Shape2D_Area2	2D Shape Sectional Area Measurement 2	FB		Shape2D_Area2_instance(Execute, Shape2DInfo, ShapeZValues, RangeHigh, RangeLow, Target, NumProcData, Abort, Done, Area, Busy, CommandAborted, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V2_0.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00234
Source code	Not published

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
RangeHigh*1	Integral Range Upper Limit	Specify the upper limit of the X-axis range that is used for integral calculation. The unit is mm.	LREAL	Negative number, positive number, or 0	0
RangeLow*1	Integral Range Lower Limit	Specify the lower limit of the X-axis range that is used for integral calculation. The unit is mm.	LREAL	Negative number, positive number, or 0	0
Target*1	Target of Integration	Specify in which direction (above or below base) the height is recognized as the target of integration. FALSE: Height above base TRUE: Height below base	BOOL	TRUE or FALSE	FALSE
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a period during segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
Area	Sectional Area	Outputs the calculated sectional area. The unit is mm^2.	LREAL	Depends on data type.	—
Busy	Measuring	TRUE when the measurement is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 334 for details.

In-Out Variables

Variable	Name	Description	Data type	Valid range
Shape2DInfo	2D Shape Data Information	Specify the 2D shape data information. Refer to *LineMeasure_Cartesian2* on page 151 for 2D shape data creation and correction.	sShape2DInfo	—
ShapeZValues[]*1	2D Shape Data Array	Specify an array where the 2D shape data is stored.	ARRAY[*] OF LREAL	—

*1. The maximum number of array elements is 20,000. The first number of array element should be 0.

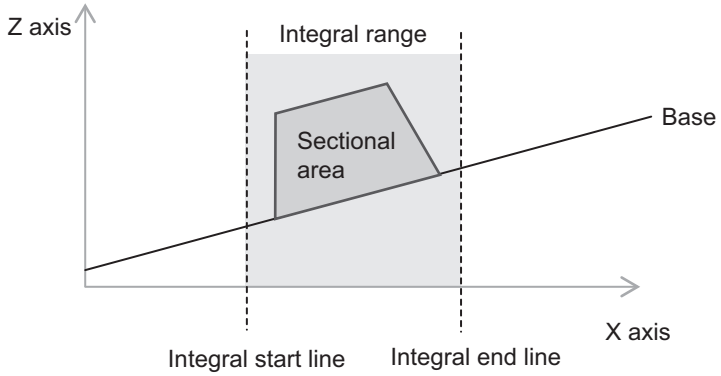
Members of Structure sShape2DInfo

Refer to *Structure* on page 256 of Shape2D_Height2.

Function

The Shape2D_Area2 function block calculates the area in the specified range of the variable-length array data. A range of the 2D shape data must be specified as the integral range.

The function block determines the base using shape data within the integral range, and calculates the area (sectional area) of a region bounded by the base and waveform of the 2D shape data.



(a) The function block checks the input parameters when *Execute* changes to TRUE. If there is no error in the input data, *Busy* (Measuring) changes to TRUE. If there is an error, the function block execution is aborted and the error codes are output to *ErrorID* and *ErrorIDEx*. For details on the error codes, refer to *Troubleshooting* on page 334.

(b) The integral range must be specified so that it contains at least one measurement data (*ShapeZValues[X]*). If X does not meet the following condition, an error occurs.

RangeLow ≤ ((*Shape2DInfo.Resolution* * (X-1)) / LREAL#1000) ≤ *RangeHigh*

X can take any value between 1 and *Shape2D.DataCount*.

(c) In the integral range, the height of each point relative to the base is calculated, and the sum of those heights are used to calculate the area.

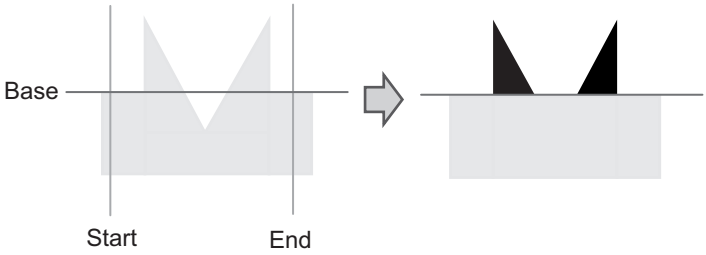
Integration is performed only for heights in the direction specified with *Target* (Target of Integration).

The heights in the opposite direction to Target are regarded as the base height.

The following is an example.

If *Target* is FALSE, the concave region is regarded as the base, as shown in the following figure (on the right).

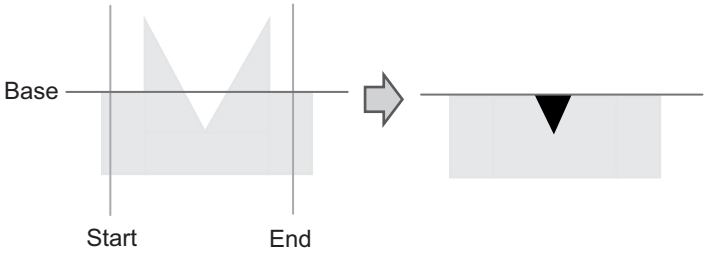
The area of the convex region (black color) is calculated.



If *Target* is TRUE, the convex region is regarded as the base, as shown in the following figure (on the right).

The area of the concave region (black color) is calculated.

The area is output as a negative value.



If there is no height of integration in the integral range, 0 is output to *Area*.

(d) This function block may cause a Task Period Exceeded error because it calculates the feature amount using a maximum of 20,000 2D shape data points.

To avoid this, you need to set the number of processed data points *NumProcData* per period in the task that is executing.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, take into account the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

(e) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

(f) If an error occurs during the function block execution, *Error* changes to TRUE and the function block execution is aborted. For details on the error codes, refer to *Troubleshooting* on page 334.

Additional Information

The time required to execute this function block varies depending on the measurement range, resolution of the 2D shape data, or other factors that determine the amount of processed data.

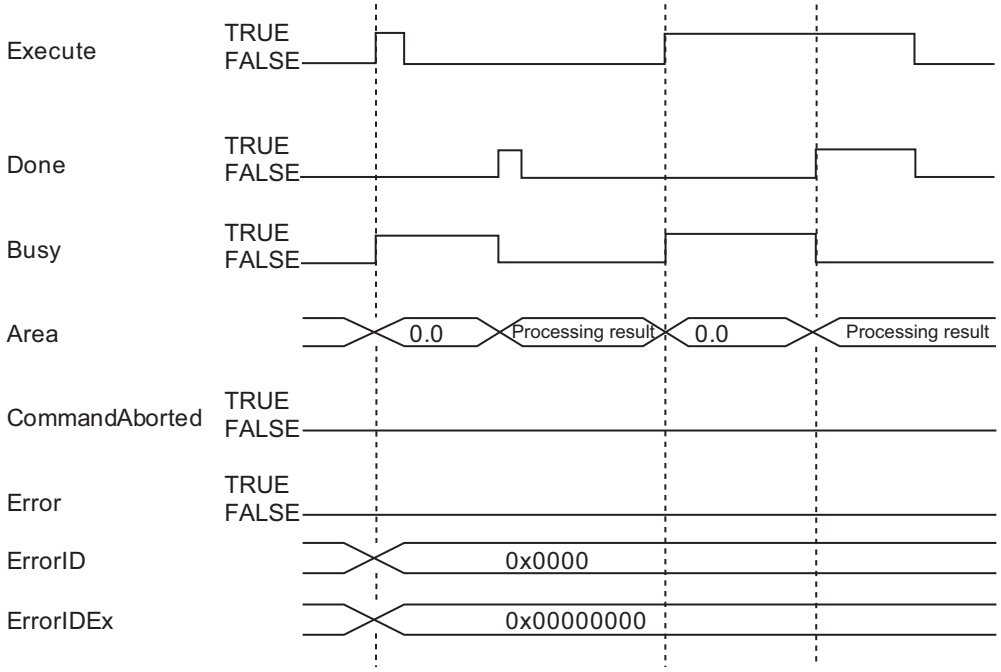
Re-execution of Function Blocks

If *Execute* is changed to TRUE again while execution is in progress (*Busy* = TRUE), the execution is aborted and the processing is performed from the beginning.

Timing Charts

The timing charts are shown below.

● Timing Chart in a Normal End State



● Timing Chart When the Function Block Is Aborted

Refer to *Timing Chart When the Function Block Is Aborted* on page 259 of Shape2D_Height2.

● Timing Chart When an Error Occurs (Startup Error)

Refer to *Timing Chart When an Error Occurs (Startup Error)* on page 260 of Shape2D_Height2.

● Timing Chart When an Error Occurs (Error during Function Block Execution)

Refer to *Timing Chart When an Error Occurs (Error during Function Block Execution)* on page 260 of Shape2D_Height2.

Precautions for Correct Use

Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x0000	0x00000000	Normal end	—
0x3CD4	0x00000001	The value of Integral Range Upper Limit is not valid.	Set a valid real number.
	0x00000002	The value of Integral Range Lower Limit is not valid.	Set a valid real number.
	0x00000003	A value specified for the integral range is not valid.	Specify values that can meet the following condition: Integral Range Upper Limit ≥ Integral Range Lower Limit.
	0x00000004	There is no 2D shape data in the integral range.	Set the integral range or 2D shape data again.
	0x00000005	The number of data points (*DataCount*) of the 2D shape data information is not valid.	Check whether the 2D shape data is created correctly.
	0x00000006	The resolution of the 2D shape data information (*Resolution*) is not valid.	Check whether the 2D shape data is created correctly.
	0x00000007	The specified array for the 2D shape data array (*ShapeZValues*) is out of the valid range.	The first number of array elements should be 0. Specify a value that the number of array elements is 20,000 or less.

Sample Programming

Program Description

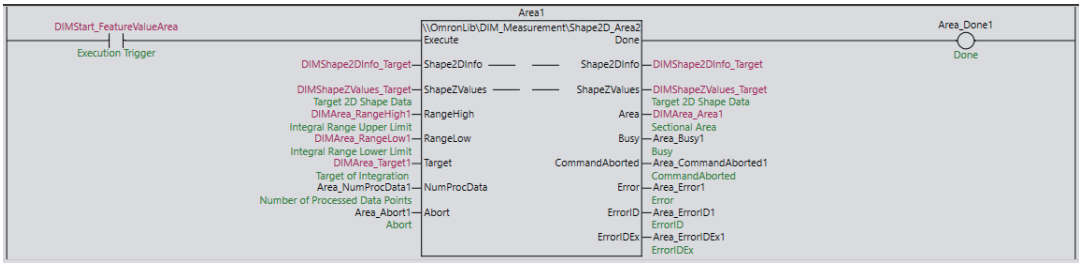
This program calculates the area in the specified measurement range of the 2D shape data.

Main Variables

Variable	Name	Data type	Default	Description
Area1	Sectional Area Measurement FB	OmronLib\DIM_Measurement\Shape2D_Area2	—	Instance of the function block that measures the sectional area of the 2D shape data.
DIMStart_FeatureValueArea	Execution Trigger	BOOL	—	The sectional area measurement for 2D shape is started when this variable changes to TRUE. Assign this variable to the *Execute* input variable of Shape2D_Area2.
DIMShape2DInfo_Target	2D Shape Data Information	OmronLib\DIM_Measurement\sShape2DInfo	—	Set the 2D shape data information. Assign this variable to the *Shape2DInfo* in-out variable of Shape2D_Area2.
DIMShapeZValues_Target	2D Shape Data Array	ARRAY[0..19999] OF LREAL	—	Set the 2D shape data array. Assign this variable to the *ShapeZValues* in-out variable of Shape2D_Area2.

Variable	Name	Data type	Default	Description
DIMArea_RangeHigh1	Integral Range Upper Limit	LREAL	—	Specify the upper limit of the X-axis measurement range 1 that is used for straight line calculation. The unit is mm. Assign this variable to the *RangeHigh* input variable of Shape2D_Area2.
DIMArea_RangeLow1	Integral Range Lower Limit	LREAL	—	Specify the lower limit of the X-axis measurement range 1 that is used for straight line calculation. The unit is mm. Assign this variable to the *RangeLow* input variable of Shape2D_Area2.
DIMArea_Target1	Target of Integration	BOOL	—	Specify in which direction (above or below base) the height is recognized as the target of integration. FALSE: Height above base TRUE: Height below base Assign this variable to the *Target* input variable of Shape2D_Area2.
Area_NumProcData1	Number of Processed Data Points	UINT	—	Set the number of data points that are processed in a task period during segmented processing. Assign this variable to the *NumProcData* input variable of Shape2D_Area2.
DIMArea_Area1	Sectional Area	LREAL	—	Outputs the calculated sectional area. The unit is mm^2. Assign this variable to the *Area* output variable of Shape2D_Area2.

Ladder Diagram



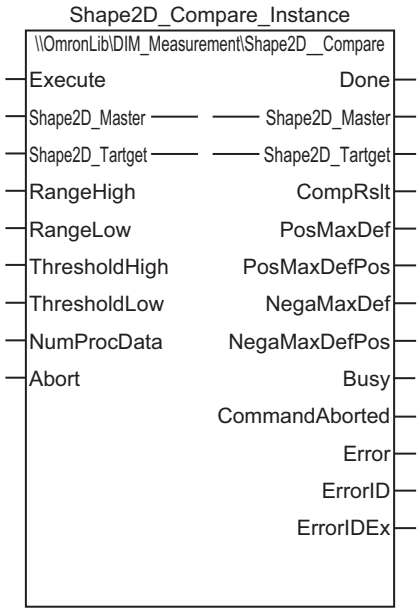
ST

```
//Sectional area calculation
Area1(
   Execute:=DIMStart_FeatureValueArea,
   Shape2DInfo:=DIMShape2DInfo_Target,
   ShapeZValues:=DIMShapeZValues_Target,
   RangeHigh:=DIMArea_RangeHigh1,
   RangeLow:=DIMArea_RangeLow1,
   Target:=DIMArea_Target1,
   NumProcData:=Area_NumProcData1,
   Abort:=Area_Abort1,
   Done=>Area_Done1,
   Area=>DIMArea_Area1,
   Busy=>Area_Busy1,
   CommandAborted=>Area_CommandAborted1,
   Error=>Area_Error1,
   ErrorID=>Area_ErrorID1,
   ErrorIDEx=>Area_ErrorIDEx1);
```

Shape2D_Compare

The Shape2D_Compare function block compares the master 2D shape data and target 2D shape data in the specified measurement range and detects the difference in height (Z direction).

You can set the permissible range to ignore differences within the range.

Function block name	Name	FB/ FUN	Graphic expression	ST expression
Shape2D_ Compare	2D Shape Comparison Measure-ment	FB		Shape2D_Compare_Instance (Execute, Shape2D_Master, Shape2D_Tartget, RangeHigh, RangeLow, ThresholdHigh, ThresholdLow, NumProcData, Abort, Done, CompRslt, PosMaxDef, PosMaxDefPos, NegaMaxDef, NegaMaxDefPos, Busy, CommandAborted, Error, ErrorID, ErrorIDEx)

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V1_1.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00146
Publish/Do not publish source code	Not published.
Function block and function version	1.00

Compatible Models

Item	Product name	Model numbers	Version
Device	Confocal Fiber Displacement Sensor Controller	ZW-8000/7000/5000	-
	Confocal Fiber Displacement Sensor Head	ZW-□□□□	-

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Default
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
RangeHigh*1	Comparison Range Upper Limit	Specify the upper limit of the X-axis range that is used for comparison. The unit is mm.	LREAL	Negative number, positive number, or 0	0
RangeLow*1	Comparison Range Lower Limit	Specify the lower limit of the X-axis range that is used for comparison. The unit is mm.	LREAL	Negative number, positive number, or 0	0
ThresholdHigh*1	Threshold (Positive Maximum Difference)	These thresholds are used to determine whether a difference from the master is within the permissible range. The unit is mm.	LREAL	Positive number or 0	0
ThresholdLow*1	Threshold (Negative Maximum Difference)		LREAL	Positive number or 0	0
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a task period in the segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
CompRslt	Comparison Result	Changes to TRUE when a difference is detected.	BOOL	TRUE or FALSE	—
PosMaxDef	Positive Maximum Difference	Outputs the maximum difference in Z positive direction. The unit is mm. If there is no difference in the positive direction, 0 is output.	LREAL	Depends on data type.	—
PosMaxDefPos	Positive Maximum Difference Position	Outputs the position of the maximum difference in Z positive direction. The unit is mm. If there is no difference in the positive direction, 0 is output.	LREAL	Depends on data type.	—
NegMaxDef	Negative Maximum Difference	Outputs the maximum difference in Z negative direction. The unit is mm. If there is no difference in the negative direction, 0 is output.	LREAL	Depends on data type.	—
NegMaxDefPos	Negative Maximum Difference Position	Outputs the position of the maximum difference in Z negative direction. The unit is mm. If there is no difference in the negative direction, 0 is output.	LREAL	Depends on data type.	—
Busy	Measuring	TRUE when the measurement is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 343.

In-Out Variables

Variable	Name	Description	Data type	Valid range
Shape2D_Master	Master 2D Shape Data	Specify the 2D shape data that is used as the master for comparison. Refer to *LineMeasure_Cartesian* on page 108 for 2D shape data creation and correction.	sShape2D	—
Shape2D_Target	Target 2D shape data	Specify the 2D shape data that is used as the target for comparison. Refer to *LineMeasure_Cartesian* on page 108 for 2D shape data creation and correction.	sShape2D	—

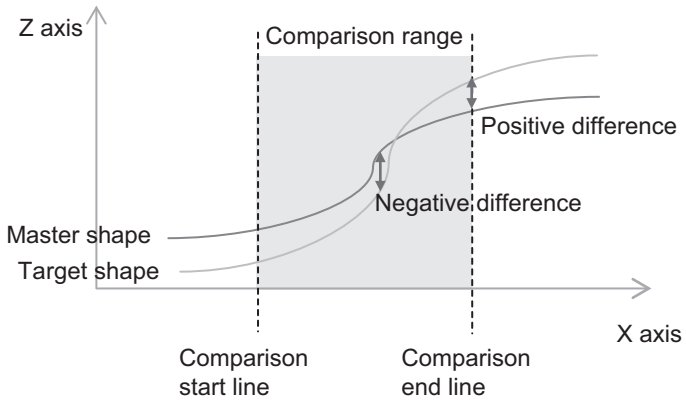
Members of Structure sShape2D

Refer to *Members of Structure sShape2D* on page 241 of Shape2D_Height.

Function

The Shape2D_Compare function block compares the master 2D shape data and target 2D shape data in the specified measurement range and detects the difference in height (Z direction).

You can set the permissible range to ignore differences within the range.



(a) The function block checks the input parameters when *Execute* changes to TRUE. If there is no error in the input data, *Busy* (Measuring) changes to TRUE. If there is an error, the function block execution is aborted and the error codes are output to *ErrorID* and *ErrorIDEx*. For details on the error codes, refer to *Troubleshooting* on page 343.

(b) An error occurs if the X start position or resolution of the master and that of the target do not match.

(c) The comparison range must be specified so that it contains at least one measurement data (Shape.Zvalue[X]).

If X does not meet the following condition, an error occurs.

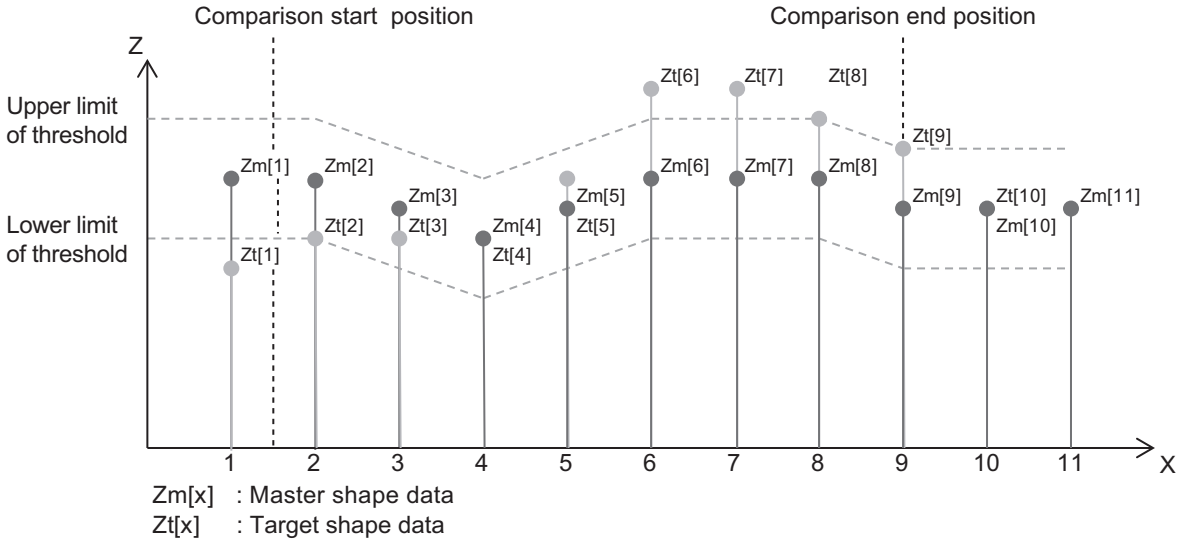
RangeLow ≤((*Shape2D.Resolution* * (X-1)) / LREAL#1000) ≤ *RangeHigh*

X can take any value between 1 and *Shape2D.DataCount*.

(d) The heights of the 2D shape data specified as the master and the heights of the 2D shape data specified as the target are compared at each position in the comparison range.

Differences outside the comparison range are not included in the comparison result.

In the following figure, positions 2 to 9 are included in the comparison range.



Zm[x] : Master shape data
Zt[x] : Target shape data

(e) The permissible range is determined by the master Z value (reference point) at each position, Threshold (Positive Maximum Difference) and Threshold (Negative Maximum Difference).

If there is a difference exceeds the permissible range, Comparison Result changes to TRUE.

If a height of the target is equal to Z ± threshold, the height is included in the permissible range.

In the above figure, Zt[6] and Zt[7] are out of range.

(f) The function block outputs the maximum differences in Z positive and negative directions together with the positions of these differences.

If there are multiple maximum differences, the one with a smaller position is output.

The maximum difference is detected from all data within the comparison range, regardless of the permissible range setting.

In the above figure, the positive maximum difference is Zt[6]-Zm[6], and the negative maximum difference is Zt[2]-Zm[2].

(g) This function block may cause a Task Period Exceeded error because it calculates the feature amount using a maximum of 20,000 2D shape data points.

To avoid this, you need to set the number of processed data points *(NumProcData)* per period.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.

When you decide the number of processed data points, take into account the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

(h) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

(i) If an error occurs during the function block execution, *Error* changes to TRUE and the function block execution is aborted. For details on the error codes, refer to *Troubleshooting* on page 343.

Additional Information

The time required to execute this function block varies depending on the measurement range, resolution of the 2D shape data, or other factors that determine the amount of processed data.

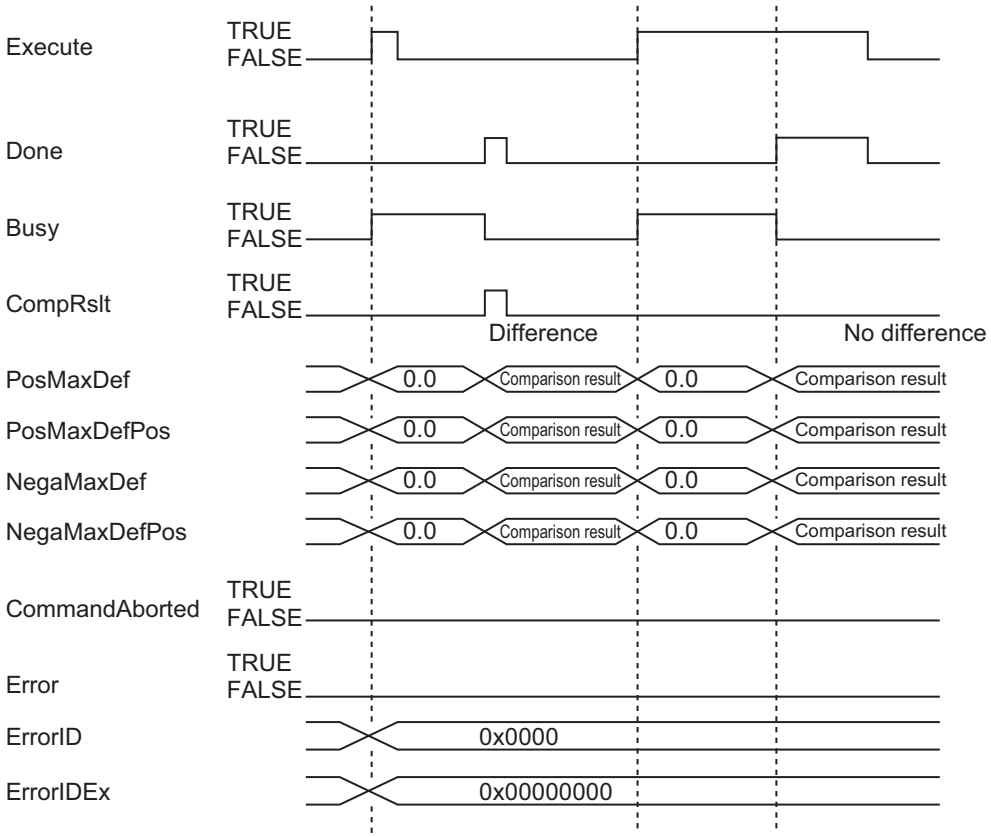
Re-execution of Function Blocks

If *Execute* is changed to TRUE again while execution is in progress (*Busy* = TRUE), the execution is aborted and the processing is performed from the beginning.

Timing Charts

The timing charts are shown below.

● Timing Chart in a Normal State



● Timing Chart When the Function Block Is Aborted

Refer to *Timing Chart When the Function Block Is Aborted* on page 244 of Shape2D_Height.

● Timing Chart When an Error Occurs

Refer to *Timing Chart When an Error Occurs (Startup Error)* on page 245 of Shape2D_Height.

Precautions for Correct Use

Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x3CD5	0x0000 0001	The value of Comparison Range Upper Limit is not valid.	Set a valid real number.
	0x0000 0002	The value of Comparison Range Lower Limit is not valid.	Set a valid real number.
	0x0000 0003	A value specified for the comparison range is not valid.	Specify values that can meet the following condition: Comparison Range Upper Limit ≥ Comparison Range Lower Limit.
	0x0000 0004	The value of Threshold (Positive Maximum Difference) is not valid.	Set a valid real number.
	0x0000 0005	The value of Threshold (Negative Maximum Difference) is not valid.	Set a valid real number.
	0x0000 0007	The resolutions do not match.	Set the shape data of the master or target again.
	0x0000 0008	There is no shape data in the comparison range. (master)	Set the comparison range or master shape data again.
	0x0000 0009	There is no shape data in the comparison range. (target)	Set the comparison range or target shape data again.
	0x0000 000A	The number of data points *(DataCount)* of the 2D shape data is not valid. (master)	Check whether the master 2D shape data is created correctly.
	0x0000 000B	The number of data points *(DataCount)* of the 2D shape data is not valid. (target)	Check whether the target 2D shape data is created correctly.
	0x0000 000C	The resolution (*Resolution*) of the 2D shape data is not valid. (master)	Check whether the master 2D shape data is created correctly.
	0x0000 000D	The resolution (*Resolution*) of the 2D shape data is not valid. (target)	Check whether the target 2D shape data is created correctly.

Sample Programming

Program Description

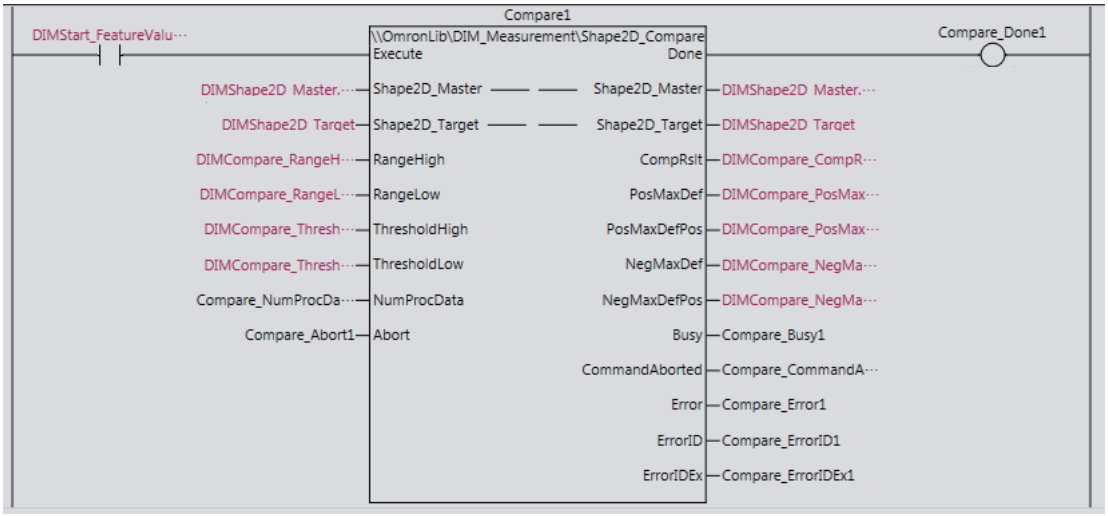
This program compares the master and target 2D shape data in the specified measurement range and calculates the difference.

Main Variables

Name	Meaning	Data type	Default	Description
DIMStart_Feature ValueCompare	Execution Trigger	BOOL	---	2D shape height measurement is started when this variable changes to TRUE. Assign this variable to the *Execute* input variable of Shape2D_Compare.
Compare1	2D Shape Comparison Measure-ment FB	OmronLib\ DIM_Mea-surement\ Shape2D_-Compare	---	Instance of the function block that compares the master and target 2D shape data in the specified measurement range, and detects the difference of the height.
DIMShape2D_Master	Master 2D Shape Data	OmronLib\ DIM_Mea-surement\ sShape2D	---	Set the master 2D shape data. Assign this variable to the *Shape2D_Master* input vari-able of Shape2D_Compare.
DIMShape2D_Target	Target 2D Shape Data	OmronLib\ DIM_Mea-surement\ sShape2D	---	Set the target 2D shape data. Assign this variable to the *Shape2D_Target* input vari-able of Shape2D_Compare.
DIMCompare_Range High1	Comparison Range Upper Limit	LREAL	---	Specify the upper limit of the X-axis range that is used for comparison. The unit is mm. Assign this variable to the *RangeHigh* input variable of Shape2D_Compare.
DIMCompare_Range Low1	Comparison Range Lower Limit	LREAL	---	Specify the lower limit of the X-axis range that is used for comparison. The unit is mm. Assign this variable to the *RangeLow* input variable of Shape2D_Compare.
DIMCompare _ThresholdHigh1	Threshold (Positive Maximum Difference)	LREAL	---	Set the threshold range. The unit is mm. Assign this variable to the *ThresholdHigh* input variable of Shape2D_Compare.
DIMCompare _ThresholdLow1	Threshold (Negative Maximum Difference)	LREAL	---	Set the threshold range. The unit is mm. Assign this variable to the *ThresholdLow* input variable of Shape2D_Compare.
DIMCompare_Comp Rslt1	Comparison Result	BOOL	---	Changes to TRUE when a difference out of the allowable range is detected.
DIMCompare_Pos MaxDef1	Positive Maximum Difference	LREAL	---	Outputs the maximum difference in Z positive direction. The unit is mm. If there is no differ-ence in the positive direction, 0 is output.
DIMCompare_Pos MaxDefPos1	Positive Maximum Difference Position	LREAL	---	Outputs the position of the maximum differ-ence in Z positive direction. The unit is mm. If there is no difference in the positive direction, 0 is output.

Name	Meaning	Data type	Default	Description
DIMCompare_NegaMaxDef1	Negative Maximum Difference	LREAL	---	Outputs the maximum difference in Z negative direction. The unit is mm. If there is no difference in the negative direction, 0 is output.
DIMCompare_NegaMaxDefPos1	Negative Maximum Difference Position	LREAL	---	Outputs the position of the maximum difference in Z negative direction. The unit is mm. If there is no difference in the negative direction, 0 is output.

Ladder Diagram



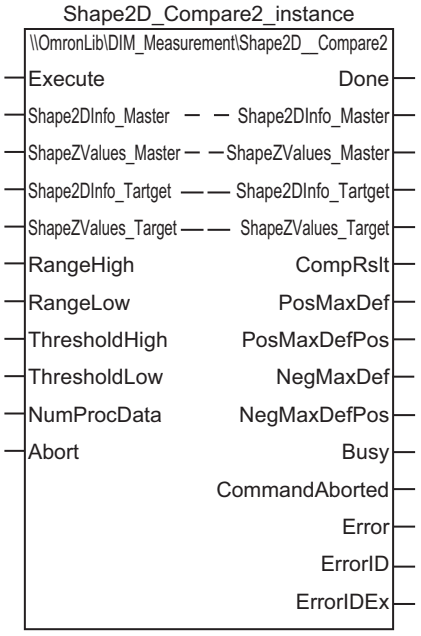
ST

```
//Comparison calculation
Compare1(
   Execute:=DIMStart_FeatureValueCompare,
   Shape2D_Master:=DIMShape2D_Master.Shape2D,
   Shape2D_Target:=DIMShape2D_Target,
   RangeHigh:=DIMCompare_RangeHigh1,
   RangeLow:=DIMCompare_RangeLow1,
   ThresholdHigh:=DIMCompare_ThresholdHigh1,
   ThresholdLow:=DIMCompare_ThresholdLow1,
   NumProcData:=Compare_NumProcData1,
   Abort:=Compare_Abort1,
   Done=>Compare_Done1,
   CompRslt=>DIMCompare_CompRslt1,
   PosMaxDef=>DIMCompare_PosMaxDef1,
   PosMaxDefPos=>DIMCompare_PosMaxDefPos1,
   NegMaxDef=>DIMCompare_NegMaxDef1,
   NegMaxDefPos=>DIMCompare_NegMaxDefPos1,
   Busy=>Compare_Busy1,
   CommandAborted=>Compare_CommandAborted1,
   Error=>Compare_Error1,
   ErrorID=>Compare_ErrorID1,
   ErrorIDEx=>Compare_ErrorIDEx1);
```

Shape2D_Compare2

The Shape2D_Compare2 function block compares the master 2D shape data and target 2D shape data in the specified measurement range and detects the difference in height (Z direction). Master and target 2D shape data are variable-length arrays.

You can set the permissible range to ignore differences within the range.

Function block and function name	Name	FB/ FUN	Graphic expression	ST expression
Shape2D_Compare2	2D Shape Comparison Measurement 2	FB		Shape2D_Compare2_instance(Execute, Shape2DInfo_Master, ShapeZValues_Master, Shape2DInfo_Target, ShapeZValues_Target, RangeHigh, RangeLow, ThresholdHigh, ThresholdLow, NumProcData, Abort, Done, CompRslt, PosMaxDef, PosMaxDefPos, NegMaxDef, NegMaxDefPos, Busy, CommandAborted, Error, ErrorID, ErrorIDEx);

Function Block and Function Information

Item	Description
Library file name	OmronLib_DIM_LineMeasurement_V2_0.slr
Namespace	OmronLib\DIM_Measurement
Function block and function number	00235
Source code	Not published

Variables

Input Variables

Variable	Name	Description	Data type	Valid range	Initial value
Execute	Execute	Executes the function block when the value is changed to TRUE.	BOOL	TRUE or FALSE	FALSE
RangeHigh*1	Comparison Range Upper Limit	Specify the upper limit of the X-axis range that is used for comparison. The unit is mm.	LREAL	Negative number, positive number, or 0	0
RangeLow*1	Comparison Range Lower Limit	Specify the lower limit of the X-axis range that is used for comparison. The unit is mm.	LREAL	Negative number, positive number, or 0	0
ThresholdHigh*1	Threshold (Positive Maximum Difference)	These thresholds are used to determine whether a difference between the master and target shapes is within the permissible range. The unit is mm.	LREAL	Positive number or 0	0
ThresholdLow*1	Threshold (Negative Maximum Difference)		LREAL	Positive number or 0	0
NumProcData*1	Number of Processed Data Points	Specify the number of data points that are processed in a period during segmented processing.	UINT	Depends on data type.	100
Abort	Abort	Aborts the function block execution.	BOOL	TRUE or FALSE	FALSE

*1. When the function block is executed, the parameter is copied internally. Therefore, any parameter change made during execution is not reflected in processing.

To change the parameter, you need to re-execute the function block.

Output Variables

Variable	Name	Description	Data type	Valid range	Default
Done	Done	TRUE when the function block execution is completed.	BOOL	TRUE or FALSE	—
CompRslt	Comparison result	Changes to TRUE when a difference is detected.	BOOL	TRUE or FALSE	—
PosMaxDef	Positive Maximum Difference	Outputs the maximum difference in Z positive direction. The unit is mm. If there is no difference in the positive direction, 0 is output.	LREAL	Depends on data type.	—
PosMaxDef Pos	Positive Maximum Difference Position	Outputs the position of the maximum difference in Z positive direction. The unit is mm. If there is no difference in the positive direction, 0 is output.	LREAL	Depends on data type.	—

Variable	Name	Description	Data type	Valid range	Default
NegMaxDef	Negative Maximum Difference	Outputs the maximum difference in Z negative direction. The unit is mm. If there is no difference in the negative direction, 0 is output.	LREAL	Depends on data type.	—
NegMaxDef Pos	Negative Maximum Difference Position	Outputs the position of the maximum difference in Z negative direction. The unit is mm. If there is no difference in the negative direction, 0 is output.	LREAL	Depends on data type.	—
Busy	Measuring	TRUE when the measurement is in progress.	BOOL	TRUE or FALSE	—
Command Aborted	Interruption Completion	TRUE when the function block execution is aborted.	BOOL	TRUE or FALSE	—
Error	Error	TRUE while there is an error.	BOOL	TRUE or FALSE	—
ErrorID	Error Code	Outputs the error code when an error occurs.	WORD	*1	—
ErrorIDEx	Expansion Error Code	Outputs the expansion error code when an error occurs.	DWORD	*1	—

*1. Refer to *Troubleshooting* on page 352 for details.

In-Out Variables

Variable	Name	Description	Data type	Valid range
Shape2DInfo_Master	Master 2D Shape Data Information	Specify the 2D shape data information that is used as the master for comparison. Refer to *LineMeasure_Cartesian2* on page 151 for 2D shape data creation and correction.	sShape2DInfo	—
ShapeZValues_Master[]*1	Master 2D Shape Data Array	Specify an array where the 2D shape data used as the master for comparison is stored.	ARRAY[*] OF LREAL	—
Shape2DInfo_Target	Target 2D Shape Data Information	Specify the 2D shape data information that is used as the target for comparison. Refer to *LineMeasure_Cartesian2* on page 151 for 2D shape data creation and correction.	sShape2DInfo	—
ShapeZValues_Target[]*1	Target 2D Shape Data Array	Specify an array where the 2D shape data used as the target for comparison is stored.	ARRAY[*] OF LREAL	—

*1. The maximum number of array elements is 20,000. The first number of array element should be 0.

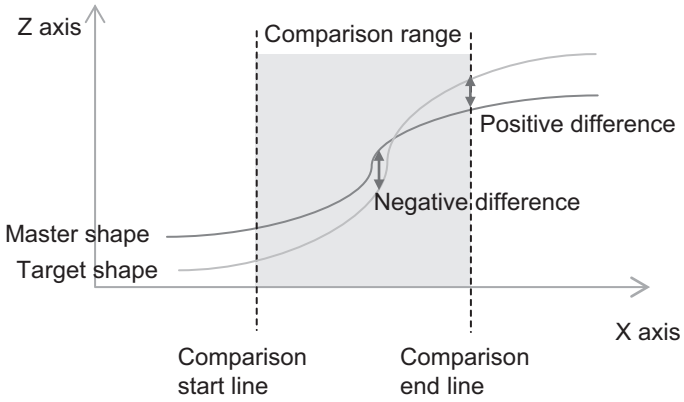
Members of Structure sShape2DInfo

Refer to *Structure* on page 256 of Shape2D_Height2.

Function

The Shape2D_Compare2 function block compares the master 2D shape data and target 2D shape data in the specified measurement range and detects the difference in height (Z direction). Master and target 2D shape data are variable-length arrays.

You can set the permissible range to ignore differences within the range.



(a) The function block checks the input parameters when *Execute* changes to TRUE. If there is no error in the input data, *Busy* (Measuring) changes to TRUE. If there is an error, the function block execution is aborted and the error codes are output to *ErrorID* and *ErrorIDEx*. For details on the error codes, refer to *Troubleshooting* on page 352.

(b) An error occurs if the X start position or resolution of the master and that of the target do not match.

(c) The comparison range must be specified so that it contains at least one measurement data (*Shape.Zvalue[X]*).

If X does not meet the following condition, an error occurs.

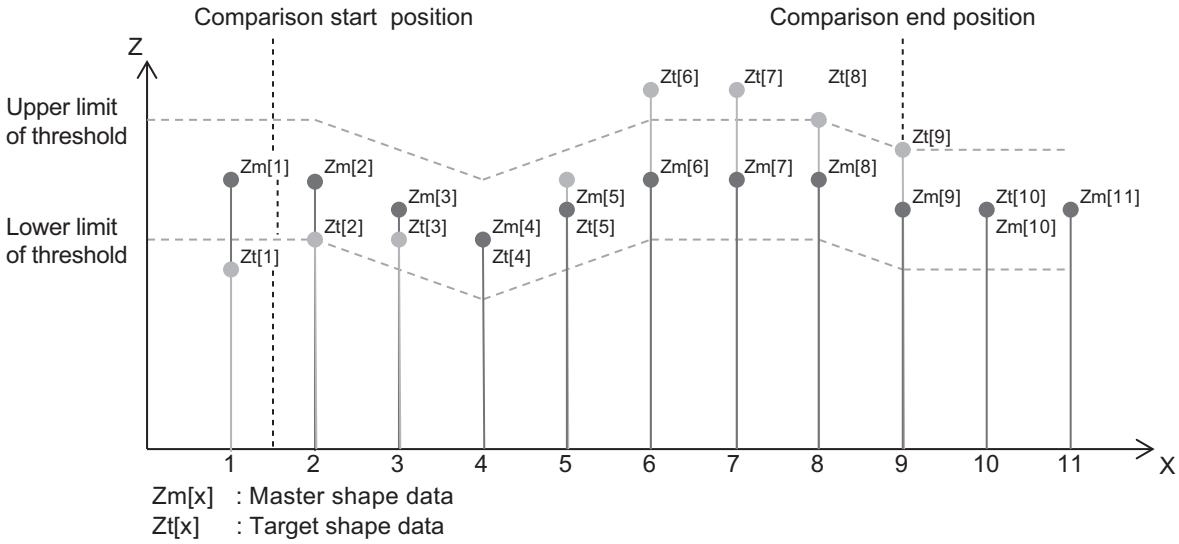
RangeLow ≤ ((*Shape2DInfo.Resolution* * (X-1)) / LREAL#1000) ≤ *RangeHigh*

X can take any value between 1 and *Shape2DInfo.DataCount*

(d) The heights of the 2D shape data specified as the master and the heights of the 2D shape data specified as the target are compared at each position in the comparison range.

Differences outside the comparison range are not included in the comparison result.

In the following figure, positions 2 to 9 are included in the comparison range.



(e) The permissible range is determined by the master Z value (reference point) at each position, Threshold (Positive Maximum Difference) and Threshold (Negative Maximum Difference).

If there is a difference exceeds the permissible range, Comparison Result changes to TRUE.

If a height of the target is equal to Z ± threshold, the height is included in the permissible range.

In the above figure, Zt[6] and Zt[7] are out of range.

(f) The function block outputs the maximum differences in Z positive and negative directions together with the positions of these differences.

If there are multiple maximum differences, the one with a smaller position is output.

The maximum difference is detected from all data within the comparison range, regardless of the permissible range setting.

In the above figure, the positive maximum difference is Zt[6]-Zm[6], and the negative maximum difference is Zt[2]-Zm[2].

(g) This function block may cause a Task Period Exceeded error because it calculates the feature amount using a maximum of 20,000 2D shape data points.

To avoid this, you need to set the number of processed data points *NumProcData* per period in the task that is executing.

If you increase the number of processed data points, the time until completion can be reduced. However, doing so increases the processing time per period.

If you decrease the number of processed data points, the processing time per period can be reduced. However, doing so increases the time until completion.
When you decide the number of processed data points, take into account the execution environment, unused time in the task period available when correction is executed, and other factors.

If the number of processed data points is set to 0, the function block execution does not end until all correction and conversion are completed.

If this function block is assigned to the event task, the number of processed data points is ignored and the function block execution also does not end until all correction and conversion are completed.

(h) If *Abort* is changed to TRUE, *CommandAborted* changes to TRUE and the function block execution is aborted.

If *Execute* and *Abort* change to TRUE at the same time, no operation is performed and *CommandAborted* changes to TRUE.

(i) If an error occurs during the function block execution, *Error* changes to TRUE and the function block execution is aborted. For details on the error codes, refer to *Troubleshooting* on page 352.

Additional Information

The time required to execute this function block varies depending on the measurement range, resolution of the 2D shape data, or other factors that determine the amount of processed data.

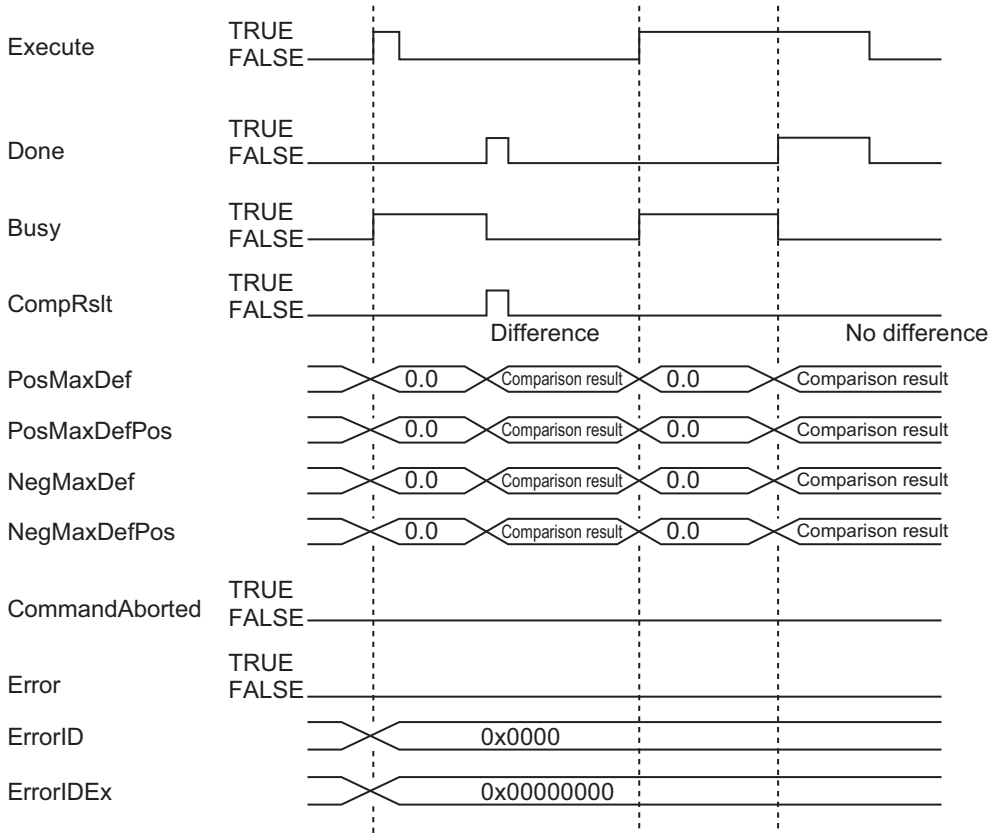
Re-execution of Function Blocks

If *Execute* is changed to TRUE again while execution is in progress (*Busy* = TRUE), the execution is aborted and the processing is performed from the beginning.

Timing Charts

The timing charts are shown below.

● Timing Chart in a Normal End State



● Timing Chart When the Function Block Is Aborted

Refer to *Timing Chart When the Function Block Is Aborted* on page 259 of Shape2D_Height2.

● Timing Chart When an Error Occurs (Startup Error)

Refer to *Timing Chart When an Error Occurs (Startup Error)* on page 260 of Shape2D_Height2.

● Timing Chart When an Error Occurs (Error during Function Block Execution)

Refer to *Timing Chart When an Error Occurs (Error during Function Block Execution)* on page 260 of Shape2D_Height2.

Precautions for Correct Use

Before executing this function block, carefully read the manuals for devices to use and ensure the safety for use.

Troubleshooting

Error code	Expansion error code	Description	Corrective action
0x0000	0x00000000	Normal end	—
0x3CD5	0x00000001	The value of Comparison Range Upper Limit is not valid.	Set a valid real number.
	0x00000002	The value of Comparison Range Lower Limit is not valid.	Set a valid real number.
	0x00000003	A value specified for the comparison range is not valid.	Specify values that can meet the following condition: Comparison Range Upper Limit ≥ Comparison Range Lower Limit.
	0x00000004	The value of Threshold (Positive Maximum Difference) is not valid.	Set a valid real number.
	0x00000005	The value of Threshold (Negative Maximum Difference) is not valid.	Set a valid real number.
	0x00000007	The resolutions do not match.	Set the shape data of the master or target again.
	0x00000008	There is no shape data in the comparison range. (master)	Set the comparison range or master shape data again.
	0x00000009	There is no shape data in the comparison range. (target)	Set the comparison range or target shape data again.
	0x0000000A	The number of data points (*DataCount*) of the 2D shape data information is not valid. (master)	Check whether the master 2D shape data is created correctly.
	0x0000000B	The number of data points (*DataCount*) of the 2D shape data information is not valid. (target)	Check whether the target 2D shape data is created correctly.
	0x0000000C	The resolution (*Resolution*) of the 2D shape data information is not valid. (master)	Check whether the master 2D shape data is created correctly.
	0x0000000D	The resolution (*Resolution*) of the 2D shape data information is not valid. (target)	Check whether the target 2D shape data is created correctly.
	0x0000000E	The specified array for the master 2D shape data array (*ShapeZValues_Master*) is out of the valid range.	The first number of array elements should be 0. Specify a value that the number of array elements is 20,000 or less.
	0x0000000F	The specified array for the target 2D shape data array (*ShapeZValues_Target*) is out of the valid range.	The first number of array elements should be 0. Specify a value that the number of array elements is 20,000 or less.

Sample Programming

Program Description

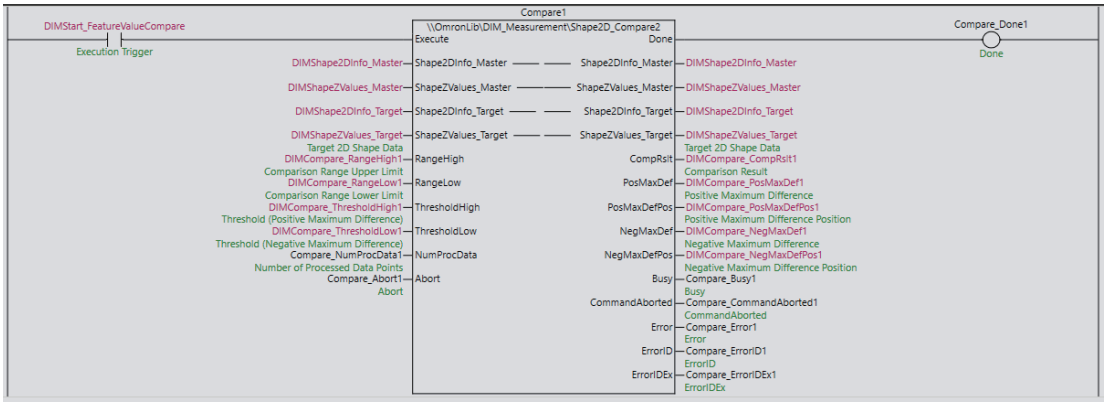
This program compares the master and target 2D shape data in the specified measurement range and calculates the difference.

Main Variables

Name	Meaning	Data type	Default	Description
Compare1	2D Shape Comparison Measurement FB	OmronLib\DIM_Measurement\Shape2D_Height2	—	Instance of the function block that compares the master and target 2D shape data in the specified measurement range, and detects the difference of the height.
DIMStart_FeatureValueCompare	Execution Trigger	BOOL	—	The comparison calculation starts when the variable changes to TRUE. Assign this variable to the *Execute* input variable of Shape2D_Compare2.
DIMShape2DInfo_Master	Master 2D Shape Data Information	OmronLib\DIM_Measurement\sShape2DInfo	—	Set the master 2D shape data information. Assign this variable to the *Shape2DInfo* in-out variable of Shape2D_Compare2.
DIMShape2DInfo_Target	Target 2D Shape Data Information	OmronLib\DIM_Measurement\sShape2DInfo	—	Set the target 2D shape data information. Assign this variable to the *Shape2DInfo* in-out variable of Shape2D_Compare2.
DIMShapeZValues_Master	Master 2D Shape Data Array	ARRAY[0..19999] OF LREAL	—	Set the master 2D shape data array. Assign this variable to the *ShapeZValues* in-out variable of Shape2D_Compare2.
DIMShapeZValues_Target	Target 2D Shape Data Array	ARRAY[0..19999] OF LREAL	—	Set the target 2D shape data array. Assign this variable to the *ShapeZValues* in-out variable of Shape2D_Compare2.
DIMCompare_RangeHigh1	Comparison Range Upper Limit	LREAL	—	Specify the upper limit of the X-axis range that is used for comparison. The unit is mm. Assign this variable to the *RangeHigh* input variable of Shape2D_Compare2.
DIMCompare_RangeLow1	Comparison Range Lower Limit	LREAL	—	Specify the lower limit of the X-axis range that is used for comparison. The unit is mm. Assign this variable to the *RangeLow* input variable of Shape2D_Compare2.
DIMCompare_ThresholdHigh1	Threshold (Positive Maximum Difference)	LREAL	—	Set the threshold range. The unit is mm. Assign this variable to the *ThresholdHigh* input variable of Shape2D_Compare2.
DIMCompare_ThresholdLow1	Threshold (Negative Maximum Difference)	LREAL	—	Set the threshold range. The unit is mm. Assign this variable to the *ThresholdLow* input variable of Shape2D_Compare2.
DIMCompare_CompRslt1	Comparison Result	BOOL	—	Changes to TRUE when a difference out of the allowable range is detected.

Name	Meaning	Data type	Default	Description
DIMCompare_PosMaxDef1	Positive Maximum Difference	LREAL	—	Outputs the maximum difference in Z positive direction. The unit is mm. If there is no difference in the positive direction, 0 is output.
DIMCompare_PosMaxDefPos1	Positive Maximum Difference Position	LREAL	—	Outputs the position of the maximum difference in Z positive direction. The unit is mm. If there is no difference in the positive direction, 0 is output.
DIMCompare_NegMaxDef1	Negative Maximum Difference	LREAL	—	Outputs the maximum difference in Z negative direction. The unit is mm. If there is no difference in the negative direction, 0 is output.
DIMCompare_NegMaxDefPos1	Negative Maximum Difference Position	LREAL	—	Outputs the position of the maximum difference in Z negative direction. The unit is mm. If there is no difference in the negative direction, 0 is output.

Ladder Diagram



ST

```
//Comparison calculation
Compare1(
   Execute:=DIMStart_FeatureValueCompare,
   Shape2DInfo_Master:=DIMShape2DInfo_Master,
   ShapeZValues_Master:=DIMShapeZValues_Master,
   Shape2DInfo_Target:=DIMShape2DInfo_Target,
   ShapeZValues_Target:=DIMShapeZValues_Target,
   RangeHigh:=DIMCompare_RangeHigh1,
   RangeLow:=DIMCompare_RangeLow1,
   ThresholdHigh:=DIMCompare_ThresholdHigh1,
   ThresholdLow:=DIMCompare_ThresholdLow1,
   NumProcData:=Compare_NumProcData1,
   Abort:=Compare_Abort1,
   Done=>Compare_Done1,
   CompRslt=>DIMCompare_CompRslt1,
   PosMaxDef=>DIMCompare_PosMaxDef1,
   PosMaxDefPos=>DIMCompare_PosMaxDefPos1,
   NegMaxDef=>DIMCompare_NegMaxDef1,
   NegMaxDefPos=>DIMCompare_NegMaxDefPos1,
   Busy=>Compare_Busy1,
   CommandAborted=>Compare_CommandAborted1,
   Error=>Compare_Error1,
   ErrorID=>Compare_ErrorID1,
   ErrorIDEx=>Compare_ErrorIDEx1);
```

Appendix

Referring to Library Information

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

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

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

- Attributes of libraries

  Information for identifying the library itself
- Attributes of function blocks and functions

  Information for identifying the function block and function contained in the library

Use the Sysmac Studio to access the library information.

Attributes of Libraries, Function Blocks and Functions

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

● Attributes of Libraries

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

*1. These numbers correspond to the numbers shown on the screen images in the next section, *Referring to Attributes of Libraries, Function Blocks and Functions* on page 359.

*2. It is provided in English and Japanese.

● Attributes of Function Blocks and Functions

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

*1. These numbers correspond to the numbers shown on the screen images in the next section, *Referring to Attributes of Libraries, Function Blocks and Functions* on page 359.

*2. It is provided in English and Japanese.

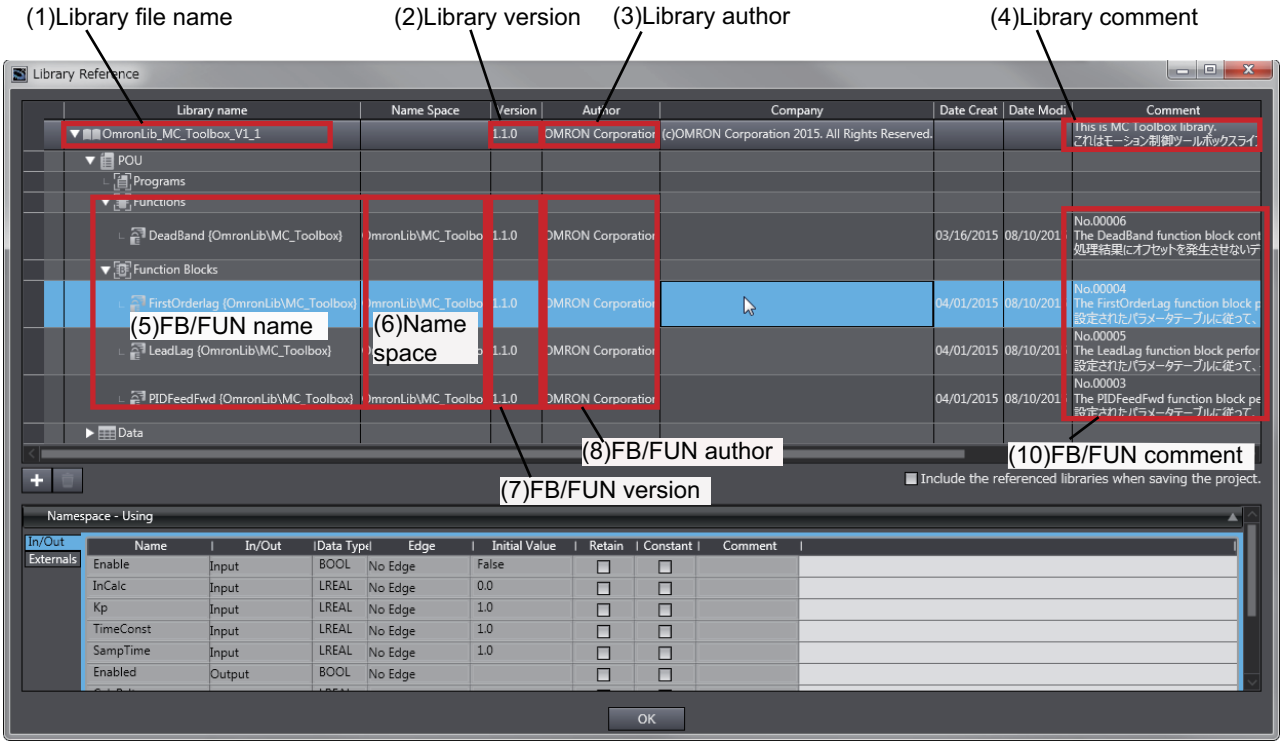
Referring to Attributes of Libraries, Function Blocks and Functions

You can refer to the attributes of libraries, function blocks and functions of the library information at the following locations on the Sysmac Studio.

- Library Reference Dialog Box
- Toolbox Pane
- Ladder Editor

(a) Library Reference Dialog Box

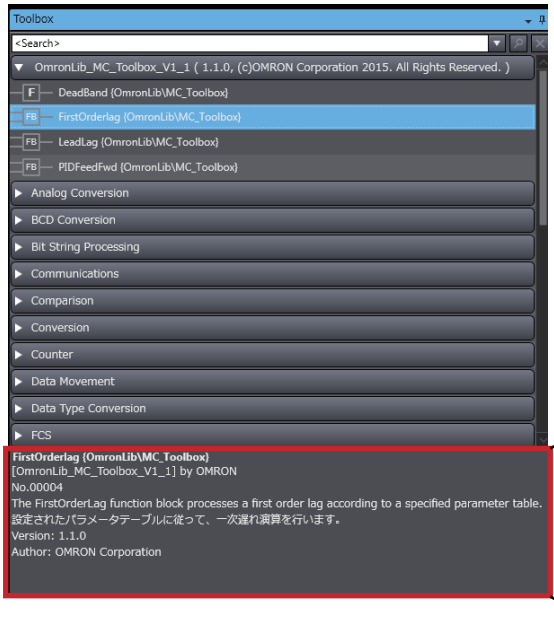
When you refer to the libraries, the library information is displayed at the locations shown below.



(b) Toolbox Pane

Select a function block and function to display its library information at the bottom of the Toolbox Pane.

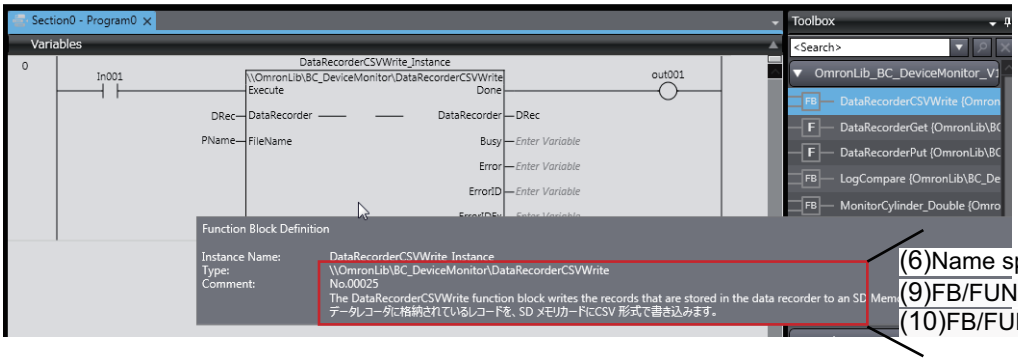
The text “by OMRON” which is shown on the right of the library name (1) indicates that this library was provided by OMRON.



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

(c) Ladder Editor

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



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

Referring to Function Block and Function Source Codes

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

User function blocks and user functions can be created based on the copies of these source codes.

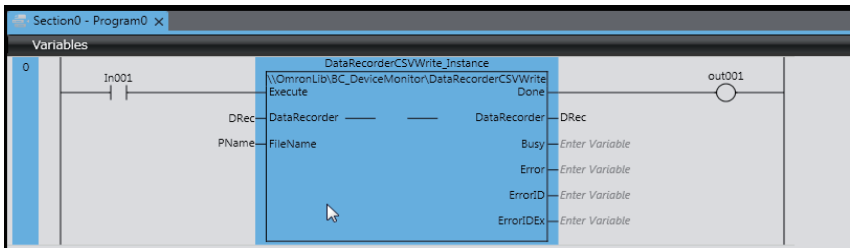
The following are the examples of items that you may need to customize.

- Customizing the size of arrays to suit the memory capacity of the user's Controller
- Customizing the data types to suit the user-defined data types

Note that you can access only function blocks and functions whose Source code published/not published is set to Published in the library information shown in their individual specifications.

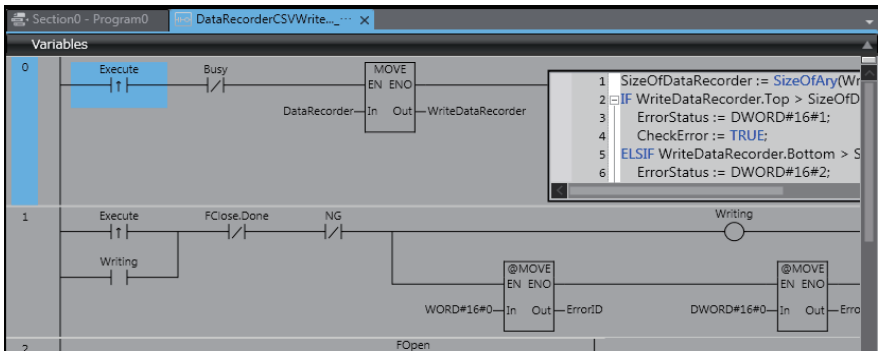
Use the following procedure to refer to the source codes of function blocks and functions.

1 Select a function block or function in the program.



2 Double-click or right-click and select **To Lower Layer** from the menu.

The source code is displayed.



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.

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 2016-2019 All Rights Reserved.
In the interest of product improvement,
specifications are subject to change without notice.

Cat. No. W574-E1-04 0119